Techniques in Operations Research
MAST30013

Dr Charl Ras

cjras@unimelb.edu.au

1 Motivation

2 Unconstrained Single Variable Optimisation

3 Unconstrained Optimisation with n Variables

4 General Constrained Optimisation

5 Convex Optimisation

Section 1

Motivation

What is Operations Research?

Operations Research (OR) and the Management Sciences (MS) are the professional disciplines that deal with the application of information technology for informed decision-making.
- *Institute for Operations Research and Management Sciences, www.informs.org*

A scientific approach to decision making, which seeks to determine how best to design and operate a system, usually under conditions requiring the allocation of scarce resources
- *from the textbook Operations Research, Winston*

Mathematics of Operations Research

- Linear programming (covered in MAST20018)
- Non-linear programming (covered in MAST30013)
- Integer programming
- Decision-making under uncertainty (random processes, statistics, queueing theory, simulation)

Operations Research at Melbourne University

Some projects:

- Communication network design and control
- Open pit/underground mine optimisation
- Modelling of patient movements in hospitals
- Mathematical models of complex queues
- Transportation and logistics

Optimisation

Optimisation is an important branch of Operations Research, concerned with finding minimum (or maximum) values of functions of one or more variables.

Minimise (or Maximise) **Objective Function** $f(x),\ f : X \to \mathbb{R}$ subject to **Constraints defining** X

- The set $X \subseteq \mathbb{R}^n$ contains the "feasible decisions".
- The value $f(x)$ for each $x \in X$ is a scalar measure of the "undesirability" of choosing decision x.
- The aim is to find an $x^* \in X$ such that

$$f(x^*) \leq f(x), \quad \text{for all } x \in X.$$

This course: continuous optimisation

- Each $x \in X$ is an n-dimensional vector, so $X \subseteq \mathbb{R}^n$.
- The components of $x \in X$ represent the *decision* or *control* variables.
- If $X = \mathbb{R}^n$, then we have an *unconstrained optimisation problem*.
- If X is a strict subset of $\mathbb{R}^n$, then we have a *constrained optimisation problem*.

Linear Programming

Linear optimisation, also known as *linear programming*, describes an optimisation problem in which

- the cost function f is linear, and
- the constraint set X is specified by linear equations and inequalities.

These problems (LPs) can be solved exactly by efficient discrete methods (such as the Simplex Algorithm).

Nonlinear Optimisation Definition

Nonlinear optimisation, also known as *nonlinear programming*, describes an optimisation problem in which at least one of the two following conditions holds:

- the cost function f is non-linear, or
- the constraint set X is specified by non-linear equations and inequalities

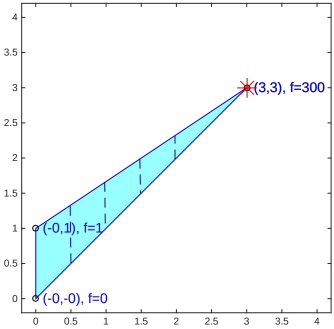
These problems (NLPs) are generally much harder to solve than LPs. The optimal solution may not occur at an extreme point of the feasible region.

This is the type of problem which we will focus on in this course.

Class Exercise 1.1

Solve the following LP, by first sketching the feasible region:

$$\begin{aligned} \text{max} \quad & f(x) = 99x_1 + x_2 \\ \text{such that} \quad & -\tfrac{2}{3}x_1 + x_2 \le 1 \\ & x_1 - x_2 \le 0 \\ & x_1 \ge 0 \\ & x_2 \ge 0 \end{aligned}$$



(3,3), f=300
(-0,1), f=1
(-0,-0), f=0

Non-linear Programming Application

Solve the following NLP:

$$
\begin{aligned}
\text{min} \quad & f(x) = (x_1 - 1)^2 - x_2 \\
\text{such that} \quad & x_2 - x_1 \leq 0 \\
& x_2 \leq 2 \\
& x_1 \geq 0 \\
& x_2 \geq 0
\end{aligned}
$$

NLP example (slide 27)

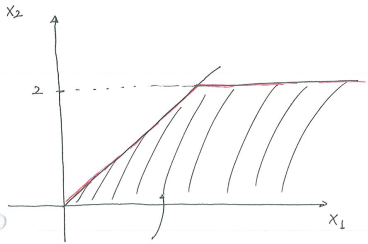
$$\text{Min } f(x) = (x_1 - 1)^2 - x_2$$

$$x_2 - x_1 \leq 0$$

$$x_1 \leq 2$$

$$x_1 \geq 0$$

$$x_2 \geq 0$$



for any given x_1, we can decrease the function by increasin x_2 $\Rightarrow$ the optimum lies in the border

$\boxed{\text{if } x_1 \leq 2} \Rightarrow x_1 = x_2$

$f(x_1) = (x_1 - 1)^2 - x_1$ stationary condition $f'(x_1) = 0$

$f'(x_1) = 2(x_1 - 1) - 1 = 0$ $\quad f(x_1^*) = 1/4 - 3/2 = -5/4$

$\Rightarrow x_1 = 3/2 \Rightarrow x_2 = 3/2 \quad \rightarrow f''(x_1) = 2 > 0$ minimum

$\boxed{\text{if} \quad x_1 \geq 2}$

$$x_2 = 2$$

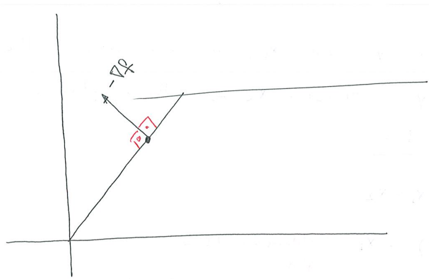
$$f(x_1) = (x_1 - 1)^2 - \cancel{x_2}^2$$

$$f'(x_1) = 2(x_1 - 1)$$

$$= 0 \Rightarrow x_1 = \frac{1}{2} \quad \text{(contradiction)}.$$

Note that

$$\nabla f(x^*) = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$



Section 2

Unconstrained Single Variable Optimisation

- Golden section search
- Fibonnacci search
- Method of false position
- Newton's method
- Comparison of 1D methods
- Extending 1D methods for half-open intervals
- Armijo-Goldstein and Wolff conditions

Unconstrained Optimisation - Single Variable Case

A Problem to Think About

- Given a continuous function $f : \mathbb{R} \to \mathbb{R}$, defined on an interval $[a, b]$, find the point $x^* \in [a, b]$ such that $f(x^*) \leq f(x)$ for all $x \in [a, b]$.

A point x^* satisfying the above condition is called a *global minimum* of f on $[a, b]$.

Global and local minima - Definitions

A *global minimum* x^* of f on $[a, b]$ is a point for which $f(x^*) \leq f(x)$ for all $x \in [a, b]$.

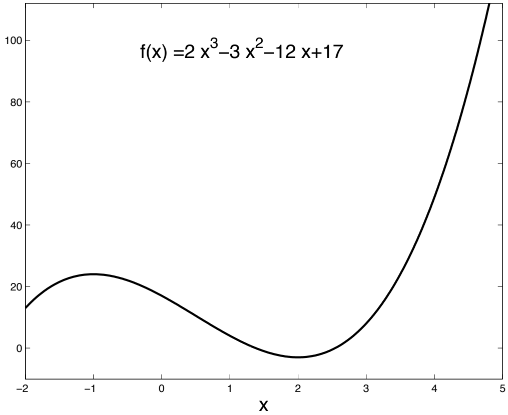
A *local minimum* x^* of f on $[a, b]$ is a point such that there exists a value $\epsilon > 0$ with $f(x^*) \leq f(x)$ for all $x \in [a, b]$ such that $|x - x^*| < \epsilon$.

By the above definitions, a global minimum is also a local minimum.

A local minimum may or may not be a global minimum.

Example:

$$\min_{x \geq -2} f(x) = 2x^3 - 3x^2 - 12x + 17$$



The Analytical Approach

If we have a straightforward formula for $f(x)$ then we can solve the problem using elementary methods of calculus:

1. set the first derivative of f equal to zero to determine the stationary points of f.
2. work out which of these are minima.
3. also check the value of f at the endpoints.

A More General Approach

However, we do not want to make the assumptions that we have a formula for f or, even if we have, that we can necessarily calculate a formula for the derivative. Rather, we want to consider f as a *black box* into which we feed in a value and get another value.

It might actually take a reasonable length of time for the black box to complete the task, so we really want a method that uses the f black box as little as possible.

How do we “find” a global minimum, just by sampling points on the domain of f?

Definition - Unimodal

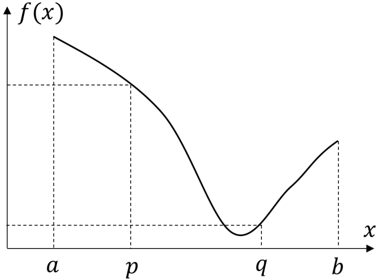
The continuous function f is *unimodal* on $[a, b]$ if it has only one local minimum.
Note: this local minimum for a unimodal function also a global minimum.

Until further notice, we shall assume that f is continuous and unimodal.

Assume that we have two f-calculations at points p and q. That is, we know $f(p)$ and $f(q)$.

A little thought shows us that

- If $f(p) \leq f(q)$, then $x_{min} \in [a, q]$.
- If $f(p) \geq f(q)$, then $x_{min} \in [p, b]$.



Estimating x_{min} for a Unimodal Function

Suppose that we want to find x_{min} to within a specified tolerance ϵ. That is we want to reduce the size of the interval in which we know the minimum lies to width 2ϵ.

How do we choose p, q and all the subsequent points at which we make an f-calculation so that we get to this stage with as few f-calculations as possible?

Subsection 1

Golden section search

The Golden Section Search

Suppose we make n f-calculations to reduce the length of the interval in which the minimum occurs.

Points p and q with $p < q$ are chosen in the following way:

$$p = a + \tau^L(b - a) \text{ and } q = b - \tau^R(b - a),$$

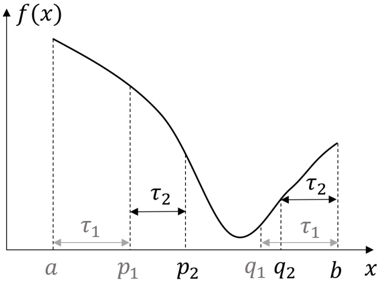
where $\tau^L, \tau^R < 0.5$.

After the first iteration, the interval will be reduced to either $[a, q]$ or $[p, b]$. We can't tell which will be the case (until after we have made the calculations). So it makes sense to choose $q - a = b - p$, which means $\tau^L = \tau^R = \tau < 0.5$.

The Golden Section Search

Now we assume (w.l.o.g.) $f(p_1) > f(q_1)$, and then the new interval is $[p_1, b]$. In the second iteration, we choose p_2 and q_2 in a similar way with the new interval. Note that

$$p_1 = a + \tau_1(b - a), \qquad q_1 = b - \tau_1(b - a),$$
$$p_2 = p_1 + \tau_2(b - p_1), \qquad q_2 = b - \tau_2(b - p_1).$$



The Golden Section Search

We want one f-calculation in this iteration, and so need $p_2 = q_1$.

$$\Rightarrow \tau_2(1-\tau_1) = 1 - 2\tau_1. \tag{2.1}$$

Assume that ratio τ_k is independent of iteration k. $\Rightarrow \tau_1 = \tau_2 = \tau$. Then rearranging Eq. (2.1) gives

$$\tau^2 - 3\tau + 1 = 0. \tag{2.2}$$

The solutions are

$$\tau = \frac{3-\sqrt{5}}{2} \text{ and } \tau = \frac{3+\sqrt{5}}{2} \text{ (disregarded).}$$

The Golden Section Search

Notice that

$$1 - \tau = \frac{\sqrt{5} - 1}{2} = \gamma.$$

The number $\gamma \approx 0.618$ is the *golden ratio*, a ratio that is supposed to possess many aesthetic properties.

Golden Section Search Algorithm

To minimise a unimodal function f over $[a, b]$ to within tolerance ϵ.

1. Set

$$k = 1 \tag{2.3}$$
$$p = b - \gamma(b - a) \tag{2.4}$$
$$q = a + \gamma(b - a) \tag{2.5}$$

   Calculate $f(p)$ and $f(q)$.

2 Set $k = k + 1$. If $f(p) \leq f(q)$, then set

$$b = q \tag{2.6}$$
$$q = p \tag{2.7}$$
$$p = b - \gamma(b - a) \tag{2.8}$$

Calculate $f(p)$. If $f(p) > f(q)$, then set

$$a = p \tag{2.9}$$
$$p = q \tag{2.10}$$
$$q = a + \gamma(b - a) \tag{2.11}$$

Calculate $f(q)$. Repeat until $(b - a) < 2\epsilon$.

We will go step-by-step through a worked example that illustrates the Golden section search.

Example 2.1

$$\text{Minimise } f(x) = x^3 - 5x^2 - 3x + 7$$

$$\text{on the interval } x \in [2, 6]$$

Solving analytically:

$$f'(x) = 3x^2 - 10x - 3$$

When this is 0,

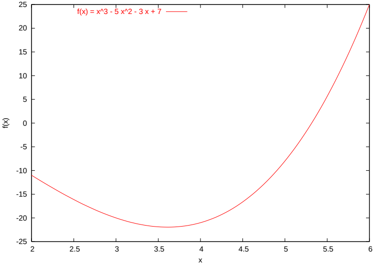
$$x = \frac{5}{3} \pm \frac{\sqrt{34}}{3} = 3.610, -0.27$$

We want the first one.

$$f''(x) = 6x - 10$$

$$f''(3.610) = 11.66 > 0$$

so it is a minimum.



We want to find the minimum to a tolerance of 0.1 for Golden section search.

Golden section search

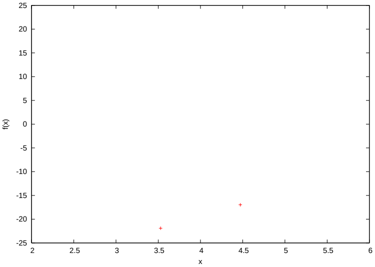
$$\gamma^7(6-2) = 0.13 < 0.2$$

so we need 8 f-calculations.

$$a = 2, b = 6$$

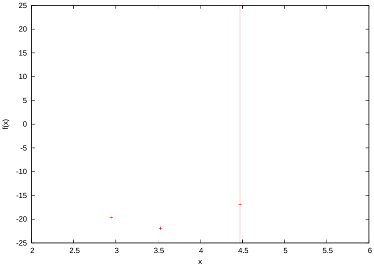
$$p = 6 - \gamma(6-2) = 3.53$$

$$q = 2 + \gamma(6-2) = 4.47$$



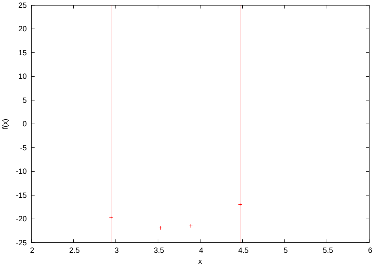
$$f(p) = -21.9055 < f(q) = -16.9737$$

$$p = 4.47214 - \gamma(4.47214 - 2) = 2.94427, q = 3.52786$$



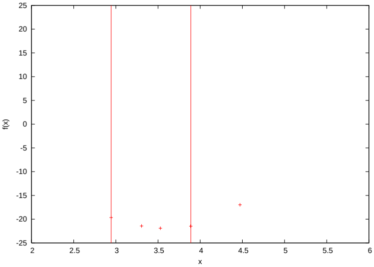
$$f(p) = -19.6534 > f(q) = -21.9055$$

$$p = 3.52786, q = 2.94427 + \gamma(4.47214 - 2.94427) = 3.88854$$



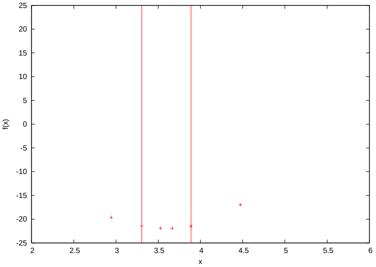
$$f(p) = -21.9055 < f(q) = -21.4717$$

$$p = 3.88854 - \gamma(3.88854 - 2.94427) = 3.30495, q = 3.52786$$



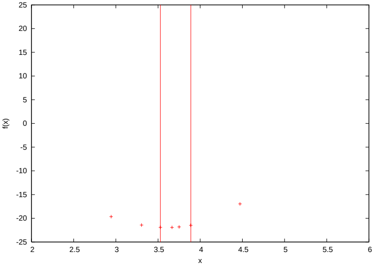
$$f(p) = -21.4294 > f(q) = -21.9055$$

$$p = 3.52786, q = 3.30495 + \gamma(3.88854 - 3.30495) = 3.66563$$



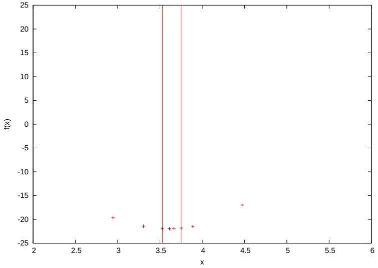
$$f(p) = -21.9055 > f(q) = -21.9266$$

$$p = 3.66563, q = 3.52786 + \gamma(3.88854 - 3.52786) = 3.75078$$

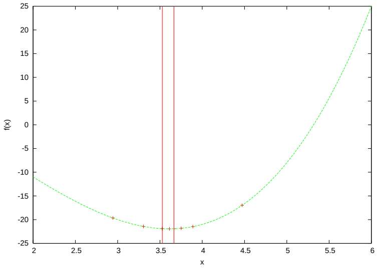


$$f(p) = -21.9266 < f(q) = -21.8268$$

$$p = 3.75078 - \gamma(3.75078 - 3.52786) = 3.61301, q = 3.66563$$



$$f(p) = -21.9446 < f(q) = -21.9266$$



Final interval is $[3.52786, 3.66563]$. Final estimate is 3.597 ± 0.069.

Subsection 2

Fibonnacci search

The Fibonacci Search

Now we are allowed to vary the ratio τ from iteration to iteration.

How can we use $n \geq 2$ f-calculations, to reduce the length of the interval in which the minimum occurs from $[a, b]$ to the smallest length possible?

Notation

α	length of the smallest interval containing x_{min} we can construct
$F_k(\alpha)$	the maximum length of an interval that can be reduced to α in k f-calculations.

Note: by definition $F_n(\alpha) = b - a$.

The Fibonacci Search

Make two f calculations at points $p < q$. Observe that the new interval will be either $[a, q]$ or $[p, b]$. Again we should choose p and q such that $q - a = b - p$.

For the purposes of this explanation, assume (w.l.o.g.) that $f(p) \geq f(q)$ and so $x_{min} \in [p, b]$.

The length of $[p, b]$ (and $[a, q]$) can be denoted $F_{n-1}(\alpha)$.

We still have $n - 2$ f-calculations to use to reduce the interval. Consider first the case $n > 2$.

The Fibonacci Search ($n > 2$)

In order to reduce the interval again, we need to evaluate f at a new point r, then compare $f(r)$ and $f(q)$. By the same argument as we used above, we should choose r such that $b - q = r - p$.

Both of the intervals $[p, r]$ and $[q, b]$ are of length $F_{n-2}(\alpha)$.

We can now write the relation

$$b - a = (b - q) + (q - a)$$

as

$$F_n(\alpha) = F_{n-2}(\alpha) + F_{n-1}(\alpha). \tag{2.12}$$

Class Exercise 2.2

Use the Fibonacci Search method to find the smallest possible interval containing the minimum of the unimodal function f, where $f(x) = (x-1)^2 - x$, over the interval $[0, 2]$ using exactly 3 computations of f. What is the final interval for x?

Class exercise 2 (37)

$(x-1)^2 - x$

(*) $[\ 0 \quad p \quad q \quad 2\ ]$

$$\boxed{q - 0 = 2 - p \quad , \quad p < q}$$

choose $p = 0.5$

$q = 1.5$

$f(p) = f(0.5) = -0.25$

$f(q) = f(1.5) = -1.25$

new interval $= [p, 2] = [0.5, 2]$

$[\quad 0.5 \quad r \quad 1.5 \quad 2\ ]$

$1.5 - 0.5 = 2 - r \Rightarrow \underline{r = 1}$

$f(r) = -1$

→ Solution between $[1, 2]$

≠ values of p yield ≠ intervals ⇒ algorithm

↳ Could be better

Following the algorithm:

(*) $k = 3$

$p = b - \frac{F_2}{F_3}(b-a) = 2 - \frac{2}{3}2 = \underline{2/3}$

$q = a + \frac{F_2}{F_3}(b-a) = 0 + \frac{2}{3} \cdot 2 = 4/3$

$f(p) = \frac{1}{9} - \frac{2}{3} = \frac{1-6}{9} = -5/9$

$f(q) = \frac{1}{9} - \frac{4}{3} = \frac{1-12}{9} = -11/9$

~~* $f(q) < f(p) \Rightarrow$ new interval $[p\ b] \neq [2/3, 2]$~~

Step 4 $\quad a = p, \quad p = q, \quad q = a + 2\varepsilon$

$[2/3, 2]$

$[\ 0 \quad 2/3 \quad 4/3 \quad 2\ ]$

new p

~new q

The Fibonacci Search ($n \leq 2$)

If $n = 2$, then the situation is a bit different; we have no more f calculations to use (except for the "basic" first two, without which we can't do anything). Thus, we want to make $q - a$ and $p - b$ as small as possible, which we can do by making them both equal to $(b - a)/2$. Therefore we take $q = p = a + (b - a)/2$.

In practice, we can't do this exactly because we need $p < q$, but we can take p and q arbitrarily close to this point.

This gives us

$$F_2(\alpha) = 2\alpha \quad \text{(approximately)}. \tag{2.13}$$

The Fibonacci Search ($n \leq 2$)

Also, we can't reduce the length of the interval at all with zero or one f-calculation, so it follows that

$$F_0(\alpha) = \alpha \tag{2.14}$$

and

$$F_1(\alpha) = \alpha. \tag{2.15}$$

Equations (2.12) define the *Fibonacci sequence*.

The initial conditions are given by (2.14) and (2.15). Thus we see that $F_n(\alpha) = F_n\alpha$ where F_n is given by the Fibonacci sequence: $F_0 = 1$, $F_1 = 1$, and $F_n = F_{n-1} + F_{n-2},\ n = 2, 3, \ldots$.

Fibonacci Search Algorithm

To minimise a continuous unimodal function f over $[a, b]$ to within tolerance ϵ.

1. Find the smallest value of n such that $(b-a)/F_n < 2\epsilon$.
2. Set

$$k = n \tag{2.16}$$

$$p = b - \frac{F_{k-1}}{F_k}(b-a) \tag{2.17}$$

$$q = a + \frac{F_{k-1}}{F_k}(b-a) \tag{2.18}$$

   Calculate $f(p)$ and $f(q)$.

3. Set $k = k - 1$. If $f(p) \leq f(q)$, then set

$$b = q \tag{2.19}$$

$$q = p \tag{2.20}$$

$$p = b - \frac{F_{k-1}}{F_k}(b - a) \tag{2.21}$$

Calculate $f(p)$. If $f(p) > f(q)$, then set

$$a = p \tag{2.22}$$

$$p = q \tag{2.23}$$

$$q = a + \frac{F_{k-1}}{F_k}(b - a) \tag{2.24}$$

Calculate $f(q)$. Repeat until $k = 3$.

4 If $f(p) \leq f(q)$, then set

$$b = q \tag{2.25}$$

$$q = p \tag{2.26}$$

$$p = b - 2\epsilon \tag{2.27}$$

Calculate $f(p)$.

If $f(p) > f(q)$, then set

$$a = p \tag{2.28}$$

$$p = q \tag{2.29}$$

$$q = a + 2\epsilon \tag{2.30}$$

Calculate $f(q)$.

5. If $f(p) \leq f(q)$, then $b = q$.

   If $f(p) > f(q)$, then $a = p$.

   The final interval is $[a, b]$. This interval has length either 2ϵ or $\alpha < 2\epsilon$.

Now we will use the Fibonacci Search Algorithm to solve Example 2.1, that is,

$$\text{Minimise } f(x) = x^3 - 5x^2 - 3x + 7$$

$$\text{on the interval } x \in [2, 6]$$

We want to find, numerically, the minimum to a tolerance of 0.1.

Fibonacci search

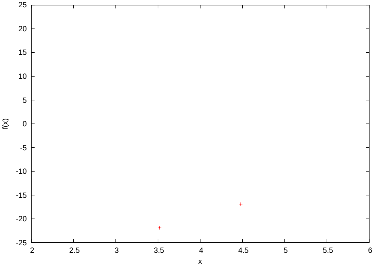
$$\frac{6-2}{F_7} = \frac{6-2}{21} = 0.19 < 0.2$$

so we need 7 calculations.

$$a = 2, b = 6$$

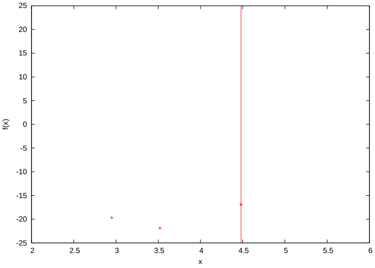
$$p = 6 - \frac{13}{21}(6-2) = 3.52$$

$$q = 2 + \frac{13}{21}(6-2) = 4.48$$



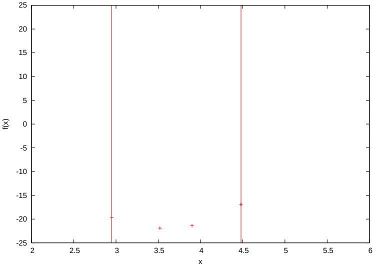
$$f(p) = -21.9 < f(q) = -16.9$$

$$p = 4.48 - \frac{8}{13}(4.48 - 2) = 2.95, q = 3.52$$



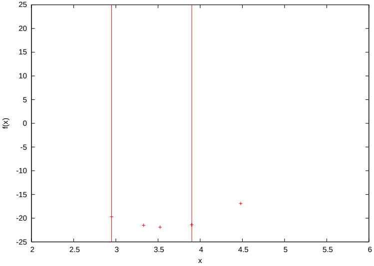
$$f(p) = -19.7 > f(q) = -21.9$$

$$p = 3.52, q = 2.95 + \frac{5}{8}(4.48 - 2.95) = 3.90$$



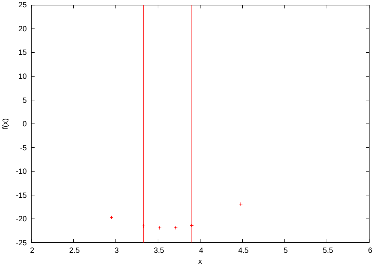
$$f(p) = -21.9 < f(q) = -21.4$$

$$p = 3.90 - \frac{3}{5}(3.90 - 2.95) = 3.33, q = 3.52$$



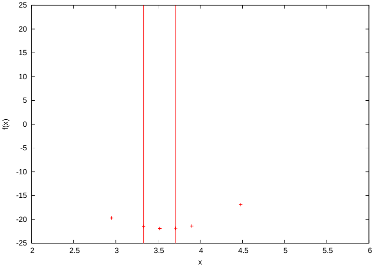
$$f(p) = -21.5 > f(q) = -21.9$$

$$p = 3.52, q = 3.33 + \frac{2}{3}(3.90 - 3.33) = 3.71$$

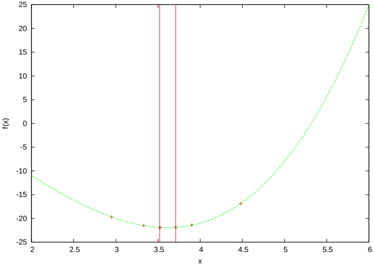


$$f(p) = -21.9 < f(q) = -21.88$$

$$p = 3.51, q = 3.5238$$



$$f(p) = -21.89779 > f(q) = -21.90163$$



Final interval is $[3.52, 3.71]$. Final estimate is 3.614 ± 0.1.

Revisit The Golden Section Search

The Fibonacci search method

- chooses ratios depending on k;
- requires pre-computation of the Fibonacci numbers;
- reduces the size of an interval in which the minimum occurs in the most efficient way possible.

The Golden section search method

- chooses ratios independent of k;
- approximates the Fibonacci search via removing the dependency of the ratios on k. Specifically,

$$\gamma = \lim_{k \to \infty} F_{k-1}/F_k.$$

Methods Using the Derivative

As before, assume f is continuous and unimodal on $[a, b]$.

Also, assume f is differentiable and that we have a "black box" for f' (and possibly f'') as well as one for f.

The problem of minimising f now reduces to the problem of finding a point x^*, where $g(x^*) = 0$, where $g = f'$.

Subsection 3

Method of false position

The Method of False Position

Let g be a continuous and increasing function on $\mathbb{R}$. The method of false position is a recursive method for finding the point x^* where $g(x^*) = 0$.

Assume that we can find two points a and b such that $g(a) < 0$ and $g(b) > 0$. The line through the points $(a, g(a))$ and $(b, g(b))$ has equation

$$y = \frac{g(b) - g(a)}{b - a}(x - a) + g(a). \tag{2.31}$$

The value of the right hand side of (2.31) is equal to zero at the point

$$x^*_{estimate} = a + \frac{(b - a)g(a)}{g(a) - g(b)}. \tag{2.32}$$

The Method of False Position

$x^*_{estimate}$ is a linear estimate for the point x^* at which $g(x^*) = 0$.

- If $g(x^*_{estimate}) = 0$ (to within some tolerance), then we have found a “near enough” approximation to the point x^*.
- If $g(x^*_{estimate}) < 0$, then we use $x^*_{estimate}$ as the lower bound for a new interval.
- If $g(x^*_{estimate}) > 0$, then we use $x^*_{estimate}$ as the upper bound for a new interval.

We can extend this to develop a recursive algorithm as follows.

Algorithm for the Method of False Position

For an increasing, continuous function g on $[a, b]$, to find a point x^* where $|g(x^*)| < \epsilon$.

1. Set

$$k = 1 \tag{2.33}$$

$$p = a + \frac{(b-a)g(a)}{g(a)-g(b)} \tag{2.34}$$

   Calculate $g(p)$.

2. Set $k = k + 1$.
If $g(p) < 0$, then set

$$a = p \tag{2.35}$$
$$p = a + \frac{(b-a)g(a)}{g(a) - g(b)} \tag{2.36}$$
$$\tag{2.37}$$

Otherwise, if $g(p) > 0$, then set

$$b = p \tag{2.38}$$
$$p = a + \frac{(b-a)g(a)}{g(a) - g(b)} \tag{2.39}$$

Calculate $g(p)$.

Repeat until $|g(p)| < \epsilon$.

If f is convex, then f' is increasing and so we can use the algorithm for the method of false position with $g = f'$ to find the minimum of f.

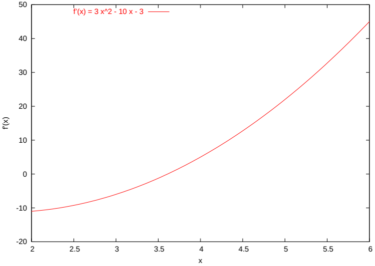
The method will work for any unimodal function f unless it has saddle points.

Now we will use the Method of False Position to solve Example 2.1, that is,

$$\text{Minimise } f(x) = x^3 - 5x^2 - 3x + 7$$

$$\text{on the interval } x \in [2, 6]$$

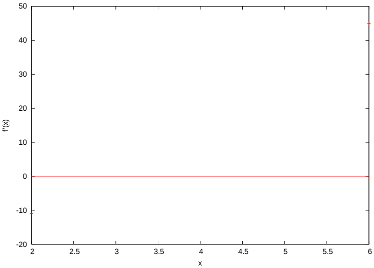
We want to find, numerically, the minimum with an absolute tolerance derivate of less than 0.1.



For the derivative methods we want to find the point where $f' = 0$.

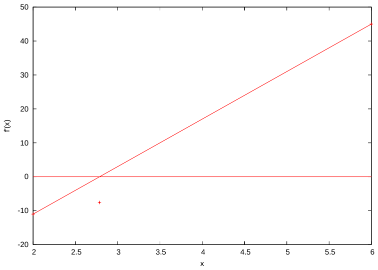
$$f'(x) = 3x^2 - 10x - 3$$

False position method



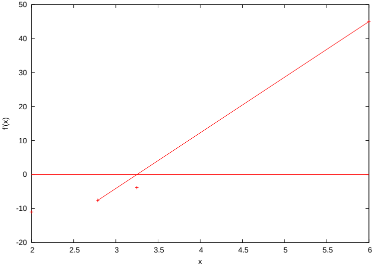
$$a = 2, b = 6$$

$$f'(a) = -11, \quad f'(b) = 45$$



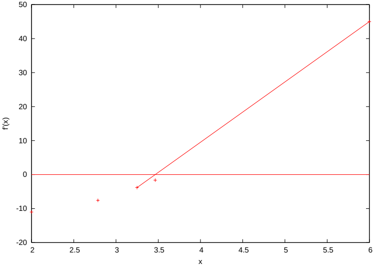
$$p = 2.78571$$

$$f'(p) = -7.57653$$



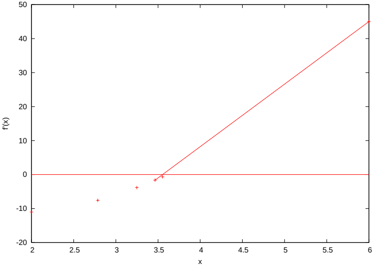
$$a = 2.78571, p = 3.24891$$

$$f'(p) = -3.82287$$



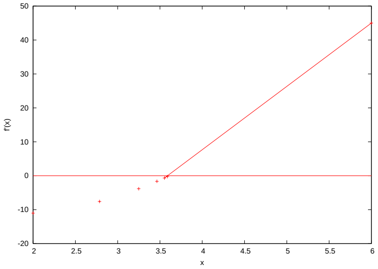
$$a = 3.24891, p = 3.46432$$

$$f'(p) = -1.63865$$



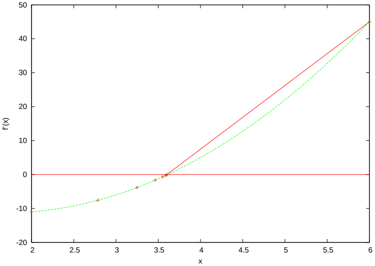
$$a = 3.46432, p = 3.55341$$

$$f'(p) = -0.653909$$



$$a = 3.55341, p = 3.58845$$

$$f'(p) = -0.253522$$



$$a = 3.58845, p = 3.60197$$

$$f'(p) = -0.0971933$$

Final interval is $[3.60197, 6]$. Final estimate is 3.602. We needed 8 calculations.

Subsection 4

Newton's method

Newton's Method (Single Variable case)

Another method that we can use to find the point at which an increasing function g is equal to zero is Newton's Method.

The method is both simple to implement and typically converges very quickly.

For this method, we need black boxes for both the function g and its derivative g'.

Newton's Method (Single Variable case)

The idea is to calculate an estimate of the point x^* at which $g(x^*) = 0$ by calculating the point at which the tangent to g at some point a would cross the x-axis.

The tangent to g at the point $(a, g(a))$ has slope $g'(a)$, and so its equation is

$$y = g'(a)(x - a) + g(a). \tag{2.40}$$

Thus $y = 0$ when $x = a - g(a)/g'(a)$.

Newton's Method (Single Variable case)

We incorporate this into a recursive algorithm by calculating the value of $g(x)$ and $g'(x)$ at a point a then compute a new estimate for x^* by finding the point at which $y = 0$ on the tangent line.

We need to beware not to "divide by zero". Thus we test whether $g'(a)$ is very small. In fact if there is a zero of g' close to the zero of g, then Newton's Method doesn't work.

Can you explain this physically?

Algorithm for Newton's Method

For an increasing, continuous function g on $\mathbb{R}$ and an initial starting point a, to find a point x^* where $|g(x^*)| < \epsilon$.

1. Set

$$k = 1 \tag{2.41}$$

$$\text{if} \quad g'(a) < \epsilon \quad \text{then signal and stop.} \tag{2.42}$$

$$\text{else} \quad p = a - \frac{g(a)}{g'(a)} \tag{2.43}$$

2 Set $k = k + 1$.

$$a = p \tag{2.44}$$

$$\text{if} \quad g'(a) < \epsilon \quad \text{then signal and stop.} \tag{2.45}$$

$$\text{else} \quad p = a - \frac{g(a)}{g'(a)} \tag{2.46}$$

Repeat until $|g(p)| < \epsilon$.

As with the method of false position, we can find the minimum of a convex and twice differentiable function f by using Newton's Method with $g = f'$ and $g' = f''$.

Now we will use the Newton's Method to solve Example 2.1, that is,

$$\text{Minimise } f(x) = x^3 - 5x^2 - 3x + 7$$

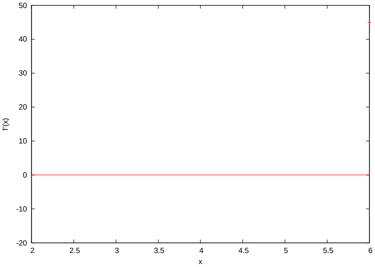
$$\text{on the interval } x \in [2, 6]$$

We want to find, numerically, the minimum with an absolute tolerance derivate of less than 0.1.

$$f'(x) = 3x^2 - 10x - 3$$

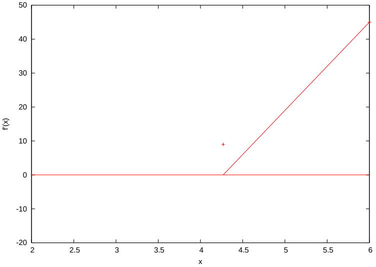
$$f''(x) = 6x - 10$$

Newton's method



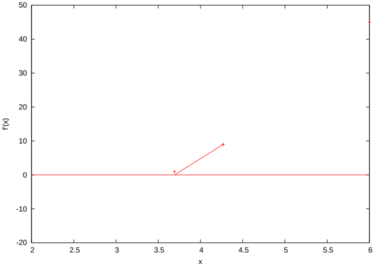
$$a = 6$$

$$f'(a) = 45, f''(a) = 26$$



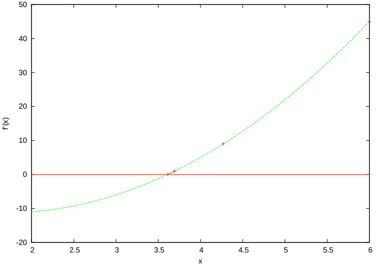
$$p = 4.26923$$

$$f'(p) = 8.98669, f''(p) = 15.6154$$



$$a = p, p = 3.69373$$

$$f'(p) = 0.993608, f''(p) = 12.1624$$



$$a = p, p = 3.61203$$

$$f'(p) = 0.0200223$$

Final estimate is 3.612. We needed 4 calculations, plus 3 f'' calculations.

Subsection 5

Comparison of 1D methods

Comparison of 1D methods for Example 2.1

Method	*Solution*	*Calculations*
Exact	3.610	1 long one
Golden	3.597	8
Fibonacci	3.614	7
False position	3.602	8
Newton's	3.612	7

This shows that:

- Fibonacci takes a long time to do! But it is efficient.
- Golden section is less efficient than Fibonacci, but not by a huge amount.
- Newton's method converges very quickly, but requires the most information.

Subsection 6

Extending 1D methods for half-open intervals

Finding the Minimum on a Half-Open Interval

In optimisation algorithms for minimising functions on $\mathbb{R}^n$, we frequently find ourselves in the situation where we want to find the minimum of a continuous and unimodal function on the interval $[0, \infty)$.

For example, many of these algorithms work by choosing a direction and then minimising the function along the chosen direction. If we consider the direction as parametrised by the single variable t, then this requires us to minimise a function of t in the interval $[0, \infty)$.

Finding the Minimum on a Half-Open Interval

Apart from Newton's Method, where we needed to have black boxes for both f' and f'', this situation is not quite the same as the ones that we have been talking about, where we started out with the assumption that the minimum occurred somewhere in a closed interval $[a, b]$.

To use our previous techniques, we need to find a number b, such that the minimum of the function f lies in the interval $[0, b]$.

Finding the Minimum on a Half-Open Interval

One way to do this is:

1. Choose a point $T \in \mathbb{R}^+$;
2. Increment this point by increasing multiples of T, until we find a point of the form kT, where k is an integer, such that $f(kT) \geq f((k-1)T)$.

Because f is assumed to be continuous and unimodal, we then know that the minimum lies in the interval $[0, kT]$ and we are in a position to use one of the algorithms that we discussed previously.

Procedure for Finding an Upper Bound on the Location of the Minimum

For a continuous, unimodal function f on $[0, \infty)$, to find a point b such that the minimum $x_{min} < b$.

1. Choose some increment value T.

$$k = 1 \tag{2.47}$$
$$p = 0 \tag{2.48}$$
$$q = T \tag{2.49}$$

Calculate $f(p)$ and $f(q)$. If $f(p) \leq f(q)$, then stop. Else...

2. Set $k = k + 1$.

$$p = q \tag{2.50}$$
$$q = kT \tag{2.51}$$

Calculate $f(q)$.

Repeat until $f(p) \leq f(q)$.

The problem with this procedure lies in choosing a suitable increment size T. Since we know nothing about the function f, if we choose T to be too small, we will be wasting a lot of time and resources calculating function values unnecessarily.

On the other hand, if we take T to be too big, we may end up with an interval $[0, b]$ which is much bigger than necessary and so we will have to make many more f-calculations than necessary to reduce the size of this interval.

To address this problem we can use a variable increment size. Usually we start out with a small increment size and increase it exponentially, for example by doubling it. We thus get the following algorithm:

An Improved Procedure for Finding an Upper Bound on the Location of the Minimum

For a continuous, unimodal function f on $[0, \infty)$, to find a point b such that the minimum $x_{min} < b$.

1. Choose some small initial increment value T.

$$k = 1 \tag{2.52}$$

$$p = 0 \tag{2.53}$$

$$q = T \tag{2.54}$$

Calculate $f(p)$ and $f(q)$. If $f(p) \leq f(q)$, then stop. Else...

2. Set $k = k + 1$.

$$p = q \tag{2.55}$$

$$q = p + 2^{k-1} T \tag{2.56}$$

Calculate $f(q)$.

Repeat until $f(p) \leq f(q)$.

An Upper Bound on the Location of the Minimum

The improved algorithm has the flexibility to "discover" the appropriate scale for the function f.

It actually does better than just find an upper bound for the minimum of f. It will stop at a point b which is of the form

$$b = T \sum_{k=0}^{n} 2^k$$

Since $f(p) > f(q)$ at the previous iteration, we know that x_{min} lies in the interval $[a, b]$ where

$$a = T \sum_{k=0}^{n-2} 2^k.$$

We are then in a position to apply any of the methods that we discussed earlier (eg: Fibonacci, Golden Section) to reduce the width of the interval in which we know the minimum lies.

Bounding the Real Number Line $\mathbb{R}$

A modification of the above method will also work if we know only that x_{min} lies in $\mathbb{R}$.

Evaluate f at two points a and $a + T$. If $f(a) > f(a + T)$, then we know $x_{min} > a$ and we can apply the above algorithm starting at a instead of at 0.

If $f(a) \leq f(a + T)$, then we know $x_{min} < a + T$ and we apply the above algorithm in the negative direction starting at $a + T$.

Subsection 7

Armijo-Goldstein and Wolff conditions

The Armijo-Goldstein and Wolff conditions

Single-variable optimisation procedures are often used as components of algorithms for minimising functions on $\mathbb{R}^n$. These *descent methods* work by choosing a descent direction and moving along that descent direction to find a point where the function value is lower than it was before.

The minimum value of the function restricted along the descent direction is not (in general) the minimum of the function over $\mathbb{R}^n$.

The Armijo-Goldstein and Wolff conditions

For this reason, it often does not make a lot of sense to spend a lot of time finding the minimum along any given descent direction. Rather, it can be enough to find a point that is a significant improvement on the one we had before.

One way of expressing what is meant by "significant improvement" is given by the Armijo-Goldstein and Wolff conditions.

The Armijo-Goldstein condition

Let $\sigma \in [0, 1]$. For a unimodal, continuous and differentiable function f on $[0, \infty)$, we say that the step size t satisfies the *Armijo-Goldstein condition* with weight σ if

$$f(t) \leq f(0) + t\sigma f'(0). \tag{2.57}$$

Note that $f'(0)$ will be negative (unless the minimum occurs at $t = 0$). Hence, the Armijo-Goldstein condition ensures that the step size t cannot be too large.

Class Exercise

We want to find the minimum of the function $f(x) = (x-3)^2$ on the interval $[0, \infty)$. Suppose we start at $x = 0$ and at the kth iteration of our search choose a step size of $t_k = 1/2^k$ in the positive direction.

1. Show that after N steps (for any positive integer N) we will still be a distance > 1 from the minimum of the function.

2. Show that despite this failure, the step size scheme $t_k = 1/2^k$ satisfies the Armijo-Goldstein condition for $f(x)$ at each step, with $\sigma = 1/2$.

Class exercise - slide 115

$$f(x) = (x-3)^2$$

(Analytically)

$$f'(x) = 2(x-3) \implies f'(x) = 0 \rightarrow x^* = 3$$

(it's a minimum since $f''(x^*) = 2 > 0$)

(with the iterative approach)

Step size: $t_k = (1/2)^k$

$x_0 = 0$; $t_0 = (1/2)^0 = 1$

$x_1 = 1$; $t_1 = 1/2$

$x_2 = 3/2$; $t_2 = 1/4$

$\vdots$

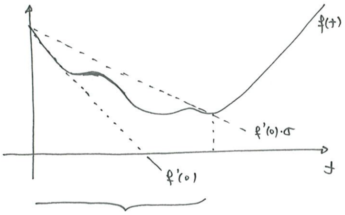
$$\boxed{x_k = 2 - (1/2)^{k-1} \quad , \quad k \geq 1 .}$$

$f(x_k) = \left(2 - (1/2)^{k-1} - 3\right)^2 = \left(-1 - (1/2)^{k-1}\right)^2 = (1 + (1/2$

$f(x_k + t_k) = f(x_{k+1}) = \left(2 - (1/2)^k - 3\right)^2 = \left(1 + (1/2)^k\right)^2$

$f'(x_k) = 2(x_k - 3) = 2\left(2 - (1/2)^{k-1} - 3\right) = -2(1 + (1/2$

$$f(t) \leq f(0) + t \cdot \sigma \cdot f'(0)$$



acceptable t (not too large).

In our case:

$$f(x_k + t_k) \leq f(x_k) + t_k \cdot \sigma \cdot f'(x_k)$$

$$\left(1 + (1/2)^k\right)^2 \leq \left(1 + (1/2)^{k-1}\right)^2 + (1/2)^k \cdot 1/2 \cdot \left(-2(1\right.$$

$$1 + (1/2)^{k-1} + (1/2)^{2k} \leq 1 + (1/2)^{k-2} + (1/2)^{2k-2} - (1/2)^k \cdot$$

$$1 + 2(1/2)^k + (1/2)^{2k} \leq 1 + 4(1/2)^k + 4 \cdot (1/2)^{2k} - (1/2)^k - 2$$

$$1 + 2(1/2)^k + (1/2)^{2k} \leq 1 + 3(1/2)^k + 2 \cdot (1/2)^{2k}$$

QED

t_k always respect the condition

The Wolff condition

The problem with the previous example is the opposite to the one that the Armijo-Goldstein condition addresses. The step size has become too small.

To get around this we introduce a new condition - the Wolff condition.

For a unimodal, continuous and differentiable function f on $[0, \infty)$, we say that the step size t satisfies the Wolff condition with weight μ if

$$f'(t) \geq \mu f'(0) \tag{2.58}$$

where $\mu \in [\sigma, 1)$.

The Wolff condition

To understand what the Wolff condition means, observe that since $f'(0) < 0$ and $\mu \in (0,1)$, then

$$f'(0) < \mu f'(0) < 0$$

Also

$$f'(t) \to f'(0)$$

as $t \to 0$ so, by making sure that $f'(t) \geq \mu f'(0)$, the Wolff condition forces t not be be too close to 0.

That is, the step size has to be "sufficiently big".

A Procedure to Find a Step Size that Satisfies The Armijo-Goldstein and Wolff Conditions

The following procedure initially reduces a large step size so that it becomes small enough to satisfy the Armijo-Goldstein condition and then increases a small step size so that it becomes large enough to satisfy the Wolff condition.

For a differentiable, unimodal function f on $[0, \infty)$. Input an initial step size T, a number $\sigma \in (0, 1)$ and a number $\mu \in [\sigma, 1)$.

1. Set

$$t_{lo} = 0 \tag{2.59}$$

$$t_{hi} = \infty \tag{2.60}$$

$$t = T \tag{2.61}$$

2 If $f(t) > f(0) + t\sigma f'(0)$, then

$$t_{hi} = t \tag{2.62}$$
$$t = 1/2(t_{lo} + t_{hi}) \tag{2.63}$$

Else if $f'(t) < \mu f'(0)$, then

$$t_{lo} = t \tag{2.64}$$
$$t = \begin{cases} 1/2(t_{lo} + t_{hi}) & \text{if} \quad t_{hi} < \infty \\ 2t & \text{otherwise} \end{cases} \tag{2.65}$$

Repeat until $f(t) \leq f(0) + t\sigma f'(0)$ and $f'(t) \geq \mu f'(0)$.

Effectiveness of the Step Size Procedure

Proposition 1:

Let $0 < \sigma \leq \mu < 1$. The above line search procedure either finds a point such that $f(t) \leq f(0) + t\sigma f'(0)$ and $f'(t) \geq \mu f'(0)$ in finitely many steps or it produces a sequence of ts, say $\{t_k\}$ such that $f(t_k) \to -\infty$ as $k \to \infty$.

This result says that either the line search procedure works as intended in a finite number of steps, or f is unbounded below. In the latter case, minimising f doesn't make sense.

Section 3

Unconstrained Optimisation with n Variables

- Review of basic definitions and notation
- Optimality conditions
- Descent methods
- Other methods
- Summary of methods

Subsection 1

Review of basic definitions and notation

Review of basic definitions and notation

A vector $v \in \mathbb{R}^n$ is a column vector consisting of n components (numbers):

$$v = \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix},$$

sometimes written $v = (v_1, \ldots, v_n)$.

This is to be distinguished from the row vector

$$v^T = [v_1 \ \ldots \ v_n].$$

The *inner or dot product* of $u, v \in \mathbb{R}^n$ is

$$\langle u, v \rangle = u \cdot v = \sum_{i=1}^{n} u_i v_i.$$

We also write $u^T v$ for the inner product.

If θ is the angle between u and v then

$$u \cdot v = \cos(\theta) \|u\| \|v\|.$$

where for $u \in \mathbb{R}^n$,

$$\|u\| = \sqrt{u \cdot u} = \left(\sum_i u_i^2 \right)^{1/2}.$$

Matrices

The *vector space* of $m \times n$ matrices is written $\mathbb{R}^{m \times n}$.

Let $B \in \mathbb{R}^{n \times n}$, then B_{ij} denotes the component of B in row i and column j.

B is *symmetric* if $B = B^T$ (transpose of B).

B is *positive definite* if $u^T Bu > 0$ for each nonzero vector $u \in \mathbb{R}^n$.

B is *positive semi-definite* if $u^T Bu \geq 0$ for each nonzero $u \in \mathbb{R}^n$.

Matrices

A theorem from linear algebra:

Theorem (Symmetric Matrices)

If B is symmetric then it is positive (semi-)definite if and only if each of its eigenvalues is positive (respectively nonnegative).

This fact gives us an easier way of checking the positive definiteness or semi-definiteness of symmetric matrices.

The gradient vector

The gradient vector of a function $f(x) : \mathbb{R}^n \rightarrow \mathbb{R}$, is defined as the column vector

$$\nabla f(x) = \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} \end{bmatrix}.$$

For a given point $x_0 \in \mathbb{R}^n$, it can be shown that the gradient vector at that point, $\nabla f(x_0)$, is a vector normal (at right angles) to the tangent plane of the curve defined by the set of points $\{x \in \mathbb{R}^n \ : \ f(x) = f(x_0)\}$, at the point x_0 (see 2nd year vector calculus).

The directional derivative

The directional derivative of f at x in the direction d is defined as

$$f'(x;d) \stackrel{def}{=} \lim_{t \to 0} \frac{f(x+td) - f(x)}{t}$$

This is essentially the one-dimensional derivative of the slice of f along the (arbitrary) direction d. To say that f is differentiable at x, we require that directional derivatives exist for all directions d.

It is not difficult to show that

$$f'(x;d) = \nabla f(x)^T d.$$

We say f is C^1 or continuously differentiable if it is differentiable and the gradient function ∇f is continuous.

The Hessian matrix

The second derivate matrix $\nabla^2 f(x)$ is called the *Hessian* of f at x.

It can be written using partial derivatives, namely the component in row i and column j of $\nabla^2 f(x)$ is

$$\nabla^2 f(x)_{ij} = \frac{\partial}{\partial x_j}\frac{\partial f}{\partial x_i} = \frac{\partial^2 f(x)}{\partial x_j \partial x_i}. \tag{3.1}$$

We say f is C^2 (twice continuously differentiable) if f is twice differentiable and the Hessian function $\nabla^2 f$ is continuous.

Taylor series approximations

Consider a small variation Δx from a given vector x^*. Taylor's theorem says that the first order approximation of f at $x^* + \Delta x$, when f is C^1, is given by

$$f(x^* + \Delta x) \approx f(x^*) + \nabla f(x^*)^T \Delta x$$

and the 2nd order approximation, when f is C^2, is given by

$$f(x^* + \Delta x) \approx f(x^*) + \nabla f(x^*)^T \Delta x + \frac{1}{2} \Delta x^T \nabla^2 f(x^*) \Delta x$$

Compare with the second order Taylor series approximation of a function of one variable

$$f(x^* + \delta x) \approx f(x^*) + \delta x f'(x^*) + \frac{(\delta x)^2}{2} f''(x^*)$$

Subsection 2

Optimality conditions

Unconstrained optimisation of functions of n variables

Throughout this section, $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable (C^1) function.

We shall frequently use definitions and concepts that are discussed in Chapter 1 of the Notes. If you don't understand the definition of something that we are talking about, the first place you should look is there.

Our problem of interest is an Unconstrained Optimisation Problem

$$\min_x f(x). \tag{3.2}$$

Definitions of Minimality

Definitions (Global and Local Minima)

A *local minimum* of f is a point $x^* \in \mathbb{R}^n$ for which there exists ϵ such that for each x with $||x - x^*|| < \epsilon$, we have $f(x^*) \leq f(x)$. Thus for each x "near" x^*, we have $f(x^*) \leq f(x)$.

A *global minimum* of f is a point $x^* \in \mathbb{R}^n$ such that for every $x \in \mathbb{R}^n$, $f(x^*) \leq f(x)$.

We will show that at a local minimum, the gradient of a differentiable function is zero; this condition is called a *first-order necessary* (optimality) condition.

First-Order Necessary Optimality Condition

Proposition (Local Minimality)

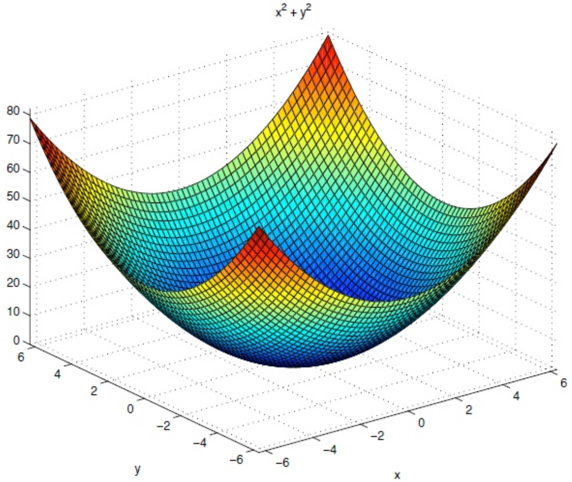
Let f be a C^1 function. If x^* is a local minimum of f then $\nabla f(x^*) = 0$.

Sketch proof: If x^* is a local minimum, then we expect the first order cost variation in any direction Δx

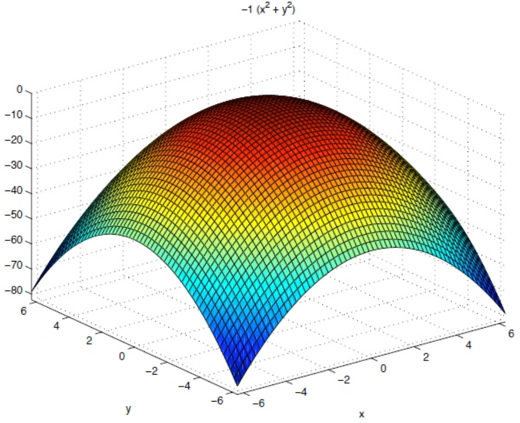
$$f(x^* + \Delta x) - f(x^*) \approx \nabla f(x^*)^T \Delta x$$

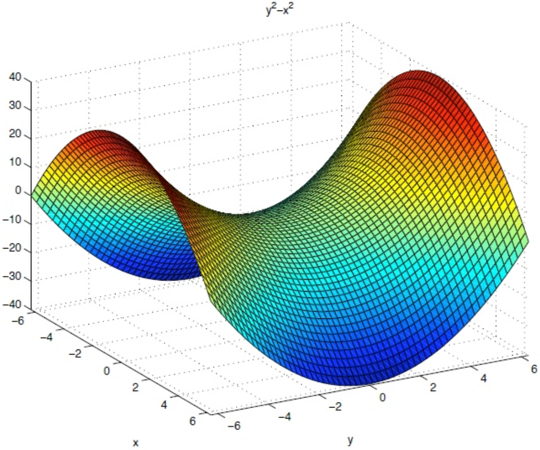
to be non-negative for suitably "small" Δx. Since the components of Δx can be positive or negative, then $\nabla f(x^*)^T \Delta x \geq 0$ for any Δx iff $\nabla f(x^*) = 0$.

Definition: We say x^* is *stationary* for f if $\nabla f(x^*) = 0$. Stationary points include local minima, local maxima and saddle points.



x2 + y2
80
70
60
50
40
30
20
10
0
6
4
2
0
−2
−4
−6
y
−6
−4
−2
0
2
4
6
x


−1 (x^2 + y^2)
0
−10
−20
−30
−40
−50
−60
−70
−80
6
4
2
0
−2
−4
−6
y
−6
−4
−2
0
2
4
6
x



y²−x²
40
30
20
10
0
−10
−20
−30
−40
−6
−4
−2
0
2
4
6
x
−6
−4
−2
0
2
4
6
y

Example 3.1

Consider the function of two variables (x_1, x_2)

$$f(x) = x_1^2 x_2 - 4x_1 x_2 + x_2^3 + x_2.$$

1. Find all stationary points.

$$\nabla f = \begin{bmatrix} 2x_1x_2 - 4x_2 \\ x_1^2 - 4x_1 + 3x_2^2 + 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\begin{aligned}
\Leftrightarrow \quad & 2x_2(x_1 - 2) = 0 \text{ and } x_1^2 - 4x_1 + 3x_2^2 + 1 = 0 \\
\Leftrightarrow \quad & (x_1 = 2 \text{ or } x_2 = 0) \text{ and } x_1^2 - 4x_1 + 3x_2^2 + 1 = 0
\end{aligned}$$

$$\Leftrightarrow \quad x = \begin{bmatrix} 2 \\ 1 \end{bmatrix} \text{ or } \begin{bmatrix} 2 \\ -1 \end{bmatrix} \text{ or } \begin{bmatrix} 2+\sqrt{3} \\ 0 \end{bmatrix} \text{ or } \begin{bmatrix} 2-\sqrt{3} \\ 0 \end{bmatrix}.$$

Thus f has the four stationary points given above.

Minimality of Convex Functions

An important result is that the first-order necessary condition is also sufficient if f is convex.

Definitions (Convex Function)

We say f is *convex* if for each $x, y \in \mathbb{R}^n$ and $\alpha \in [0, 1]$,

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

One way to test for convexity is as follows.

Proposition (Convexity Condition)

Suppose f is C^2. Then f is convex if and only if $\nabla^2 f(x)$ is positive semi-definite for every $x \in \mathbb{R}^n$.

Minimality of Convex Functions

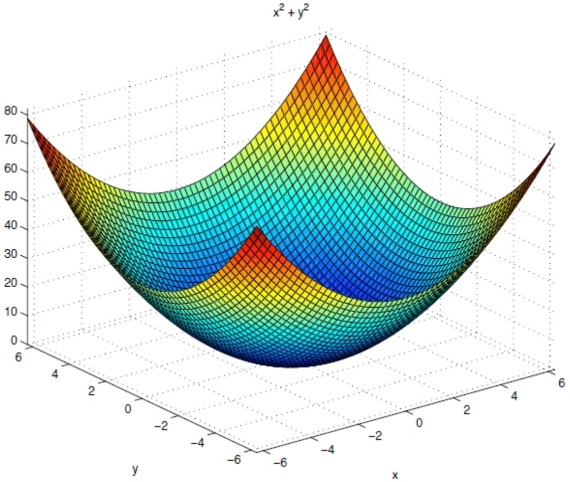
Once we know a function is convex, we have the following useful results.

Proposition (Minima for Convex Functions)

1. Any local minimum of a convex function is also a global minimum.
2. If f is convex and C^1 then $\nabla f(x^*) = 0$ if and only if x^* is a global minimum of f.

Proposition (Hessian of a Convex Function)

If f is convex then $\nabla^2 f(x)$ is invertible if and only if $\nabla^2 f(x)$ is positive definite.



$x^2 + y^2$
80
70
60
50
40
30
20
10
0
6
4
2
0
−2
−4
−6
y
−6
−4
−2
0
2
4
6
x

Example 3.2

Consider the quadratic function $f(x) = \frac{1}{2}x^T Bx + c^T x + \alpha$, where

B is symmetric and positive definite. Then f is convex, since $\nabla^2 f(x) = B$ and B is positive definite. By the Proposition (Hessian of a Convex Function), B in this example is invertible, and so

$$\begin{aligned} \nabla f(x) = 0 &\iff Bx + c = 0 \\ &\iff Bx = -c \\ &\iff x = -B^{-1}c. \end{aligned}$$

Since f is convex, $x = -B^{-1}c$ must be the global minimum of f. Furthermore, since $x = -B^{-1}c$ is the *only* point at which $\nabla f(x)$ is zero, it must be the *unique* global minimum of f.

Second order necessary and sufficient conditions

For a possibly non-convex but C^2 function, to ensure that a stationary point x^* is a local minimum we appeal to a *second-order sufficient condition*.

Proposition (2nd Order Condition for Minimality)

Let f be C^2. If $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive definite then x^* is a local minimum of f.

Proposition (Second Order Condition for Minimality)

Sketch Proof: Consider the second order cost variation due to a small variation $\Delta x \neq 0$ from a local minimum x^*

$$f(x^* + \Delta x) - f(x^*) \approx \nabla f(x^*)^T \Delta x + \frac{1}{2} \Delta x^T \nabla^2 f(x^*) \Delta x$$

Since $\nabla f(x^*)^T \Delta x = 0$ (first-order necessary condition), then the second order necessary and sufficient condition corresponds to

$$\Delta x^T \nabla^2 f(x^*) \Delta x > 0$$

which implies that at x^*, the second order cost variation is strictly positive.

Example 3.3

Consider the function of two variables (x_1, x_2)

$$f(x) = x_1^2 x_2 - 4x_1 x_2 + x_2^3 + x_2.$$

1. Find all stationary points.
2. Check the second-order sufficiency condition for a local minimum at each point. Determine whether or not each stationary point is a local minimum.

Four stationary points were found in Example 3.1:

$$x = \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ -1 \end{bmatrix}, \begin{bmatrix} 2+\sqrt{3} \\ 0 \end{bmatrix}, \begin{bmatrix} 2-\sqrt{3} \\ 0 \end{bmatrix}.$$

We now check the second order sufficiency condition for a local minimum at each stationary point.

$$\nabla^2 f(x) = \begin{bmatrix} 2x_2 & 2x_1 - 4 \\ 2x_1 - 4 & 6x_2 \end{bmatrix}$$

so

$$\nabla^2 f(\begin{bmatrix} 2 \\ 1 \end{bmatrix}) = \begin{bmatrix} 2 & 0 \\ 0 & 6 \end{bmatrix}$$

which is positive definite: it has two positive eigenvalues, $\lambda = 2, 6$.

The second order sufficiency condition is satisfied and so $x = (2, 1)$ is a local minimum of f.

For the next stationary point:

$$\nabla^2 f\left(\begin{bmatrix} 2 \\ -1 \end{bmatrix}\right) = \begin{bmatrix} -2 & 0 \\ 0 & -6 \end{bmatrix}.$$

This is *not* positive definite, so the sufficient condition for a local minimum has not been met. In fact, the Hessian here is *negative* definite, since both eigenvalues are negative, $\lambda = -2, -6$.

(In fact $x = (2, -1)$ is a local minimum of $-f$, and hence a local *maximum* of f.)

For the third stationary point:

$$\nabla^2 f(\begin{bmatrix} 2+\sqrt{3} \\ 0 \end{bmatrix}) = \begin{bmatrix} 0 & 2\sqrt{3} \\ 2\sqrt{3} & 0 \end{bmatrix}.$$

which is not positive definite (and nor is it negative definite), since it has both a positive and a negative eigenvalue, $\lambda = \pm 2\sqrt{3}$, so the second order sufficiency conditions are not satisfied.

This point is not a local minimum, but how do we show this?

Note that $(1, -1)$ is the eigenvector of the Hessian corresponding to the eigenvalue $-2\sqrt{3}$. We will consider moving from the stationary point in the direction of this eigenvector.

Let $\epsilon > 0$ and consider

$$\begin{aligned}
& f(\begin{pmatrix} 2+\sqrt{3} \\ 0 \end{pmatrix} + \epsilon \begin{pmatrix} 1 \\ -1 \end{pmatrix}) \\
= & f(\begin{pmatrix} 2+\sqrt{3}+\epsilon \\ -\epsilon \end{pmatrix}) \\
= & -\epsilon(2+\sqrt{3}+\epsilon)^2 + 4\epsilon(2+\sqrt{3}+\epsilon) - \epsilon^3 - \epsilon \\
= & -\epsilon\left((2+\sqrt{3}+\epsilon)(-2+\sqrt{3}+\epsilon) + \epsilon^2 + 1\right) \\
= & -2\epsilon^2\left(\epsilon+\sqrt{3}\right) \\
< & \, 0 = f(\begin{pmatrix} 2+\sqrt{3} \\ 0 \end{pmatrix}).
\end{aligned}$$

So some points "near" $x = (2+\sqrt{3}, 0)$ have a lower function value; x cannot be a local minimum.

In general:

- moving a little way along the eigenvector of the Hessian corresponding to a negative eigenvalue will lead to a decrease in the objective function,
- moving a little way along the eigenvector corresponding to a positive eigenvalue will lead to an increase in the objective function.

To see why, consider expanding f in a Taylor series near the stationary point x^*:

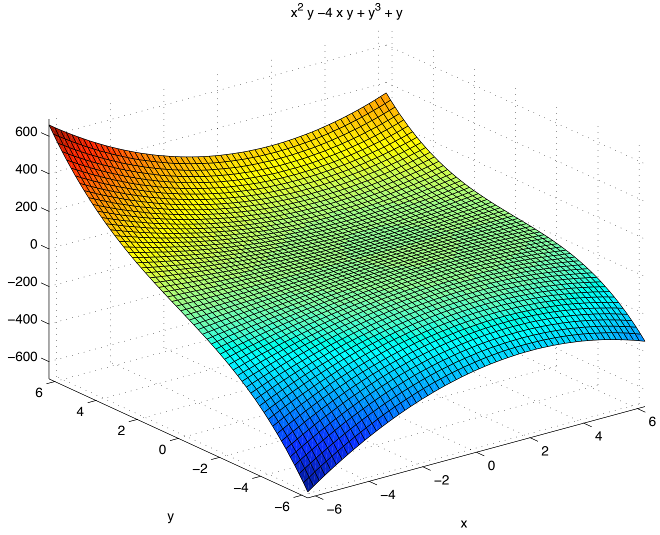
$$f(x^* + \Delta x) \approx f(x^*) + \nabla f(x^*)^T(\Delta x) + \frac{1}{2}(\Delta x)^T \nabla f^2(x^*)(\Delta x).$$

Now the fact that x^* is stationary means $\nabla f(x^*) = 0$.

Consider points a little way along an eigenvector y of the Hessian corresponding to eigenvalue $\lambda \neq 0$, that is points $\Delta x = \epsilon y$ where $\nabla f^2(x^*)y = \lambda y$, then

$$\begin{aligned} f(x) &\approx f(x^*) + 0^T(\epsilon y) + \tfrac{1}{2}(\epsilon y)^T \nabla f^2(x^*)(\epsilon y) \\ &= f(x^*) + \tfrac{1}{2}\epsilon^2 y^T \nabla f^2(x^*) y \\ &= f(x^*) + \tfrac{1}{2}\epsilon^2 y^T \lambda y \\ &= f(x^*) + \tfrac{1}{2}\epsilon^2 \lambda ||y||^2. \end{aligned}$$

Clearly $\frac{1}{2}\epsilon^2||y||^2 > 0$ for y nonzero, and $f(x) < f(x^*)$ if $\lambda < 0$, while $f(x) > f(x^*)$ if $\lambda > 0$. Thus, if the Hessian at a stationary point has both a negative *and* a positive eigenvalue, then the stationary point cannot be either a local maximum or a local minimum.



$x^2 y - 4 x y + y^3 + y$
600
400
200
0
−200
−400
−600
y
6
4
2
0
−2
−4
−6
x
−6
−4
−2
0
2
4
6

Exercises. For the following functions, (i) find all stationary points, and for each stationary point x, (ii) check whether the second-order sufficient condition holds at x, and thus (iii) state whether x is a local minimum.

1. $f(x_1, x_2) = x_1 + x_2 + \frac{1}{2}x_1^2 - 4x_1x_2 + \frac{1}{2}x_2^2$
2. $$f(x_1, x_2) = [1, 1]^T x + \frac{1}{2}x^T \begin{bmatrix} 1 & -4 \\ -4 & 1 \end{bmatrix} x$$
3. $f(x) = 3x_1^3x_2 - 3x_1x_2 + (x_2 + e)\log(x_2 + e)$, where the logarithm is the natural logarithm
4. $f(x) = x_1^2 - 5x_1x_2 + x_2^4 - 25x_1 - 8x_2$, where you may find it helpful to note that $y = 3$ is a root of the polynomial $8y^3 - 25y - 141$

Summary

Proposition (Local Minimality)

Let f be a C^1 function. If x^* is a local minimum of f then $\nabla f(x^*) = 0$.

Proposition (Minima for Convex Functions)

If f is convex and C^1 then $\nabla f(x^*) = 0$ if and only if x^* is a global minimum of f.

Proposition (2nd Order Condition for Minimality)

Let f be C^2. If $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive definite then x^* is a local minimum of f.

Subsection 3

Descent methods

Descent methods

In our discussion of single-variable optimisation methods, we discussed both methods that assumed that we could solve $f'(x) = 0$ analytically and iterative methods which did not make this assumption.

The multivariable optimisation methods that we have just considered are analogous to first class of single variable methods - essentially they calculate the stationary points by solving $\nabla f(x) = 0$ and then working out which of the stationary points are local minima.

In the next few lectures we will study iterative methods for multivariable problems which do not assume that we can solve $\nabla f(x) = 0$ analytically (though we do assume that we can **calculate** $\nabla f(x)$ and in some cases, $\nabla^2 f(x)$).

Descent methods

Our aim is to find

$$\min_{x} f(x). \tag{3.3}$$

Recursive methods for doing this often involve evaluating the function f and its derivative ∇f at some point, choosing a direction in which f decreases and moving along that direction for some distance.

Definition:
Given $x \in \mathbb{R}^n$, a vector $d \in \mathbb{R}^n$ is a *descent direction* for f at x if

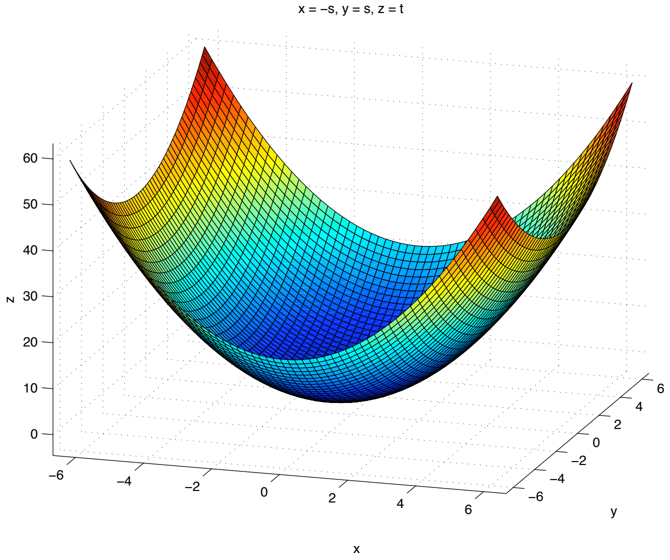
$$\nabla f(x)^T d \ < \ 0.$$

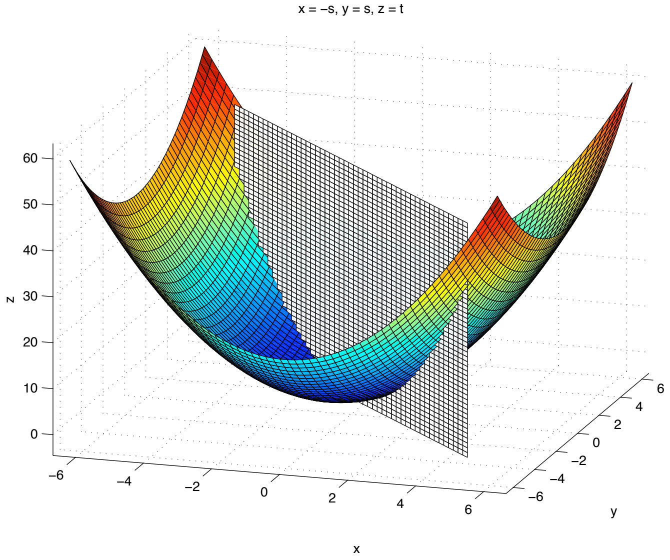
Such a direction d is called a descent direction because the function value decreases as we move along it, starting from x, at least initially.

Descent methods

To help you understand why this is so, consider the one-dimensional slice of f in the direction d. This is a function of a single variable. The derivative of this function is given by the directional derivative, $f'(x; d) = \langle \nabla f, d \rangle$. If this is negative, that is if $f'(x; d) < 0$, then the slope of this one-dimensional slice of f is negative as we move along d.

So f decreases, at least initially, as we move along d from x. In other words, d provides a direction of function *descent* from x.


x = –s, y = s, z = t
z
60
50
40
30
20
10
0
–6
–4
–2
0
2
4
6
x
6
4
2
0
–2
–4
–6
y


x = −s, y = s, z = t
60
50
40
30
20
10
0
z
−6
−4
−2
0
2
4
6
x
6
4
2
0
−2
−4
−6
y

Step Size for Descent

Once we have chosen a descent direction d at x, to decrease the value of f we must choose a *step size* or *step length* $t > 0$ such that

$$f(x + td) < f(x).$$

Example 3.4

Consider the unconstrained minimization problem

$$\min_x f(x) = x^2 - 2x$$

where $x \in \mathbb{R}$, and suppose we start at the point $x = 0$.

Here $\nabla f = 2x - 2$.
From a sketch of f, it is clear that $f(t) = f(0 + t \times 1)$ is less than $f(0)$ for small $t > 0$, i.e. it appears that $d = 1$ is a descent direction at $x = 0$.

To check that $d = 1$ is a descent direction, observe that

$$\nabla f(0)^T d \;=\; (-2) \times 1 \;=\; -2 \;<\; 0,$$

so 1 is a descent direction at 0.

Let's try some different step sizes.

Step size $\mathbf{t = 1}$ actually minimises $f(t)$: $f(1) = -1$.

Step size $\mathbf{t = \frac{1}{2}}$ yields a decrease in f:

$$f(\frac{1}{2}) = \frac{1}{4} - 1 = -\frac{3}{4} < 0 = f(0)$$

Step size $\mathbf{t = 3}$ causes an increase in f (bad!):

$$f(3) = 9 - 6 = 3 > 0 = f(0).$$

Summary - Descent methods

Our aim is to find $\min_x f(x)$ where $x \in \mathbb{R}^n$.
Recursive methods for doing this often involve evaluating the function f and its derivative ∇f at some point, choosing a direction in which f decreases and moving along that direction for some distance.

Definition:
Given $x \in \mathbb{R}^n$, a vector $d \in \mathbb{R}^n$ is a *descent direction* for f at x if

$$\nabla f(x)^T d \ < \ 0.$$

Once we have chosen a descent direction d at x, to decrease the value of f we must choose a *step size* or *step length* $t > 0$ such that

$$f(x + td) < f(x).$$

Example 3.5

Consider

$$f(x) = x_1^2 + \frac{1}{2}x_2^2$$

where $x = (x_1, x_2) \in \mathbb{R}^2$ (so f is a paraboloid surface). Given $x = (1, 0)$, what directions d are descent directions for f at x ?

Solution:

Note that $\nabla f = \left(\frac{\partial f(x)}{\partial x_1}, \frac{\partial f(x)}{\partial x_2}\right) = (2x_1, x_2)$. So $\nabla f(1, 0) = (2, 0)$ and $\nabla f(1, 0)^T d = 2d_1$, where $d = (d_1, d_2)$; hence

$$\nabla f(1,0)^T d \; < \; 0 \quad \iff \quad 2d_1 \; < \; 0 \quad \iff \quad d_1 \; < \; 0.$$

For instance, each of $(-1, 2)$, $(-2, 0)$, and $(-0.00000001, 10000)$ is a descent direction at $(1, 0)$.

Step Sizes for Example 3.5:

Let us consider step sizes for the first two of these descent directions. To do so, consider the one-dimensional slice of the function in each of these directions, positioned with the origin at $x = (1, 0)$. Note that the function value at this point is 1, i.e., $f(1, 0) = 1$.

Case 1: We first examine direction $d = (-1, 2)$, and consider the function

$$\begin{aligned}\phi_1(t) &= f(x + td) = f((1,0) + t(-1,2)) = f(1 - t, 2t)\\ &= (1-t)^2 + \frac{1}{2}(2t)^2 = 3t^2 - 2t + 1.\end{aligned}$$

Step Sizes for Example 3.5:

Sketching ϕ_1, we see it decreases as t increases from zero, and has a minimum value of $\frac{2}{3}$ at $t = \frac{1}{3}$.

So if we choose to go in this direction, we get the most function decrease by choosing a step size of $t = \frac{1}{3}$.

In this case, the new point would be $x := x + td = (1, 0) + \frac{1}{3}(-1, 2) = (\frac{2}{3}, \frac{2}{3})$, which has function value $f(x) = \frac{2}{3}$, as expected. This new point has function value less than that of the old point (recall $f(1, 0) = 1$), but it is not the minimum value of f, so we need to try again: we should seek a descent direction at this new point, determine a good step size, and repeat.

Step Sizes for Example 3.5:

Case 2: Now examine direction $d = (-2, 0)$, and consider the function

$$\begin{aligned}\phi_2(t) &= f(x + td) = f((1, 0) + t(-2, 0)) = f((1 - 2t, 0)) \\ &= (1 - 2t)^2.\end{aligned}$$

Sketching ϕ_2, we see that it also decreases as t increases from zero, and has a minimum value of 0 at $t = \frac{1}{2}$. So for this case, we get the maximum decrease by choosing a step size of $t = \frac{1}{2}$.

The new point is $x := x + td = (1, 0) + \frac{1}{2}(-2, 0) = (0, 0)$, which has function value $f(x) = 0$. In fact, this is the minimum value for f, so this choice of descent direction was good: it got us to a minimising point in one step!

Moving in a Descent Direction decreases f

We now prove formally that f decreases initially as we move along a descent direction.

Proposition (Descent Directions)

If $x, d \in \mathbb{R}^n$ and d is a descent direction for f at x, then for sufficiently small $t > 0$,

$$f(x + td) \; < \; f(x).$$

In fact given any constant $\sigma \in (0, 1)$, then for sufficiently small $t > 0$,

$$f(x + td) \; \leq \; f(x) + t\sigma \nabla f(x)^T d \; < \; f(x). \tag{3.4}$$

Proof of Proposition:

Let d be a descent direction for f at x, and $\sigma \in (0, 1)$. Now for a scalar t we have, by Taylor's Theorem:

$$f(x + td) = f(x) + t\nabla f(x)^T d + e(t) \tag{3.5}$$

where the "error term" $e(t)$ is $o(t)$, that is $e(t)/t \to 0$ as $t \to 0$, $t \neq 0$. Since $-\nabla f(x)^T d > 0$, $1 - \sigma > 0$ and $e(t)/t \to 0$ as $t \to 0$, then for sufficiently small $t > 0$,

$$-(1 - \sigma)\nabla f(x)^T d > e(t)/t,$$

hence $e(t) < -t(1 - \sigma)\nabla f(x)^T d$. Substituting into Equation 3.5:

$$f(x + td) < f(x) + t\sigma\nabla f(x)^T d < f(x). \tag{3.6}$$

Class Exercise 3.6

Again, consider Example 3.5:

$$f(x) = x_1^2 + \frac{1}{2}x_2^2$$

where $x = (x_1, x_2) \in \mathbb{R}^2$.

Given $x = (\frac{2}{3}, \frac{2}{3})$, determine a general condition giving *all* descent directions $d = (d_1, d_2)$ for f at x. Also, write down a few specific examples of descent directions for f at x.

Class exercise, s. 178

$$f(x) = x_1^2 + \tfrac{1}{2} x_2^2$$

$$\nabla f(x) = \begin{bmatrix} 2x_1 \\ x_2 \end{bmatrix} \quad ; \quad \text{at } x_0 = (2/3,\ 2/3)$$

$$\nabla f(x_0) = \begin{bmatrix} 4/3 \\ 2/3 \end{bmatrix}$$

we look for descent directions at $x = (2/3, 2/3)$

$$\nabla f(x)^T \cdot d < 0$$

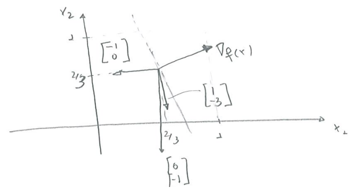
$$\begin{bmatrix} 4/3 & 2/3 \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} < 0 \quad \Rightarrow \quad \tfrac{4}{3} d_1 + \tfrac{2}{3} d_2 < 0$$

$$4 d_1 + 2 d_2 < 0$$

$$d_2 < -2 d_1$$

Some descent directions:

$$d = \begin{bmatrix} 1 \\ -3 \end{bmatrix} \qquad d = \begin{bmatrix} 0 \\ -1 \end{bmatrix} \qquad d = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$$



A General Descent Method

To solve unconstrained optimisation problems, or at least to find a local minimum point, we will develop a general descent method.

We assume we cannot find stationary points analytically.

For the general descent method, we assume we are given a starting point $x^0 \in \mathbb{R}^n$, and then construct a sequence of *iterates* $x^k \in \mathbb{R}^n$, for $k = 1, 2 \ldots$ such that $f(x^{k+1}) < f(x^k)$ for each k.

To find x^{k+1} we choose a descent direction d^k for f at x^k, an appropriate step size $t_k > 0$, and set $x^{k+1} = x^k + t_k d^k$.

A General Descent Method

Algorithm (General Descent Method)

Start with $x^0 \in \mathbb{R}^n$.

set $k = 0$.

while x^k is not "satisfactory", do **Iteration** $k+1$:

1. Given x^k and k, "choose" a descent direction d^k for f at x^k.
2. "Choose" a step size $t_k > 0$ such that $f(x^k + t_k d^k) < f(x^k)$.
3. Let $x^{k+1} := x^k + t_k d^k$, $k := k+1$.

end while.

A General Descent Method

To make this method practical, we need to be explicit about the terms in quotation marks. In particular to specify fully a descent method, we need to decide two main issues:

1. *which* descent direction do we choose to move in, and
2. *how far* do we step along that direction?

There is also the issue of when to stop the method, that is of how to know whether the iterate is "satisfactory" or not.

A General Descent Method

Defining an optimality condition

From the first-order optimality condition, x^k is "satisfactory" if it is stationary: $\nabla f(x^k) = 0$.

In floating point arithmetic, we have to implement this by choosing $\epsilon > 0$ and calling x^k satisfactory if $\|\nabla f(x^k)\| < \epsilon$ (unsatisfactory means $\|\nabla f(x^k)\| \geq \epsilon$).

A General Descent Method

Choosing a descent direction

In comparing different descent directions to try to decide which one to use, we need to be aware of the effect of *scaling*.

One might try to choose the direction in which the directional derivative is as small as possible, ie, choose the direction which minimises $f'(x; d) = \nabla f(x)^T d$. But consider the effect of multiplying d by a scalar.

Suppose $\nabla f(x)^T d < 0$. Then

$$\nabla f(x)^T (2d) = 2\nabla f(x)^T d < \nabla f(x)^T d < 0.$$

Doubling the direction vector makes the direction seem twice as steep. Clearly d and $2d$ point in exactly the same direction, so to fairly compare different directions, we must scale them to have the same *length*.

A General Descent Method

Choosing a descent direction

We seek a direction which shows the steepest descent, out of all those with a fixed length, say length 1. We thus consider

$$\begin{cases} \min\limits_{d} & \langle \nabla f(x), d \rangle \\ \text{s.t.} & ||d|| = 1 \\ & d \in \mathbb{R}^n \end{cases} = \begin{cases} \min\limits_{d} & \cos\theta ||\nabla f(x)|| \cdot ||d|| \\ \text{s.t.} & ||d|| = 1 \\ & d \in \mathbb{R}^n \\ & \theta \text{ is the angle between } d \text{ and } \nabla f(x) \end{cases}$$

$$= \begin{cases} \min\limits_{\theta} & \cos\theta ||\nabla f(x)|| \\ \text{s.t.} & \theta \in \mathbb{R}. \end{cases}$$

A General Descent Method

Choosing a descent direction

Since $\cos\theta \in [-1, 1]$, the minimum is achieved at $\cos\theta = -1$, that is at $\theta = \pi$. Now there is only one direction that can be at an angle of π from $\nabla f(x)$, and that is the "opposite" direction, $-\nabla f(x)$.

Thus the obvious choice of a descent direction for f at x is $d = -\nabla f(x)$. In fact for this d, either $\langle \nabla f(x), d \rangle < 0$ or x is stationary. To see this, observe that

$$\langle \nabla f(x), d \rangle \quad = \quad \langle \nabla f(x), -\nabla f(x) \rangle \quad = \quad -\|\nabla f(x)\|^2 \quad \leq \quad 0.$$

A General Descent Method

Choosing a descent direction

Definitions (steepest descent)

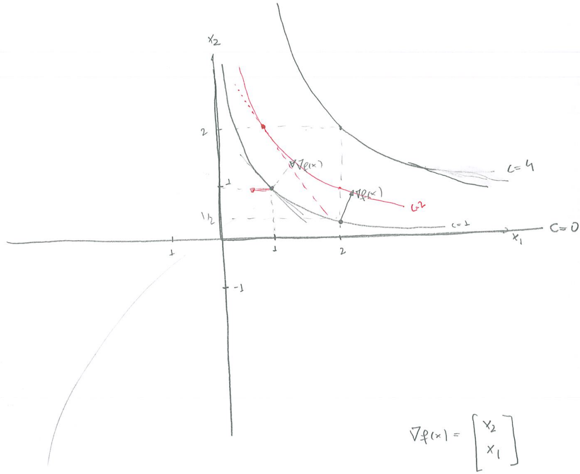
- The direction $d = -\nabla f(x)$ is called the *steepest descent* direction for f at x.
- The *Steepest Descent Method* is the general descent method with $d^k := -\nabla f(x^k)$ for each k.

This steepest descent direction/method is just one of many descent directions/methods.

Homework Exercise.

Consider the function $f(x) = x_1 x_2$. Sketch contour lines for f, that is lines of the form $f(x) = c$, for constant values c, in the x_1-x_2 plane. Make sure you include values $c = 0, 1, 2$. For each of the points $(0, 1)$, $(1, 1)$, and $(1, 2)$, describe all descent directions for f at the point, plot the point on your sketch, and indicate all descent directions at each point. In each case, what is the steepest descent direction?

Homework exercise 189



$$\nabla f(x) = \begin{bmatrix} x_2 \\ x_1 \end{bmatrix}$$

At point (1,1) :

$$[1 \ 1]\begin{bmatrix} d_1 \\ d_2 \end{bmatrix} < 0 \Rightarrow d_1 < -d_2$$

$$x_1 x_2 = c$$

$$x_1 = c/x_2$$

At point (0,1)

$$[1 \ 0]\begin{bmatrix} d_1 \\ d_2 \end{bmatrix} < 0 \Rightarrow d_1 < 0$$

At point (1,2)

$$[2, 1]\begin{bmatrix} d_1 \\ d_2 \end{bmatrix} < 0 \Rightarrow 2d_1 < -d_2$$

A General Descent Method

Choosing a step size

The step size can be determined by

- using any of the methods for finding the minimum of a function on the interval $[0, \infty)$ (Chapter 2 of the Notes); or
- choosing a step size that may not minimise the single variable function, but does satisfy the Armijo-Goldstein and Wolff conditions.

This is what we meant when we said that solution of single-variable optimisation problems is often a component of methods for solving multivariable optimisation problems.

These choices lead to **The Steepest Descent Algorithm** in which, at each step, we choose the steepest descent direction as our descent direction and the step size to be that which minimises the function in the steepest descent direction.

Summary - General Descent Method

Algorithm (General Descent Method)

Start with $x^0 \in \mathbb{R}^n$.

set $k = 0$.

while x^k is not "satisfactory", do **Iteration** $k+1$:

1. Given x^k and k, "choose" a descent direction d^k for f at x^k.
2. "Choose" a step size $t_k > 0$ such that $f(x^k + t_k d^k) < f(x^k)$.
3. Let $x^{k+1} := x^k + t_k d^k$, $k := k+1$.

end while.

A good choice of descent direction is $d^k = -\nabla f(x^k)$, the *steepest descent* direction for f at x^k.

Subsubsection 1

Steepest Descent

The Steepest Descent Method

The choices discussed in the previous lecture lead to **The Steepest Descent Algorithm** in which, at each step, we choose the steepest descent direction as our descent direction and the step size to be that which minimises the function in the steepest descent direction.

The Steepest Descent Method

Minimise a unimodal function $f : \mathbb{R}^n \to \mathbb{R}$ to within tolerance ϵ.

Algorithm (Steepest Descent Method)

1. **Select** $x^0 \in \mathbb{R}^n$.
   Set $k = 0$.
2. **Calculate** $d^k = -\nabla f(x^k)$.
   If $||d^k|| < \epsilon$ **then** stop.
3. **Select** step length t_k by solving the single-variable minimisation problem: $\min q(t) = f(x^k + t_k d^k)$.
4. **Set** $x^{k+1} = x^k + t_k d^k$.
   Set $k = k + 1$.
   Return to step 2.

Class Exercise 3.7

Consider the function $f(x) = x_1^2 + x_2^2/4$. Apply one step of the steepest descent method, starting from $x^0 = (1, 1)$.

Homework exercise 196

$$f(x) = x_1^2 + x_2^2/4$$

$$\nabla f(x) = \begin{bmatrix} 2x_1 \\ \frac{1}{2}x_2 \end{bmatrix} \quad \text{at} \quad x_0 = (1 \quad 1) \rightarrow \nabla f(x_0) = \begin{bmatrix} 2 \\ \frac{1}{2} \end{bmatrix}$$

$$d = \begin{bmatrix} -2 \\ -\frac{1}{2} \end{bmatrix}$$

$$x_{k+1} = (1-2t)^2 + (1 - \tfrac{1}{2}t)^2/4$$

$$= 4t^2 - 4t + 1 + \frac{\frac{1}{4}t^2 - t + 1}{4}$$

$$= \frac{16t^2 - 16t + 4 + \frac{1}{4}t^2 - t + 1}{4} = \frac{\frac{65}{4}t^2 - 17t + 5}{4}$$

this function is minimised for t in

$$\frac{65}{2}t - 17 = 0 \Rightarrow t = 34/65$$

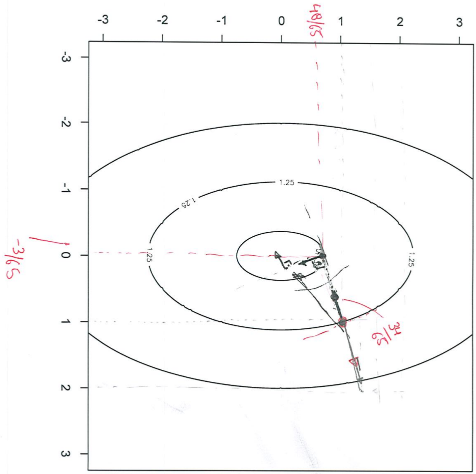
$$f'' = 65/2 > 0 \quad \underline{\text{minimum}}$$

$$x_{k+1} = (1 - 68/65 ; 1 - 17/65) = (-3/65 , 48/65)$$

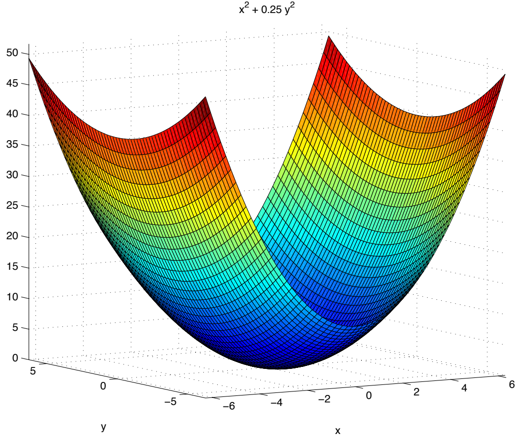
$$\nabla f(x_{k+1}) = \begin{bmatrix} -6/65 \\ 24/65 \end{bmatrix}$$

$$\nabla f(x_k) \cdot \nabla f(x_{k+1})$$

$$\begin{bmatrix} 2 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} -6/65 \\ 24/65 \end{bmatrix} = 0$$



-3
-2
-1
0
1
2
3
1.25
48/65
-3/65
34/65



$x^2 + 0.25\ y^2$
50
45
40
35
30
25
20
15
10
5
0
5
0
−5
y
−6
−4
−2
0
2
4
6
x

The Steepest Descent Method

It makes intuitive sense that the steepest descent algorithm in particular, and descent algorithms in general, should converge, but needs to be proved.
We will look at a theorem that justifies the use of a whole class of descent algorithms.

The angle θ_k between $-\nabla f(x^k)$ and d^k plays an important role.
Recall from that if d^k and $\nabla f(x^k)$ are nonzero, then θ_k is defined by

$$\cos(\theta_k) \;=\; -\frac{\langle \nabla f(x^k), d^k \rangle}{\|\nabla f(x^k)\| \, \|d^k\|}.$$

The Steepest Descent Method

Theorem (Condition for Convergence)

Let $f : \mathbb{R}^n \to \mathbb{R}$ be C^1, $x^0 \in \mathbb{R}^n$, and $0 < \sigma \leq \mu < 1$. Suppose, for each $k = 0, 1, \ldots$, we choose t_k such that the Armijo-Goldstein and Wolff conditions hold. If there is a constant v independent of k such that

$$0 \;<\; v \;\leq\; \cos(\theta_k), \tag{3.7}$$

where θ_k is the angle between $-\nabla f(x^k)$ and d^k, then every cluster (or limit) point x^* of the sequence $\{x^k\}$ is a stationary point of f.

Homework Exercise: Show that (3.7) holds for the steepest descent method.

The Steepest Descent Method

One property of the steepest descent algorithm, which is not desirable, is that each descent direction is perpendicular to the previous descent direction. Thus it is possible that the algorithm can take a very long time to converge.

It is possible to design other algorithms where this is not the case. Before we discuss these, we shall spend some time discussing the issue of convergence.

Summary – Steepest Descent Method

Minimise a unimodal function $f : \mathbb{R}^n \to \mathbb{R}$ to within tolerance ϵ.

Algorithm (Steepest Descent Method)

1. **Select** $x^0 \in \mathbb{R}^n$.
   Set $k = 0$.
2. **Calculate** $d^k = -\nabla f(x^k)$.
   If $||d^k|| < \epsilon$ **then** stop.
3. **Select** step length t_k by solving the single-variable minimisation problem: $\min q(t) = f(x^k + t_k d^k)$.
4. **Set** $k = k + 1$.
   Set $x^{k+1} = x^k + t_k d^k$.
   Return to step 2.

Subsubsection 2

Rates of convergence

Rates of convergence

Definition: Let $\{x^k\}$ converge to x^* in $\mathbb{R}^n$.

1. If, for some constant $c \in (0,1)$ and all large enough k, we have
$$\|x^{k+1} - x^*\| \ \leq \ c\|x^k - x^*\|,$$
then we say that the *rate of convergence of* $\{x^k\}$ *is linear*, or $x^k \to x^*$ *linearly* (as $k \to \infty$).
2. If, for some sequence $\{c_k\}$ of positive scalars such that $c_k \to 0$, we have
$$\|x^{k+1} - x^*\| \leq c_k\|x^k - x^*\|,$$
then the rate of convergence is *superlinear*, and $x^k \to x^*$ superlinearly.
3. If, for some $c > 0$ and all large enough k,
$$\|x^{k+1} - x^*\| \ \leq \ c\|x^k - x^*\|^2,$$
the rate of convergence is *quadratic*, and $x^k \to x^*$ *quadratically*.

Rates of convergence

Linear convergence is considered rather slow, especially compared with quadratic convergence.

Example 1: Consider the sequence $x^k = 10^{-k}$; then $x^k \to x^* = 0$ linearly. Actually we get one extra decimal place of accuracy per iteration, in the sense that x^{k+1} is one tenth the size of x^k (or ten times closer to x^*). So it takes 10 (respectively n) iterations for the iterates to be less than 10^{-10} (resp. 10^{-n}).

Example 2: Compare this with the sequence $x^k = 10^{-k^2} \to 0$; note $x^{k+1} = 10^{-k^2-2k-1}$ has $2k+1$ more decimal places of accuracy than x^k. In particular, it only takes 4 (resp. 5) iterations for the iterates to be less than 10^{-10} (10^{-20}).

Rates of convergence

Note: It is easy to show that superlinear convergence is also linear (although the converse is not true!): as $c_k \to 0$, there must be a k' such that for all $k \geq k'$, $c_k < 1$. Then take $c = \max_{k \geq k'} c_k$.

Class Exercise 3.8

Show that $10^{-k^2} \to 0$ superlinearly.

Solution

$$x^k = 10^{-k^2} \to 0$$

$$\|x^{k+1} - x^*\| = 10^{-(k+1)^2} = 10^{-k^2-2k-1} = 10^{-2k-1} 10^{-k^2}$$

$$= 10^{-2k-1} \|x^k - x^*\|$$

If we take $c_k = 10^{-2k-1}$, this shows that the sequence converges superlinearly.

Class Exercise 3.9

1. Show that $10^{-2^k} \to 0$ quadratically.

2. Show that quadratic convergence is also superlinear (but note that the converse is not true!).

Solutions to Class Exercises 3.9

1.
$$x^k = 10^{-2^k} \to 0$$

$$\|x^{k+1} - x^*\| = 10^{-2^{k+1}} = 10^{-2^k \cdot 2} = \left[10^{-2^k}\right]^2$$

so we can take $c = 1$ to satisfy the quadratic criterion.

2. To show that quadratic convergence is also superlinear, take $c_k = c\|x^k - x^*\|$. Note that if the sequence converges then this must converge to 0!

Rates of convergence

Class Exercise 3.10

Prove that the rate of convergence of the sequence $x^k = (\frac{k-1}{k+2}, \frac{1}{2})$ is *not* linear. [In fact it is slower than linear.]

Solution

$$x^k = (\frac{k-1}{k+2}, \frac{1}{2}) \to (1, \frac{1}{2})$$

$$\|x^k - x^*\| = \sqrt{\left(\frac{k-1}{k+2} - 1\right)^2} = \frac{1}{k+2}\sqrt{(k-1-k-2)^2} = \frac{3}{k+2}$$

$$\frac{\|x^{k+1} - x^*\|}{\|x^k - x^*\|} = \frac{3}{k+3}\frac{k+2}{3} = \frac{k+2}{k+3} \to 1$$

so there is no c which satisfies the linearity criterion. Hence this sequence converges slower than linearly.

Convergence of the "Grad"

In practice we don't know the value of the minimum x^*, but we do know that $\nabla f(x^*) = 0$.
So instead of looking at $\|x^k - x^*\|$ we look at $\|\nabla f(x^k) - \nabla f(x^*)\| = \|\nabla f(x^k)\|$, the norm of the "residual" $\nabla f(x^k)$.

Lemma (Convergence of the Residual)

Let f be C^2, $\nabla f(x^*) = 0$, and the Hessian matrix $\nabla^2 f(x^*)$ be invertible. If $x^k \to x^*$ then $\nabla f(x^k) \to \nabla f(x^*)$, and the rate of convergence of these two sequences is identical.

This justifies using a test of the form $\|\nabla f(x^k)\| < \epsilon$ when we write computer code to test whether x^k is "close enough" to a stationary point.

Summary – Rates of convergence

Definition: Let $\{x^k\}$ converge to x^* in $\mathbb{R}^n$.

1. If, for some constant $c \in (0,1)$ and all large enough k, we have
$$\|x^{k+1} - x^*\| \; \leq \; c\|x^k - x^*\|,$$
then we say that the *rate of convergence of* $\{x^k\}$ *is linear*, or $x^k \to x^*$ *linearly* (as $k \to \infty$).
2. If, for some sequence $\{c_k\}$ of positive scalars such that $c_k \to 0$, we have
$$\|x^{k+1} - x^*\| \leq c_k\|x^k - x^*\|,$$
then the rate of convergence is *superlinear*, and $x^k \to x^*$ superlinearly.
3. If, for some $c > 0$ and all large enough k,
$$\|x^{k+1} - x^*\| \; \leq \; c\|x^k - x^*\|^2,$$
the rate of convergence is *quadratic*, and $x^k \to x^*$ *quadratically*.

Subsubsection 3

Newton's Method

Newton's method

To increase the rate of convergence of an iterative descent method, we must use or approximate second-order information, that is knowledge of the Hessian function $\nabla^2 f$.
Since it does not use knowledge of the Hessian, the steepest descent algorithm is a first-order method.
The classical second-order method is *Newton's Method*.

The idea behind Newton's Method is to minimise at each iteration the *quadratic approximation* of f around the current iterate x^k.
This generates at each iteration a sub-problem that is "simpler", in the sense that we know the mathematical properties of the quadratic approximation.

Definition: (Newton direction)

Suppose f is C^2. Consider an iterate x^k such that $\nabla f(x^k) \neq 0$ and the Hessian $\nabla^2 f(x^k)$ is invertible. Then the *Newton direction* is given by

$$d^k \ := \ -\nabla^2 f(x^k)^{-1} \nabla f(x^k). \tag{3.8}$$

By choosing the direction d^k according as the Newton direction, we are choosing the direction that would get to the minimum in one step with a step length of $t_k = 1$ if the second order approximation held exactly, that is if f were a quadratic function.

For an arbitrary function in C^2, there is no guarantee that $\nabla^2 f(x^k)$ is invertible, nor is it always true that the Newton direction is a descent direction.

Therefore, at each iteration we must check both of these conditions. Specifically we must check

- that $\nabla^2 f(x^k)$ is invertible and,
- that the Newton direction $d^k = -\nabla^2 f(x^k)^{-1} \nabla f(x^k)$ (if it is well-defined) is a descent direction.

If both of these conditions are satisfied, we can then use a line search procedure to find a suitable step size $t_k > 0$, and finally set $x^{k+1} = x^k + t_k d^k$.

Lemma (Conditions for the Newton Direction)

If $\nabla f(x^k) \neq 0$ and $\nabla^2 f(x^k)$ is positive definite, then $\nabla^2 f(x^k)$ is invertible and the Newton direction is a descent direction for f at x^k.

Proof:

Because $\nabla^2 f(x^k)$ is positive definite and symmetric, it has all positive eigenvalues. Therefore the determinant, which is the product of the eigenvalues, is nonzero, so it is invertible. Furthermore, the eigenvalues of the inverse are the reciprocals of the eigenvalues of $\nabla^2 f(x^k)$, so the inverse is also positive definite.

$$\langle d^k, \nabla f(x^k) \rangle = \langle \nabla f(x^k), -\nabla^2 f(x^k)^{-1} \nabla f(x^k) \rangle < 0$$

from positive definiteness. So the Newton direction is a descent direction.

Algorithm for Newton's Method.

We now present an algorithms for Newton's Method for minimising a unimodal function $f : \mathbb{R}^n \to \mathbb{R}$ to within tolerance ϵ.

This follows the framework of the general descent method. At iteration k, the descent direction is $d^k = -\nabla^2 f(x^k)^{-1} \nabla f(x^k)$ if $\nabla^2 f(x^k)$ is positive definite, otherwise d^k is any descent direction for f at x^k, for example $-\nabla f(x^k)$. In either case, t_k is determined by the line search procedure.

Algorithm (Newton's Method)

1. Select $x^0 \in \mathbb{R}^n$.
   Set $k = 0$.
2. If $||\nabla f(x^k)|| < \epsilon$ then stop.
   If $\nabla^2 f(x^k)$ is positive definite, then
   Set $d^k = -\nabla^2 f(x^k)^{-1} \nabla f(x^k)$.
   Else, set $d^k = -\nabla f(x^k)$
3. Select step length t_k either
   - by solving the single-variable minimisation problem:
     $\min q(t) = f(x^k + td^k)$.
   - by using our procedure for finding a step length that satisfies the Armijo-Goldstein and Wolff conditions.
4. Set $k = k + 1$.
   Set $x^{k+1} = x^k + t_k d^k$.
   Return to step 2.

Newton's Method

Theorem (Effectiveness of Newton's Method)

Suppose f is C^2, $x^0 \in \mathbb{R}^n$, $0 < \sigma < 1/2 \leq \mu < 1$, and we implement Newton's Method as above with the step length chosen to satisfy the Armijo-Goldstein and Wolff conditions. If $\{x^k\}$ has a cluster point x^* such that $\nabla^2 f(x^*)$ is positive definite then

1. x^* is a local minimum of f.
2. For sufficiently large k, d^k is the Newton direction, $t_k = 1$ and $x^{k+1} = x^k + d^k$.
3. $x^k \to x^*$ superlinearly; indeed the rate of convergence is quadratic if f is C^3.

Newton's Method

Example 1:
If we apply Newton's method to the function $f(x) = \sin(x)$ to find the minimum $3\pi/2$, the Theorem shows that if the step length satisfies the Armijo-Goldstein and Wolff conditions, then since

$$\nabla^2 f(3\pi/2) = -\sin(3\pi/2) = 1 > 0,$$

we will find converge to the minimum at a quadratic rate.

Example 2:
On the other hand, if we apply Newton's method to the function $f(x) = x^4$, then as we get close to the minimum of 0, we cannot tell if $\nabla^2 f(x^k)$ is positive definite or not. This would depend on the thresholds we use for ∇f and $\nabla^2 f$. If we set them badly, we may have to revert to the steepest descent method.

Newton's Method

Class Exercise 3.11

Consider the problem of minimizing

$$f(x) = x_1^2 + x_2^2 - x_1 x_2 - 3x_1 + 3x_2 + 3.$$

Apply one step of the steepest descent method starting with the point $x^0 = (0, 0)$. Apply one step of Newton's Method under the same conditions. By finding all stationary points and checking second order conditions, find an exact solution.

Homework exercise, slide 225

$$f(x) = x_1^2 + x_2^2 - x_1x_2 - 3x_1 + 3x_2 + 3$$

$$\nabla f(x) = \begin{bmatrix} 2x_1 - x_2 - 3 \\ 2x_2 - x_1 + 3 \end{bmatrix}$$

$$\nabla^2 f(x) = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix} \rightsquigarrow \text{eigenvalues}$$

$$\begin{vmatrix} (2-\lambda) & -1 \\ -1 & (2-\lambda) \end{vmatrix} = 0 \Rightarrow \lambda^2 - 4\lambda + 3 = 0$$

$$\lambda = \frac{4 \pm \sqrt{16 - 4 \cdot 1 \cdot 3}}{2}$$

$$\lambda = \begin{cases} 3 \\ 1 \end{cases}$$

$$\left[\nabla^2 f(x)\right]^{-1} = \begin{bmatrix} 2/3 & 1/3 \\ 1/3 & 2/3 \end{bmatrix}$$

$$d_k = -\begin{bmatrix} 2/3 & 1/3 \\ 1/3 & 2/3 \end{bmatrix} \begin{bmatrix} -3 \\ 3 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$$x_{k+1} = (0,0)^T + t \cdot (1,-1)^T = \begin{bmatrix} t \\ -t \end{bmatrix}$$

$$f(x_{k+1}) = 3t^2 - 6t + 3$$

$$\nabla f(x_{k+1}) = 6t - 6 \Rightarrow t = 1$$

$$x_{k+1} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

Stationary points.

$$\begin{array}{l} (1) \\ (2) \end{array} \begin{bmatrix} 2x_1 - x_2 - 3 \\ 2x_2 - x_1 + 3 \end{bmatrix} = 0$$

$(2) \cdot 2 + (1)$

$3x_2 + 3 = 0 \Rightarrow x_2 = -1$

$\Rightarrow x_1 = 1$

since $\nabla^2 f(x)$ is ~~semi~~definite positive

this is a minimum

Class Exercise 3.12

Compute the Newton direction for the function

$$f(x) = x_1^2 x_2^2 + 2x_1^2 + 2x_2^2 - 4x_1 + 4x_2$$

using initial point $x^0 = (0, 0)$.

Homework Exercise: Using this direction and initial point x^0, apply Newton's Method to the problem of minimising $f(x)$.

Answers to Class Exercise 3.12

Newton's direction is $(1, -1)$.

Minimum via Newton's method is $x^1 = (0.77, -0.77)$.

Newton's Method

The main reason that Newton's Method converges faster than the steepest descent method, in some circumstances, is that it uses a better approximation of f to determine x^{k+1} from x^k.

Definition: An *affine* function g is a linear function plus a translation ($g(x) = L(x) + b$, where L is linear).

The Steepest Descent Method approximates f using the affine tangent function

$$f(x) \approx f(x^k) + \nabla f(x^k)^T (x - x^k),$$

which uses the first two terms of the Taylor series for f at x^k. The descent direction is the direction of steepest descent of the tangent plane, which is equivalent to the "instantaneous" direction of steepest descent of the original function f.

Newton's Method

In contrast, Newton's Method approximates f using the quadratic function

$$f(x) \approx Q_k(x) = f(x^k) + \nabla f(x^k)^T (x - x^k) + \frac{1}{2}(x - x^k)^T \nabla^2 f(x^k)(x - x^k)$$

which consists of the first three terms of the Taylor series. The next iterate is taken to be the vector x^{k+1} which corresponds to the unique minimum of $Q_k(x)$ (provided the Hessian of f is invertible at x^k).

Newton's Method

To find this, we solve $\nabla Q_k(x) = 0$:

$$0 = \nabla Q_k(x) = \nabla f(x^k) + \nabla^2 f(x^k)(x - x^k)$$

$$\nabla^2 f(x^k)(x - x^k) = -\nabla f(x^k)$$

$$x = x^k - \nabla^2 f(x^k)^{-1} \nabla f(x^k)$$

which is exactly the x^k plus the Newton direction! So taking step size $t_k = 1$ solves the approximation exactly.

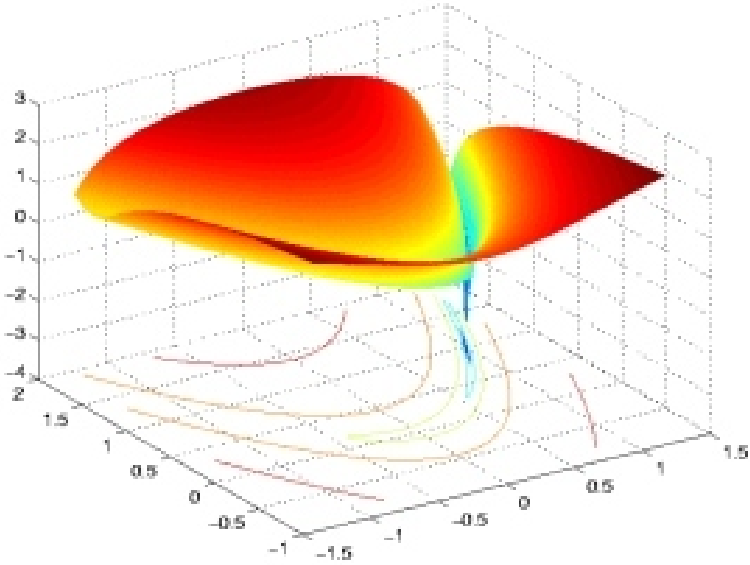
Newton's Method

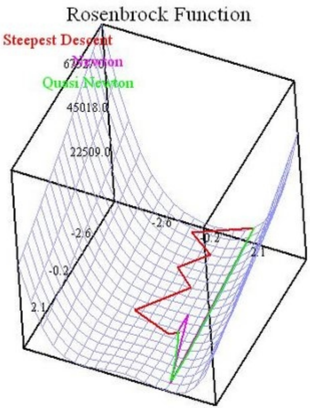
To illustrate the faster convergence of Newton's Method over steepest descent, consider the problem

$$\min f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2.$$

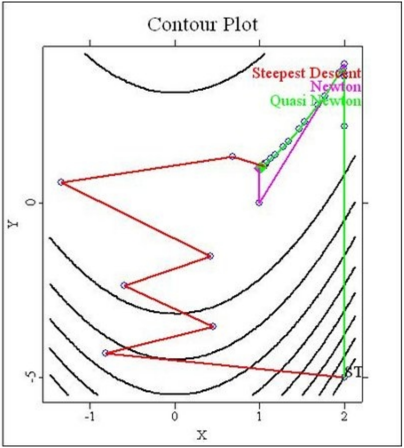
The function f above is known as the *Rosenbrock function*.

Typically Newton's Method takes substantially fewer steps on this problem than the steepest descent method: For some starting points it may only require 5 steps as opposed to the more than 12,000 required by steepest descent.


3
2
1
0
-1
-2
-3
-4
2
1.5
1
0.5
0
-0.5
-1
-1.5
-1
-0.5
0
0.5
1
1.5



Rosenbrock Function
Steepest Descent
Newton
Quasi Newton
45018.0
22509.0
-2.6
-0.2
2.1
-2.6
-0.2
2.1



Contour Plot
Steepest Descent
Newton
Quasi Newton
Y
0
-5
-1
0
1
2
X
ST

Subsubsection 4

Quasi-Newton methods

Quasi-Newton methods

There are two particular drawbacks of Newton's method.

1. We must be able to compute the Hessian matrix $\nabla^2 f(x^k)$ for each k. Even writing a mathematical expression of the Hessian matrix can be a major task when there are a large number of variables.

2. Determining the Newton direction requires solving a system of n linear equations in n unknowns, in particular finding d^k which solves
$$-\nabla^2 f(x^k)d^k = \nabla f(x^k). \tag{3.9}$$

Quasi-Newton methods

Using Gaussian elimination to calculate d^k requires around $n^3/3$ flops (floating point operations such as adding, subtracting, multiplying or dividing two numbers on a computer).

To give some perspective on the cost of Gaussian elimination, if we are given a problem with twice as many variables, $2n$ in all, then each time we determine d^k it takes $(2n)^3/3 = 8(n^3/3)$ flops, that is 8 times more work than before. In other words, the amount of work required per iteration of Newton's method increases dramatically as n increases.

Quasi-Newton methods

For these reasons, approximate Newton or quasi-Newton methods have become a very important theoretical and practical part of optimization.

The basic idea is that, rather than defining the Newton direction $d^k = -\nabla^2 f(x^k)^{-1} \nabla f(x^k)$, we construct a matrix $H_k \approx \nabla^2 f(x^k)^{-1}$ and let

$$d^k = -H_k \nabla f(x^k);$$

this vector d^k is called the *quasi-Newton* direction.

Quasi-Newton methods

Quasi-Newton methods can be relatively simple and cheap: $\nabla^2 f(x^k)$ is not needed, and the linear system $0 = \nabla f(x^k) + \nabla^2 f(x^k) d^k$ does not need to be solved. Note that the matrix-vector multiplication $H_k \nabla f(x^k)$ takes only around n^2 flops, and $n^2 \ll n^3/3$ for even moderate sized n.

Two of the most basic quasi-Newton methods are the DFP (Davidon-Fletcher-Powell) method and the BFGS (Broyden-Fletcher-Goldfarb-Shanno) method. The latter is generally accepted as the better method for optimisation, hence we shall concentrate on it.

Quasi-Newton methods - Motivation

The secant method, familiar from solving a single equation with 1 variable, motivates the choice of H_k.

Suppose we want to find the minimum of C^2 function $f : \mathbb{R} \to \mathbb{R}$. This is the same as finding the root of the C^1 function $g = f' : \mathbb{R} \to \mathbb{R}$. Instead of using Newton's method,

$$\begin{aligned} x^{k+1} &= x^k - \frac{f'(x^k)}{f''(x^k)} \\ &= x^k - \frac{g(x^k)}{g'(x^k)}, \end{aligned}$$

we approximate $f''(x^k) = g'(x^k)$ by the gradient of the secant on the graph of g which joins the $(x^{k-1}, g(x^{k-1}))$ to $(x^k, g(x^k))$.

The gradient of this secant is

$$\frac{g(x^k) - g(x^{k-1})}{x^k - x^{k-1}},$$

and the secant method calculates x^{k+1} as

$$\begin{aligned} x^{k+1} &= x^k - \frac{(x^k - x^{k-1})}{g(x^k) - g(x^{k-1})} g(x^k) \\ &= x^k - \frac{(x^k - x^{k-1})}{f'(x^k) - f'(x^{k-1})} f'(x^k) \end{aligned}$$

Putting it another way, the secant method sets

$$H_k \;=\; \left(\frac{f'(x^k) - f'(x^{k-1})}{x^k - x^{k-1}}\right)^{-1} \;=\; \frac{x^k - x^{k-1}}{f'(x^k) - f'(x^{k-1})}, \qquad (3.10)$$

and $x^{k+1} := x^k - H_k f'(x^k)$.

Quasi-Newton methods - Generalising to $\mathbb{R}^n$

In general, if Newton's Method for finding the minimum of f converges quadratically to x^*, the secant method converges superlinearly. So we hope, in higher dimensions ($n \geq 2$), that we can construct H_k in a relatively cheap way and still get superlinear convergence, without the cost of computing $\nabla^2 f(x^k)$ or solving an $n \times n$ linear system.

Quasi-Newton methods - Generalising to $\mathbb{R}^n$

For $n \geq 2$, H_k is usually chosen

1. to be symmetric, because $\nabla^2 f(x^k)$, hence $\nabla^2 f(x^k)^{-1}$ is symmetric;
2. to satisfy

$$H_{k+1}(\nabla f(x^{k+1}) - \nabla f(x^k)) = x^{k+1} - x^k; \tag{3.11}$$

3. to be positive definite.

The secant equation (3.11) is the n-dimensional analogue of the single-variable secant equation (3.10).

Quasi-Newton methods - Generalising to $\mathbb{R}^n$

Symmetricity and positive definiteness of H_k ensure that the quasi-Newton direction $d^k = -H_k \nabla f(x^k)$ is a descent direction. This follows from the following Lemma:

Lemma 7 (Conditions for Descent Direction)

Let B be a symmetric matrix in $\mathbb{R}^{n \times n}$. Then B is positive definite if and only if

- B is invertible and B^{-1} is positive definite.

If, in addition, $\nabla f(x) \neq 0$ then both $-B^{-1} \nabla f(x)$ and $-B \nabla f(x)$ are descent directions for f at x.

Proof:

(only if) First show B positive definite $\Rightarrow$ B is invertible.

Choose any $d \neq 0$.

Then, $d^T Bd > 0$ (since B is positive definite)

$\Rightarrow Bd \neq 0$

$\Rightarrow \text{nullspace}(B) = 0$ (since the above is true for all $d \neq 0$)

$\Rightarrow$ B is invertible.

Proof – Continued

Now show B positive definite $\Rightarrow$ B^{-1} is positive definite.

Again let $d \neq 0$. To show that B^{-1} is positive definite, we need to show that $d^T B^{-1} d > 0$. This follows since

$$\begin{aligned} d^T B^{-1} d &= d^T B^{-1} B B^{-1} d \\ &= \left((B^{-1})^T d \right)^T B \left(B^{-1} d \right) \\ &= \left(B^{-1} d \right)^T B \left(B^{-1} d \right) \\ &> 0 \end{aligned}$$

where the third equality follows because B is symmetric and the final inequality because B is positive definite.

Proof – Ccontinued

(if) A similar argument with B^{-1} and B interchanged can be used to show that if B is invertible and B^{-1} is positive definite then B is positive definite.

To show that $-B^{-1}\nabla f(x)$ is a descent direction, we need to show that $\nabla f(x)^T(-B^{-1}\nabla f(x)) < 0$. This follows immediately because B^{-1} is positive definite.

A similar argument shows that $-B\,\nabla f(x)$ is a descent direction.

Subsubsection 5

The BFGS Quasi-Newton Method

The BFGS (Broyden-Fletcher-Goldfarb-Shanno) Method

The BFGS method is the general descent method in which d^k is chosen as the BFGS direction, that is $d^k = -H_k \nabla f(x^k)$ where H_k is constructed using the BFGS update below.

Objective
Minimise a unimodal function $f : \mathbb{R}^n \to \mathbb{R}$ to within tolerance ϵ.

Algorithm (The BFGS Method)

1. Select $x^0 \in \mathbb{R}^n$.
   Set $k = 0$. Set $H_0 \in \mathbb{R}^{n \times n}$ to be a symmetric positive definite matrix (for example $H_0 = I$).
2. If $||\nabla f(x^k)|| < \epsilon$ then stop.
   Set $d^k = -H_k \nabla f(x^k)$.
3. Select step length t_k either
   - by solving the single-variable minimisation problem: $\min q(t) = f(x^k + td^k)$.
   - by using our procedure for finding a step length that satisfies the Armijo-Goldstein and Wolff conditions.
4. Set $x^{k+1} = x^k + t_k d^k$.
   Update H_k using the BFGS Update.
   Set $k = k + 1$.
   Return to step 2.

Algorithm - Part 2 (The BFGS Update)

$$s^k = x^{k+1} - x^k$$

$$g^k = \nabla f(x^{k+1}) - \nabla f(x^k)$$

$$r^k = H_k g^k / \langle s^k, g^k \rangle$$

$$H_{k+1} = H_k + \frac{1 + \langle r^k, g^k \rangle}{\langle s^k, g^k \rangle} s^k (s^k)^T - [s^k (r^k)^T + r^k (s^k)^T]$$

Note: *Outer products* such as $s^k(s^k)^T$ and $s^k(r^k)^T$ are actually $n \times n$ matrices, because they are each the product of an $n \times 1$ matrix (column vector) with a $1 \times n$ matrix (row vector).
For example, if

$$s^k = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \; r^k = \begin{bmatrix} -3 \\ 5 \end{bmatrix},$$

then

$$s^k(r^k)^T \quad = \quad \begin{bmatrix} 1 \\ 2 \end{bmatrix} [-3 \quad 5] \quad = \quad \begin{bmatrix} -3 & 5 \\ -6 & 10 \end{bmatrix}.$$

Where does the The BFGS Update formula come from? [Not covered in lectures]

That is a good question! To see where, we look briefly at two other quasi-Newton methods - the rank-one correction formula and the DFP algorithm.

All of these methods (rank-one, DFP, BFGS) are quasi-Newton methods and follow the same ideas. Their sole difference lies in choice of H_k. They also all ensure that

$$H_{k+1}(\nabla f(x^{k+1}) - \nabla f(x^k)) = x^{k+1} - x^k$$

as discussed above.

Furthermore, they all select H_k by updating from H_{k-1}. The difference lies in how they update it.

In the rank-one correction method, we try to update H_k as little as possible. To do this, we add a matrix of rank one to it:

$$H_{k+1} = H_k + z^k(z^k)^T.$$

After some calculation, which I will not give here, it turns out that ensuring $H_{k+1}(\nabla f(x^{k+1}) - \nabla f(x^k)) = x^{k+1} - x^k$ gives the formula

$$H_{k+1} = H_k + \frac{(s^k - H_k g^k)(s^k - H_k g^k)^T}{(g^k)^T(s^k - H_k g^k)}$$

using the notation given before.

So far so good; but this approach has problems. In particular, the updating scheme does not preserve positive definiteness. The DFP updating scheme, however, does.
The DFP scheme requires a rank 2 correction:

$$H_{k+1} = H_k + y^k (y^k)^T + z^k (z^k)^T.$$

To ensure that this works, we need

$$\begin{aligned} H_{k+1}(\nabla f(x^{k+1}) - \nabla f(x^k)) &= (H_k + y^k (y^k)^T + z^k (z^k)^T) g^k \\ &= x^{k+1} - x^k = s^k. \end{aligned}$$

Therefore

$$y^k(y^k)^T g^k + z^k(z^k)^T g^k = s^k - H_k g^k.$$

There are a few ways to make this happen; the simplest way (but not necessarily most effective!) is just to choose y^k and z^k so that the first terms on each side match, and the second terms on each side match. Then

$$y^k = \frac{1}{(y^k)^T g^k} s^k$$

$$y^k(y^k)^T = \frac{1}{((y^k)^T g^k)^2} s^k(s^k)^T$$

which is good, except for the (y^k) on the right-hand side.

To get rid of this we notice that

$$(g^k)^T y^k (y^k)^T g^k = [(y^k)^T g^k]^T (y^k)^T g^k = (g^k)^T s^k$$

and since $(y^k)^T g^k$ is a scalar, the left-hand side is $((y^k)^T g^k))^2$. So

$$y^k (y^k)^T = \frac{1}{\langle g^k, s^k \rangle} s^k (s^k)^T.$$

We can do the same thing for z^k.

$$z^k = -\frac{1}{(z^k)^T g^k} H_k g^k$$

$$z^k(z^k)^T = \frac{1}{((z^k)^T g^k)^2}(H_k g^k)(H_k g^k)^T$$

and again we have to get rid of z^k on the right-hand side.

$$(g^k)^T z^k (z^k)^T g^k = -(g^k)^T H_k g^k$$

and again, the left-hand side is $((z^k)^T g^k)^2$. So

$$z^k(z^k)^T = -\frac{1}{\langle g^k, H_k g^k\rangle}(H_k g^k)(H_k g^k)^T.$$

Putting it all together, we arrive at the DFP update scheme:

$$\begin{aligned} H_{k+1} &= H_k + y^k(y^k)^T + z^k(z^k)^T \\ &= H_k + \frac{1}{\langle g^k, s^k \rangle} s^k(s^k)^T - \frac{1}{\langle g^k, H_k g^k \rangle}(H_k g^k)(H_k g^k)^T. \end{aligned}$$

By inserting this formula into the general quasi-Newton framework, we get the DFP algorithm.

In fact not only does this satisfy the property

$$H_{k+1}(\nabla f(x^{k+1}) - \nabla f(x^k)) = x^{k+1} - x^k,$$

but it turns out that

$$H_{k+1}(\nabla f(x^{i+1}) - \nabla f(x^i)) = x^{i+1} - x^i$$

for any $0 \leq i \leq k$!

As if this wasn't complicated enough, some bright fellows decided to make it even more complicated!
Broyden, Fletcher, Goldfarb and Shanno noticed that the DFP algorithm updates the Hessian to preserve the relation

$$H_{k+1}g^k = s^k.$$

But, if we turn it around and have a sequence of matrices B_k which satisfy

$$B_{k+1}s^k = g^k,$$

then we can invert B_k to get H_k. Furthermore, we can actually use the formula we use to update H_k to update B_k if we simply swap g^k and s^k around!

This leads to the BFGS update system:

$$B_k = H_k^{-1}$$

$$B_{k+1} = B_k + \frac{1}{\langle s^k, g^k \rangle} g^k (g^k)^T - \frac{1}{\langle s^k, B_k s^k \rangle} (B_k s^k)(B_k s^k)^T$$

$$H_{k+1} = B_{k+1}^{-1}$$

This is a fairly horrendous mess by now, but it can be simplified somewhat by means of the following lemma.

Lemma 8:

Let A be a nonsingular matrix, and let u and v be column vectors such that $1 + v^T A^{-1} u \neq 0$. Then $A + uv^T$ is nonsingular, and

$$(A + uv^T)^{-1} = A^{-1} - \frac{(A^{-1}u)(v^T A^{-1})}{1 + v^T A^{-1} u}.$$

Proof: Exercise....

By using the lemma twice on B_{k+1}, we obtain the formula for H_{k+1} that we specified above.
It turns out that the BFGS direction also satisfies the property

$$H_{k+1}(\nabla f(x^{i+1}) - \nabla f(x^i)) = x^{i+1} - x^i$$

for any $0 \leq i \leq k$.

BFGS direction satisfies Equation (3.11)

$$
\begin{aligned}
& H_{k+1}(\nabla f(x^{k+1}) - \nabla f(x^k)) \\
& = H_{k+1} g^k \\
& = \left[H_k + \frac{1 + \langle r^k, g^k \rangle}{\langle s^k, g^k \rangle} s^k (s^k)^T - s^k (r^k)^T - r^k (s^k)^T \right] g^k \\
& = H_k g^k + \frac{1 + \langle r^k, g^k \rangle}{\langle s^k, g^k \rangle} s^k (s^k)^T g^k - s^k (r^k)^T g^k \\
& \quad - \frac{1}{\langle s^k, g^k \rangle} H_k g^k (s^k)^T g^k \\
& = H_k g^k + (1 + \langle r^k, g^k \rangle) s^k - \langle r^k, g^k \rangle s^k - H_k g^k \\
& = s^k \\
& = x^{k+1} - x^k.
\end{aligned}
$$

Properties of BFGS direction

It follows from Lemma 7, that if H_k is symmetric and positive definite and x^k is not stationary, then the quasi-Newton direction will be a descent direction.

It is straightforward to show that H_k is symmetric via induction. [Try as a homework exercise.]

The next Lemma says that, if we use the line search procedure satisfying the Armijo-Goldstein and Wolfe conditions to determine the step size t_k, the BFGS update of the quasi-Newton matrix H_k preserves positive definiteness.

Properties of BFGS direction

Lemma 9 (H_k is symmetric and positive definite)

Let H_0 be a symmetric positive definite matrix in $\mathbb{R}^{n\times n}$. If, for each k, $d^k := -H_k\nabla f(x^k)$ and the step size t_k satisfies the Wolfe condition (W), then H_k is symmetric and positive definite.

It follows that the BFGS method is a well defined descent method. It is also a 'good' algorithm, as shown in the next convergence result due to Powell (1976).

Properties of the BFGS method

Theorem (Convergence of BFGS method)

Let $H_0 \in \mathbb{R}^{n\times n}$ be symmetric positive definite, $x^0 \in \mathbb{R}^n$, $\sigma \in (0, 1/2)$, and $\mu \in (\sigma, 1)$. If f is convex and C^2, and the lower level set

$$\{x \in \mathbb{R}^n \,:\, f(x) \leq f(x^0)\}$$

is bounded, then the sequence produced by BFGS has a cluster point x^* which is a (local and) global minimum of f. If, in addition, $\nabla^2 f(x^*)$ is positive definite, then $x^k \to x^*$ superlinearly.

Class Exercise 3.13

Starting at the point $x^0 = (0, 0)$, perform a single iteration of the BGFS method to find a better approximation to the minimum for the function

$$f(x) = x_1^2 + x_2^2 - x_1 x_2 - 3x_1 + 3x_2 + 3.$$

Take $H_0 = \begin{bmatrix} 2/3 & 0 \\ 0 & 1/3 \end{bmatrix}$ as the initial matrix for computing the quasi-Newton direction.

You should compute x^1, as well as s^0, g^0, r^0 and H_1 (in preparation for a second iteration).

Extra Homework Exercise: The true minimum occurs at $(1, -1)$. Perform a second iteration and show that the true minimum is reached in exactly 2 iterations.

Class exercise. 267

$f(x) = x_1^2 + x_2^2 - x_1 x_2 - 3x_1 + 3x_2 + 3$

$\nabla f(x) = \begin{bmatrix} 2x_1 - x_2 - 3 \\ 2x_2 - x_1 + 3 \end{bmatrix}$ $\qquad H_0 = \begin{bmatrix} 2/3 & 0 \\ 0 & 1/3 \end{bmatrix}$

$\boxed{1}$ $x_0 = (0,0) \qquad \varepsilon = 0.1$

$H_0 = \begin{bmatrix} 2/3 & 0 \\ 0 & 1/3 \end{bmatrix}$

$\nabla f(x_0) = \begin{bmatrix} -3 \\ 3 \end{bmatrix}, \quad \|\nabla f(x_0)\| > 0.1$

$\boxed{2}$ $d^0 = -H^0 \cdot \nabla f(x_0) = \begin{bmatrix} 2/3 & 0 \\ 0 & 1/3 \end{bmatrix} \begin{bmatrix} -3 \\ 3 \end{bmatrix} = \begin{bmatrix} +2 \\ -1 \end{bmatrix}$

$\boxed{3}$ $f(x_{k+1}) = f\left(\binom{0}{0} + t\binom{+2}{-1}\right) = f(+2t, -t)$

$= 4t^2 + t^2 + 2t^2 - 6t - 3t + 3$

$= 7t^2 - 9t + 3$

$f'(t) = 14t - 9 \Rightarrow \quad t^* = 9/14$

$\boxed{4}$ $x_1 =$

$$x_{k+1} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} + \frac{9}{14} \cdot \begin{pmatrix} 2 \\ -1 \end{pmatrix} = \begin{pmatrix} 18/14 \\ -9/14 \end{pmatrix}$$

$$s_0 = x_1 - x_0 = \begin{pmatrix} 18/14 \\ -9/14 \end{pmatrix} - \begin{pmatrix} 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 18/14 \\ -9/14 \end{pmatrix}$$

$$g_0 = \nabla f(x^1) - \nabla f(x^0) = \begin{pmatrix} -3 \\ 3 \end{pmatrix} + \begin{pmatrix} 2 \cdot 18/14 + 9/14 - 3 \\ -2 \cdot 9/14 - 18/14 + 3 \end{pmatrix} = \begin{pmatrix} 45/14 \\ -36/14 \end{pmatrix}$$

$$r_0 = H_0 \cdot g^0 / \langle s^0, g^0 \rangle$$

$$r_0 = \begin{bmatrix} 2/3 & 0 \\ 0 & 1/3 \end{bmatrix} \begin{bmatrix} 45/14 \\ -36/14 \end{bmatrix} \Big/ (18 \cdot 45 + 9 \cdot 36)/14 \cdot 14 = \begin{bmatrix} 90/81 \cdot 3 \\ -36/81 \cdot 3 \end{bmatrix} =$$

$$H_1 = H_0 + 1 + \langle r^0, g^0 \rangle / \langle s^0, g^0 \rangle \cdot (s^k)(s^k)^T - [s^k (r^k)^T + r^k (s^k)^T]$$

$$H_1 = \begin{bmatrix} 2/3 & 0 \\ 0 & 1/3 \end{bmatrix} + \frac{1 + \begin{bmatrix} 90/243 & -36/243 \end{bmatrix} \begin{bmatrix} 45/14 \\ -36/14 \end{bmatrix}}{\begin{bmatrix} 18/14 & -9/14 \end{bmatrix} \begin{bmatrix} 45/14 \\ -36/14 \end{bmatrix}} \cdot \begin{pmatrix} 18/14 \\ -9/14 \end{pmatrix} \begin{pmatrix} 18/14 & -9/14 \end{pmatrix}$$

$$- \left[\begin{pmatrix} 18/14 \\ -9/14 \end{pmatrix} \begin{pmatrix} 90/243 & -36/243 \end{pmatrix} + \begin{pmatrix} 90/243 \\ -36/243 \end{pmatrix} \begin{pmatrix} 18/14 & -9/14 \end{pmatrix} \right]$$

0.44

$$H_1 = \begin{bmatrix} 2/3 & 0 \\ 0 & 1/3 \end{bmatrix} + \frac{1 + 1.5714}{5.7857} \cdot \begin{bmatrix} 1.6530 & -0.8265 \\ -0.8265 & +0.4133 \end{bmatrix} - \left[\begin{bmatrix} 0.4762 & -0.1905 \\ -0.2381 & 0.0952 \end{bmatrix} + \begin{bmatrix} 0.4762 & -0.2381 \\ -0.1905 & 0.0952 \end{bmatrix} \right]$$

$$H_1 = \begin{bmatrix} 0.4490 & 0.0612 \\ 0.0612 & 0.3265 \end{bmatrix}$$

Subsection 4

Other methods

Non-derivative methods

The algorithms discussed so far for finding unconstrained minima of functions of n variables require Hessian and/or gradient calculations:

- The Steepest Descent Method requires the calculation of $\nabla f(x)$
- Newton's Method requires the calculation of $\nabla f(x)$ and $\nabla^2 f(x)$
- Quasi-Newton Methods require the calculation of $\nabla f(x)$

Recall the line-search methods for finding minima of functions of one variable. The Fibonacci Search and Golden Section search algorithms did not require any derivative calculations. Are there non-derivative methods for finding minima of functions of n variables ? *Yes*:

Non-derivative methods

- The Nelder-Mead "Simplex Method" (not to be confused with Simplex Method for solving LPs)
- The coordinate descent method
- Methods which approximate derivatives using finite difference formulae

Stochastic optimisation

If f is unimodal, then descent methods are generally able to find the (unique) minimum of f.

If f has multiple minima, for example, some local minima and an absolute minimum, then descent methods will find *one* of these for each application of the method (we may get different minima with different initial iterates x^0).

Usually, we are interested in attaining global minima, and we wish to avoid getting "stuck" in local minima. One approach is to try a range of different initial iterates x^0.

Alternative: stochastic search methods, which have some positive probability of searching in a non-descent direction. A classic example is the method of Simulated Annealing.

Subsection 5

Summary of methods

Summary of methods

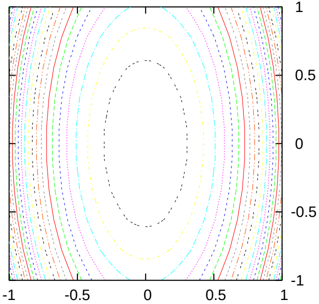
We have discussed 3 algorithms for finding unconstrained minima of functions of n variables.

- The Steepest Descent Method requires the calculation of $\nabla f(x)$
- Newton's Method requires the calculation of $\nabla f(x)$ and $\nabla^2 f(x)$
- Quasi-Newton Methods require the calculation of $\nabla f(x)$

Comparison of n-dimensional methods

$$f(x) = e^{x_1^2 + \frac{1}{4}x_2^2}$$

By inspection, the minimum lies at $(x_1, x_2) = (0, 0)$.



First we find the minimum analytically. We find the stationary points first.

$$\nabla f(x) = (2x_1 e^{x_1^2 + \frac{1}{4}x_2^2}, \frac{1}{2}x_2 e^{x_1^2 + \frac{1}{4}x_2^2})$$

When $\nabla f(x) = (0, 0)$, we have

$$2x_1 e^{x_1^2 + \frac{1}{4}x_2^2} = 0 \qquad \Rightarrow \qquad x_1 = 0$$

and

$$\frac{1}{2}x_2 e^{x_1^2 + \frac{1}{4}x_2^2} = 0 \qquad \Rightarrow \qquad x_2 = 0.$$

The Hessian is

$$\nabla^2 f(x) = \begin{bmatrix} 2e^{x_1^2+\frac{1}{4}x_2^2} + 4x_1^2 e^{x_1^2+\frac{1}{4}x_2^2} & x_1 x_2 e^{x_1^2+\frac{1}{4}x_2^2} \\ x_1 x_2 e^{x_1^2+\frac{1}{4}x_2^2} & \frac{1}{2}e^{x_1^2+\frac{1}{4}x_2^2} + \frac{1}{4}x_2^2 e^{x_1^2+\frac{1}{4}x_2^2} \end{bmatrix}$$

and at $(x_1, x_2) = (0, 0)$, this gives

$$\nabla^2 f(0,0) = \begin{bmatrix} 2 & 0 \\ 0 & \frac{1}{2} \end{bmatrix}$$

which is definitely positive definite (it has all positive eigenvalues). Therefore $(0, 0)$ is a minimum of $f(x)$.

Line searching - Formula for Step Size

Suppose that we are at the point $x = (x_1, x_2)$ and have chosen the descent direction $d = (d_1, d_2)$ to search in. Then we select the step size as follows.

$$f(x + td) = e^{(x_1+td_1)^2+\frac{1}{4}(x_2+td_2)^2}$$

$$\frac{d}{dt}f(x+td) = e^{(x_1+td_1)^2+\frac{1}{4}(x_2+td_2)^2}\left[2(x_1 + td_1)d_1 + \frac{1}{2}(x_2 + td_2)d_2\right]$$

The $e^{\cdots}$ term is strictly positive. So when $\frac{d}{dt}f(x + td) = 0$, we get

$$2(x_1 + td_1)d_1 + \frac{1}{2}(x_2 + td_2)d_2 = 0$$

$$(2d_1^2 + \frac{1}{2}d_2^2)t = -2x_1 d_1 - \frac{1}{2}x_2 d_2$$

$$t = -\frac{2x_1 d_1 + \frac{1}{2}x_2 d_2}{2d_1^2 + \frac{1}{2}d_2^2}$$

We can use this formula to find the step size for any of the algorithms.

Method 1: Steepest Descent

Initial Values:

$$x^0 = (1, 1)$$

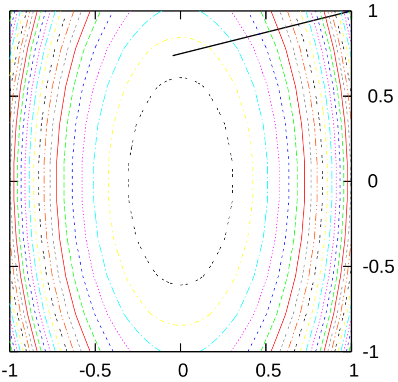
$$\nabla f(x^0) = (6.98, 1.75), \|\nabla f(x^0)\| = 7.2 > 0.001$$

Iteration 1:

$$d^0 = (-6.98, -1.75), t = 0.15$$

$$x^1 = (1, 1) + 0.15(-6.98, -1.75) = (-0.0462, 0.738)$$

$$\nabla f(x^1) = (-0.106, 0.424), \|\nabla f(x^1)\| = 0.437 > 0.001$$


1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1

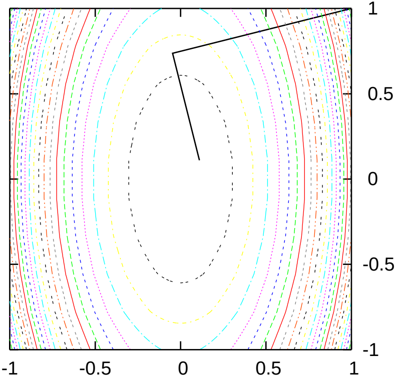
Method 1: Steepest Descent – Continued

Iteration 2:

$$d^1 = (0.106, -0.424), t = 1.48$$

$$x^2 = (-0.0462, 0.738) + 1.48(0.106, -0.424) = (0.111, 0.111)$$

$$\nabla f(x^2) = (0.225, 0.0562), \|\nabla f(x^2)\| = 0.232 > 0.001$$


1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1

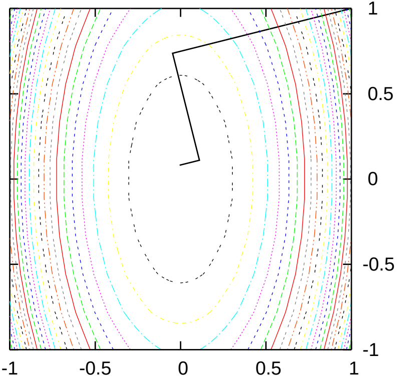
Method 1: Steepest Descent – Continued

Iteration 3:

$$d^2 = (-0.225, -0.0562), t = 0.515$$

$$x^3 = (0.111, 0.111) + 0.515(-0.225, -0.0562) = (-0.00511, 0.0818)$$

$$\nabla f(x^3) = (-0.0102, 0.041), \|\nabla f(x^3)\| = 0.0422 > 0.001$$


1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1

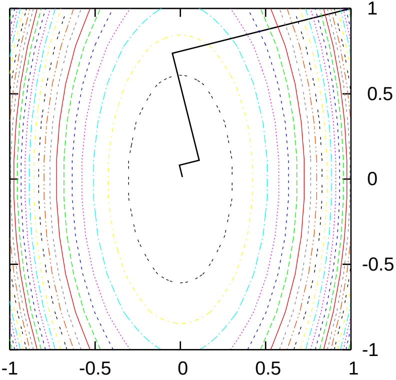
Method 1: Steepest Descent – Continued

Iteration 4:

$$d^3 = (0.0102, -0.041), t = 1.7$$

$$x^4 = (-0.00511, 0.0818) + 1.7(0.0102, -0.041) = (0.0123, 0.0123)$$

$$\nabla f(x^4) = (0.0245, 0.00614), \|\nabla f(x^4)\| = 0.0253 > 0.001$$



1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1

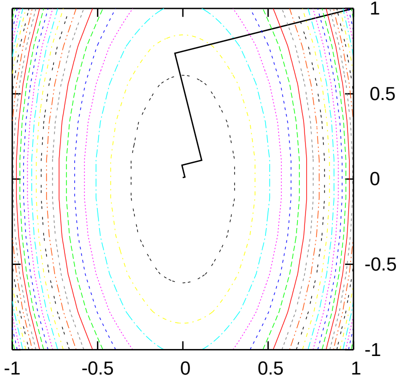
Method 1: Steepest Descent – Continued

Iteration 5:

$$d^4 = (-0.0245, -0.00614), t = 0.523$$

$$\begin{aligned} x^5 &= (0.0123, 0.0123) + 0.523(-0.0245, -0.00614) \\ &= (-0.000566, 0.00906) \end{aligned}$$

$$\nabla f(x^5) = (-0.00113, 0.00453), \|\nabla f(x^5)\| = 0.00467 > 0.001$$


1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1

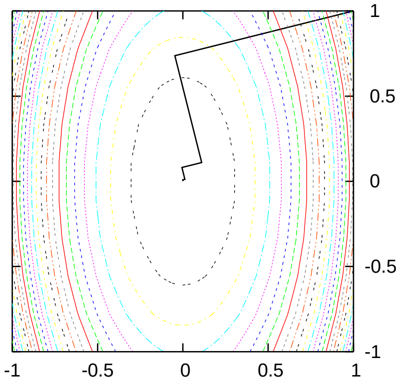
Method 1: Steepest Descent – Continued

Iteration 6:

$$d^5 = (0.00113, -0.00453), t = 1.7$$

$$\begin{aligned} x^6 &= (-0.000566, 0.00906) + 1.7(0.00113, -0.00453) \\ &= (0.00136, 0.00136) \end{aligned}$$

$$\nabla f(x^6) = (0.00272, 0.00068), \|\nabla f(x^6)\| = 0.0028 > 0.001$$


1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1

Method 1: Steepest Descent – Continued

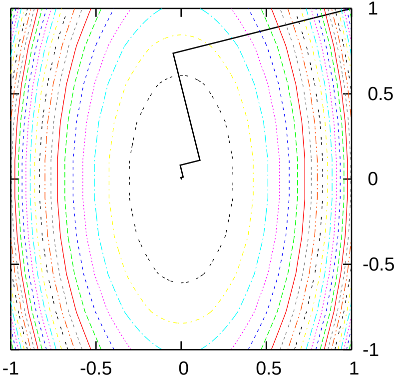
Iteration 7:

$$d^6 = (-0.00272, -0.00068), t = 0.523$$

$$\begin{aligned} x^7 &= (0.00136, 0.00136) + 0.523(-0.00272, -0.00068) \\ &= (-0.0000627, 0.001) \end{aligned}$$

$$\nabla f(x^7) = (-0.000125, 0.000502), \|g(x^7)\| = 0.000517 < 0.001$$

Final estimate is $(-0.0000627, 0.001)$. We used 8 calculations (7 iterations).


1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1

Method 2: Newton's method

Initial Values:

$$x^0 = (1, 1)$$

$$\nabla f(x^0) = (6.98, 1.75), \|\nabla f(x^0)\| = 7.2 > 0.001$$

$$\nabla^2 f(x^0) = \begin{bmatrix} 20.9 & 3.49 \\ 3.49 & 2.62 \end{bmatrix}$$

Method 2: Newton's method

Iteration 1:

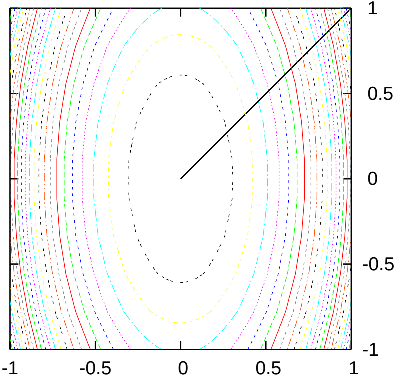
$$d^0 = -\left[\begin{array}{cc} 20.9 & 3.49 \\ 3.49 & 2.62 \end{array}\right]^{-1}\left[\begin{array}{c} 6.98 \\ 1.75 \end{array}\right] = (-0.286, -0.286)$$

$$t = 3.5$$

$$x^1 = (1,1) + 3.5(-0.286, -0.286) = (-2.22 \cdot 10^{-16}, 0)$$

$$\nabla f(x^1) = (-4.44 \cdot 10^{-16}, 0), \|g(x^1)\| = 4.44 \cdot 10^{-16} < 0.001$$

Final estimate is $(-2.22 \cdot 10^{-16}, 0)$. We used 3 calculations but only 1 iteration (Newton's method was too good for the function!).


1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1

Method 3: BFGS method

Initial Values:

$$x^0 = (1, 1)$$

$$H_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\nabla f(x^0) = (6.98, 1.75), \|\nabla f(x^0)\| = 7.2 > 0.001$$

Method 3: BFGS method – Continued

Iteration 1:

$$d^0 = -\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 6.98 \\ 1.75 \end{bmatrix} = (-6.98, -1.75)$$

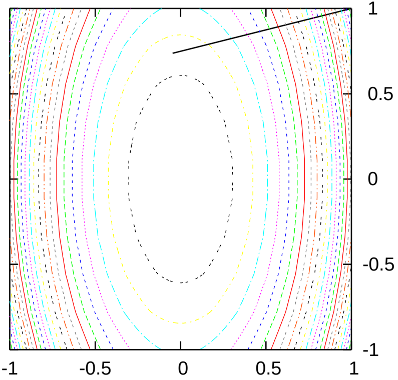
$$t = 0.15$$

$$x^1 = (1,1) + 0.15(-6.98, -1.75) = (-0.0462, 0.738)$$

$$s^0 = (-1.05, -0.262), \quad g^0 = (-7.09, -1.32)$$

$$r^0 = (-0.913, -0.17), \quad H_1 = \begin{bmatrix} 0.175 & -0.146 \\ -0.146 & 0.979 \end{bmatrix}$$

$$\nabla f(x^1) = (-0.106, 0.424), \quad \|\nabla f(x^1)\| = 0.437 > 0.001$$


1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1

Method 3: BFGS method – Continued

Iteration 2:

$$d^1 = -\begin{bmatrix} 0.175 & -0.146 \\ -0.146 & 0.979 \end{bmatrix} \begin{bmatrix} -0.106 \\ 0.424 \end{bmatrix} = (0.0803, -0.431)$$

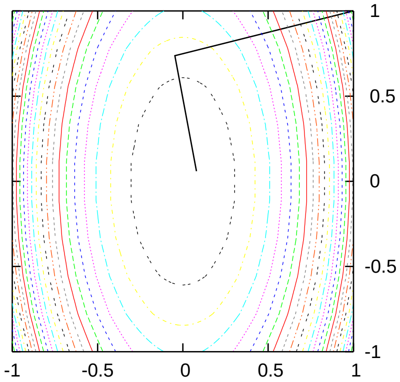
$$t = 1.58$$

$$x^2 = (-0.0462, 0.738) + 1.58(0.0803, -0.431) = (0.0803, 0.0599)$$

$$s^1 = (0.126, -0.679), \quad g^1 = (0.268, -0.394)$$

$$r^1 = (0.346, -1.41), \quad H_2 = \begin{bmatrix} 0.175 & -0.202 \\ -0.202 & 1.59 \end{bmatrix}$$

$$\nabla f(x^2) = (0.162, 0.0302), \quad \|\nabla f(x^2)\| = 0.165 > 0.001$$


1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1

Method 3: BFGS method – Continued

Iteration 3:

$$d^2 = -\begin{bmatrix} 0.175 & -0.202 \\ -0.202 & 1.59 \end{bmatrix} \begin{bmatrix} 0.162 \\ 0.0302 \end{bmatrix} = (-0.0222, -0.0151)$$

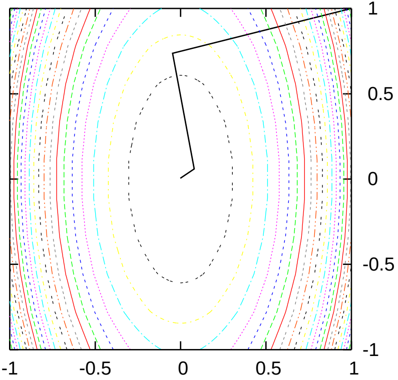
$$t = 3.66$$

$$x^3 = (0.0803, 0.0599) + 3.66(-0.0222, -0.0151) = (-0.000803, 0.00472)$$

$$s^2 = (-0.0811, -0.0552), \quad g^2 = (-0.163, -0.0278)$$

$$r^2 = (-1.55, -0.745), \quad H_3 = \begin{bmatrix} 0.49 & 0.0373 \\ 0.0373 & 1.77 \end{bmatrix}$$

$$\nabla f(x^3) = (-0.00161, 0.00236), \quad \|\nabla f(x^3)\| = 0.00286 > 0.001$$


1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1

Method 3: BFGS method – Continued

Iteration 4:

$$d^3 = -\begin{bmatrix} 0.49 & 0.0373 \\ 0.0373 & 1.77 \end{bmatrix} \begin{bmatrix} -0.00161 \\ 0.00236 \end{bmatrix} = (0.000699, -0.00411)$$

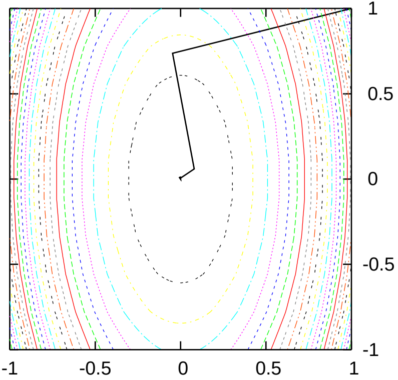
$$t = 1.15$$

$$x^4 = (-0.000803, 0.00472) + 1.15(0.000699, -0.00411)$$

$$= (5.03 \cdot 10^{-7}, 3.42 \cdot 10^{-7})$$

$$\nabla f(x^4) = (1.01 \cdot 10^{-6}, 1.71 \cdot 10^{-7}), \|g(x^4)\| = 1.02 \cdot 10^{-06} < 0.001$$

Final estimate is $(5.03 \cdot 10^{-7}, 3.42 \cdot 10^{-7})$. We used 5 calculations.


1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1

Comparison

Method	*Solution*	*Normed error*	*Calcs*
Exact	$(0, 0)$		
Steepest descent	$(-0.0000627, 0.001)$	0.00101	8
Newton	$(-2.22 \cdot 10^{-16}, 0)$	$2.22 \cdot 10^{-16}$	3
BFGS	$(5.03 \cdot 10^{-7}, 3.42 \cdot 10^{-7})$	$6.08 \cdot 10^{-7}$	5

This shows that:

- Newton's method is the most efficient, for this problem.
- Steepest descent is the worst, while BFGS lies somewhere in between.
- However Newton's method requires the Hessian, and must invert it as well.
- The BFGS does not - however it also has a lot of calculation! But the computational time used for this calculation is small.

Section 4

General Constrained Optimisation

- Optimality conditions
- Minimality conditions
- Lagrange multipliers and sensitivity analysis
- KKT Conditions
- Constraint qualifications
- Sufficient optimality conditions
- Penalty methods for Constrained Optimisation
- Log Barrier penalty methods
- Exact penalty methods
- Comparison of penalty methods

Nonlinear Constrained Optimisation

We define a *nonlinear program (NLP)* as a problem of the form:

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{subject to} \quad & g(x) \le 0 \\ & h(x) = 0, \end{aligned} \tag{NLP}$$

where the vector inequality $g(x) \le 0 \in \mathbb{R}^p$ means $g_i(x) \le 0$ for each $i = 1, \ldots, p$, and the vector equality $h(x) = 0 \in \mathbb{R}^q$ means $h_j(x) = 0$ for each $j = 1, \ldots, q$.

Nonlinear Constrained Optimisation

- If $p = 0$, (NLP) is an equality-constrained problem.
- If $p = q = 0$, (NLP) is unconstrained and the first-order necessary condition for x^* to minimise f would be that $\nabla f(x^*) = 0$.

We will generalise this stationarity condition to obtain first-order necessary conditions for (NLP) to minimise f. First we consider the easier case of nonlinear programs with only equality constraints.

Subsection 1

Optimality conditions

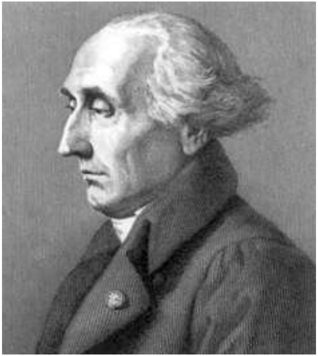
Optimality conditions for equality-constrained optimisation

The equality-constrained NLP is

$$\begin{aligned} \min_x \quad & f(x) \\ \text{subject to} \quad & h(x) = 0. \end{aligned} \tag{4.1}$$

This is just (NLP) with no inequality constraints, $p = 0$.

The mathematician Lagrange gave the first-order necessary conditions for this problem.



Optimality conditions for equality-constrained optimisation

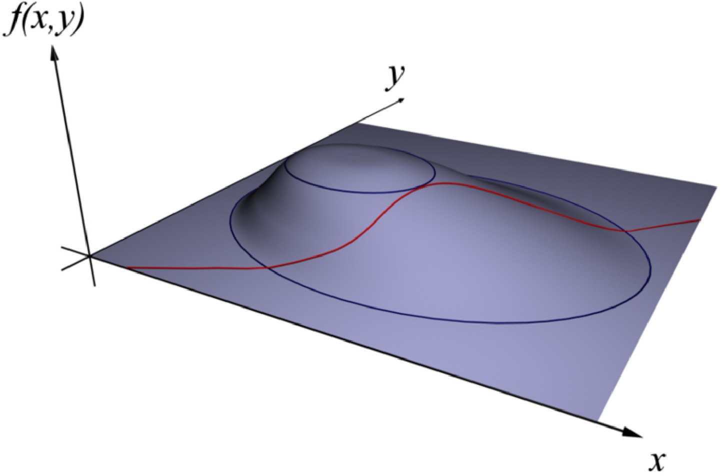
The *Lagrangian* function for (4.1) is

$$L(x,\eta) \ := \ f(x) + \sum_{j=1}^{q} \eta_j h_j(x) \ = \ f(x) + \langle \eta, h(x) \rangle,$$

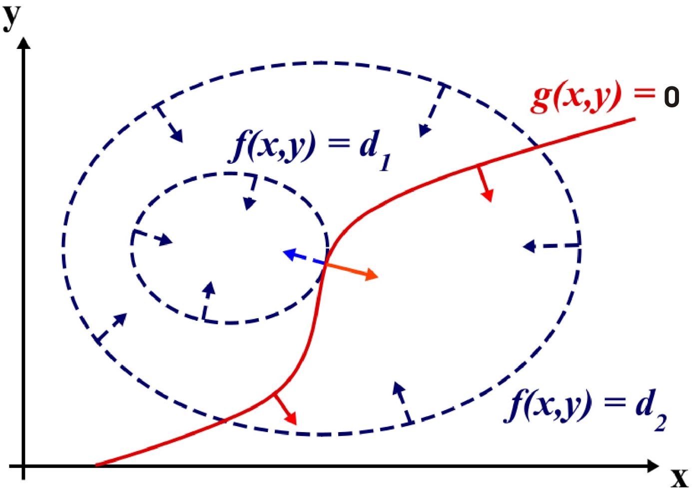
where $\eta \in \mathbb{R}^q$.

Definition:
The vector η is called the *Lagrange multiplier* corresponding to $h(x)$.
Each component η_j of η is a *multiplier* corresponding to each component $h_j(x)$ of $h(x)$.

Example: (2 variables, 1 constraint)



Example: (2 variables, 1 constraint)



The Lagrange method

Constraint Qualifications

Suppose that either

- h is an affine (linear plus constant) function, or
- the set of gradients $\{\nabla h_j(x) : \text{all } j\}$ is linearly independent.

Each of these conditions is called a *constraint qualification*, and ensures that h is not "badly behaved".

The first-order necessary conditions for x^* to be a local minimum of (4.1) are given in the following theorem.

The Lagrange method

Theorem 4 (Condition on the Lagrangian for minimality)

Let f and h be C^1 functions and assume that either of the above constraint qualifications hold at x^*. If x^* is a local minimum of (4.1) then there exists $\eta^* \in \mathbb{R}^q$ such that

$$\begin{aligned} 0 &= \nabla_x L(x^*, \eta^*) \\ 0 &= h(x^*). \end{aligned}$$

The vector η^* is called an *optimal* (Lagrange) multiplier for x^*. The first equation reflects the fact that the Lagrangian must be stationary with respect to x, the second equation requires x^* to be feasible.

The Lagrange method

Observe that the gradient of L is

$$\nabla_x L(x^*, \eta^*) \;=\; \nabla f(x^*) + \sum_j \eta_j^* \nabla h_j(x^*) \;=\; \nabla f(x^*) + \nabla h(x^*)\eta^*,$$

where the matrix $\nabla h(x^*)$, known as the *Jacobian*, is an $n \times q$ matrix.
The fact that local optima are stationary points of the Lagrange function with respect to x in the case of a single equality constraint can be seen by looking at the level curves of f and the feasible space.

The Lagrange method

At the optimum, the level curves must be parallel to the feasible region (if not, then it is possible to travel inside the feasible region in a way which decreases f). Therefore their normals must be parallel:

$$\nabla f(x^*) \propto \nabla h(x^*)$$

$$\nabla f(x^*) = -\eta^* \nabla h(x^*)$$

$$\nabla f(x^*) + \eta^* \nabla h(x^*) = 0$$

(remembering that the gradient is taken with respect to the x's only).
This is the *Lagrange condition*.

Summary – Equality-constrained optimisation

The equality-constrained NLP is

$$\begin{array}{ll} \min_x & f(x) \\ \text{subject to} & h(x) = 0. \end{array} \tag{4.2}$$

The *Lagrangian* function for (4.2) is

$$L(x, \eta) \;:=\; f(x) + \sum_{j=1}^{q} \eta_j h_j(x) \;=\; f(x) + \langle \eta, h(x) \rangle,$$

where $\eta \in \mathbb{R}^q$.

The Constraint Qualification are that either

- h is an affine (linear plus constant) function, or
- the set of gradients $\{\nabla h_j(x) : \text{all } j\}$ is linearly independent.

Summary – Equality-constrained optimisation

The first-order necessary conditions for x^* to be a local minimum of (4.2) are given in the following theorem.

Theorem 4 (Condition on the Lagrangian for minimality)

Let f and h be C^1 functions and assume that either of the above constraint qualifications hold at x^*. If x^* is a local minimum of (4.2) then there exists $\eta^* \in \mathbb{R}^q$ such that

$$
\begin{aligned}
0 &= \nabla_x L(x^*, \eta^*) \\
0 &= h(x^*).
\end{aligned}
$$

Observe that the gradient of L is

$$\nabla_x L(x^*, \eta^*) \;=\; \nabla f(x^*) + \sum_j \eta_j^* \nabla h_j(x^*) \;=\; \nabla f(x^*) + \nabla h(x^*)\eta^*.$$

Class Exercise 4.1

Solve the equality-constrained NLP

$$
\begin{array}{lllllllll}
\min & f(x) = & 2x_1^2 + 2x_2^2 + 4x_1x_2 + x_1x_3 + x_2x_3 & & & & & & \\
\text{s.t.} & h_1(x) = & x_1^2 & + & x_2^2 & & & -1 & = 0 \\
 & h_2(x) = & 2x_1 & + & 2x_2 & + & x_3 & -1 & = 0.
\end{array}
$$

Note: This NLP is not too hard to solve by inspection: by substituting $x_3 = 1 - 2(x_1 + x_2)$, graphing the resulting 2-variable feasible region and lines of constant objective function, and observing the symmetry of the problem, it can be seen that there is a unique optimal solution given by $x^* = \left(-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, 1 + 2\sqrt{2}\right)$.

Solution to Exercise 4.1 [Part 1]

We now seek a solution by examining the Lagrange function:

$$\begin{aligned} L(x,\eta) &= 2x_1^2 + 2x_2^2 + 4x_1x_2 + x_1x_3 + x_2x_3 \\ &\quad + \eta_1(x_1^2 + x_2^2 - 1) + \eta_2(2x_1 + 2x_2 + x_3 - 1). \end{aligned}$$

Assume that one of the constraint qualifications holds, (we will check this later). By Theorem 4, if x is a local optimum, it must be that $h_1(x) = h_2(x) = 0$ and there must exist η such that

$$\nabla_x L(x,\eta) = \begin{bmatrix} 4x_1 + 4x_2 + x_3 + 2\eta_1 x_1 + 2\eta_2 \\ 4x_2 + 4x_1 + x_3 + 2\eta_1 x_2 + 2\eta_2 \\ x_1 + x_2 + \eta_2 \end{bmatrix} = 0.$$

Solution to Exercise 4.1 [Part 1] – Continued

From the first two rows, we see that

$4x_1 + 4x_2 + x_3 + 2\eta_1 x_1 + 2\eta_2 = 0 = 4x_2 + 4x_1 + x_3 + 2\eta_1 x_2 + 2\eta_2.$

This tells us that $x_1 = x_2$.

Substituting $x_1 = x_2$ into $h_1(x) = x_1^2 + x_2^2 - 1 = 0$ we get $x_2 = \pm\frac{1}{\sqrt{2}}$, and hence $x_1 = \pm\frac{1}{\sqrt{2}}$.

Substituting these values into $h_2(x) = 2x_1 + 2x_2 + x_3 - 1 = 0$ yields $x_3 = 1 \mp 2\sqrt{2}$.

Solution to Exercise 4.1 [Part 1] – Continued

To find the optimal Lagrange multipliers corresponding to the two points found, we use the partial derivative of the Lagrangian with respect to x_3:

$$x_1 + x_2 + \eta_2 = 0$$

to give $\eta_2 = -(x_1 + x_2) = \mp\sqrt{2}$

We then use the partial derivative of the Lagrangian with respect to x_1:

$$4x_1 + 4x_2 + x_3 + 2\eta_1 x_1 + 2\eta_2 = 0$$

to give $\eta_1 = -\frac{1}{2x_1}(4x_1 + 4x_2 + x_3) - \frac{\eta_2}{x_1} = \mp\frac{1}{\sqrt{2}}$.

Solution to Exercise 4.1 [Part 1] – Continued

To summarise, the NLP has two stationary points:

$$x^* = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \\ 1 - 2\sqrt{2} \end{bmatrix} \quad \text{and} \quad x^* = \begin{bmatrix} -\frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \\ 1 + 2\sqrt{2} \end{bmatrix}.$$

The Lagrange multipliers for $x^* = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 1 - 2\sqrt{2}\right)$ are $\eta^* = \left(-\frac{1}{\sqrt{2}}, -\sqrt{2}\right)$, and the Lagrange multipliers for $x^* = \left(-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, 1 + 2\sqrt{2}\right)$ are $\eta^* = \left(\frac{1}{\sqrt{2}}, \sqrt{2}\right)$.

Solution to Exercise 4.1 [Part 1] – Continued

We now check the constraint qualifications at each of these points (to confirm that we could apply Theorem 4).
In this example, h is *not* affine, (h_1 in particular is not affine), so the first condition does not hold. Hence we look at the constraint gradients at each point. We have

$$\nabla h_1(x) = \begin{bmatrix} 2x_1 \\ 2x_2 \\ 0 \end{bmatrix} \quad \text{and} \quad \nabla h_2(x) = \begin{bmatrix} 2 \\ 2 \\ 1 \end{bmatrix}.$$

and the second constraint qualification is satisfied if these vectors are linearly independent, ie, if the matrix

$$\nabla h(x) = \begin{bmatrix} 2x_1 & 2 \\ 2x_2 & 2 \\ 0 & 1 \end{bmatrix}$$

has full column rank.

Solution to Exercise 4.1 [Part 1] – Continued

Now

$$\nabla h(\begin{bmatrix} -\frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \\ 1+2\sqrt{2} \end{bmatrix}) = \begin{bmatrix} -\sqrt{2} & 2 \\ -\sqrt{2} & 2 \\ 0 & 1 \end{bmatrix} \sim \begin{bmatrix} 1 & -\sqrt{2} \\ 0 & 1 \\ 0 & 0 \end{bmatrix}$$

and similarly

$$\nabla h(\begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \\ 1-2\sqrt{2} \end{bmatrix}) = \begin{bmatrix} \sqrt{2} & 2 \\ \sqrt{2} & 2 \\ 0 & 1 \end{bmatrix} \sim \begin{bmatrix} 1 & \sqrt{2} \\ 0 & 1 \\ 0 & 0 \end{bmatrix}$$

each of which have rank 2. So the constraint qualification holds at both stationary points.

Solution to Exercise 4.1 [Part 1] – Continued

Thus, by Theorem 4, (and since f and h are obviously C^1), if the NLP has a locally optimal point, it must be one of these two points. To find out more [Part 2], we need to look at second-order information.

Class Exercise 4.2

Write down and solve the first-order necessary conditions for the following equality-constrained NLP (with $n = 2$):

$$\begin{aligned} \min \quad f(x) = &\quad x_1^2 + x_2^2/4 \\ \text{s.t.} \quad h(x) = &\quad x_1 + x_2 + 1 = 0 \end{aligned}$$

Class exercise, slide 328.

$$\min \; f(x) = x_1^2 + x_2^2/4$$
$$\text{s.t.} \quad h(x) = x_1 + x_2 + 1 = 0$$

* Constraint qualifications: h is affine. ✓

$$\mathcal{L}(x,\eta) = x_1^2 + x_2^2/4 + \eta\,(x_1 + x_2 + 1)$$

* From theorem 4 (~~looking for stationary points~~)
Condition on the Lagrangian for minimality

① $0 = \nabla \mathcal{L}_x(x^*, \eta^*) = 0$

② $0 = h(x^*)$

①
$$\begin{bmatrix} 2x_1 + \eta \\ \frac{x_2}{2} + \eta \end{bmatrix} = 0 \Rightarrow \begin{array}{l} x_1 = -\eta/2 \\ x_2 = -2\eta \end{array}$$

② $x_1 + x_2 + 1 = 0$

$$\frac{-\eta}{2} - 2\eta + 1 = 0 \Rightarrow \boxed{\begin{array}{l} \eta^* = 2/5 \\ x_1^* = -1/5 \\ x_2^* = -4/5 \end{array}}$$

$$\nabla f(x^*) = \begin{bmatrix} -2/5 \\ -2/5 \end{bmatrix} \quad \nabla h(x^*) = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad \eta = 2/5$$

We now prove Theorem 4 for the case $n = 2$, $q = 1$ (one constraint) more formally.

Partial Proof of Theorem 4:

The feasible region must be able to be expressed as a parametric curve $x(t)$, where $x \in \mathbb{R}^n$ and t is a single variable. However, the feasible region is defined as the region $h(x) = 0$. Therefore

$$h(x(t)) = 0 \text{ for all } t.$$

Partial Proof of Theorem 4 – Continued

Differentiating this expression by t using the multi-dimensional chain rule gives us

$$\begin{aligned}\frac{d}{dt}h(x(t)) &= \sum_{i=1}^{n}\frac{\partial}{\partial x_i}h(x(t))\frac{d}{dt}x_i(t)\\ &= \langle \nabla h(x(t)), x'(t)\rangle\\ &= 0.\end{aligned}$$

This shows us that $\nabla h(x(t))$ is orthogonal to $x'(t)$. In particular, if x^* is a local minimum and corresponds to t^* on the parametric curve, we know that $\nabla h(x^*)$ is orthogonal to $x'(t^*)$.

Partial Proof of Theorem 4 – Continued

However, because x^* is a local minimum, the one-dimensional function

$$q(t) = f(x(t)),$$

which is the function 'sliced' along the feasible region, achieves a minimum at t^*. This implies that $q'(t^*) = 0$. However, again from the chain rule,

$$\begin{aligned} q'(t) &= \frac{d}{dt} f(x(t)) \\ &= \sum_{i=1}^{n} \frac{\partial}{\partial x_i} f(x(t)) \frac{d}{dt} x_i(t) \\ &= \langle \nabla f(x(t)), x'(t) \rangle. \end{aligned}$$

Partial Proof of Theorem 4 – Continued

This implies that $\langle \nabla f(x(t^*)), x'(t^*) \rangle = 0$, or that $\nabla f(x(t^*))$ and $x'(t^*)$ are orthogonal. But $x'(t^*)$ is also orthogonal to $\nabla h(x^*)$! Therefore $\nabla h(x^*)$ and $\nabla f(x(t^*))$ are parallel (remember we are in two dimensions). This gives

$$\nabla f(x(t^*)) \propto \nabla h(x^*)$$

$$\nabla f(x(t^*)) = -\eta^* \nabla h(x^*)$$

$$\nabla f(x(t^*)) + \eta^* \nabla h(x^*) = 0$$

which, again, is Lagrange's condition.

Subsection 2

Minimality conditions

Determining Minimality of Stationary Points

Theorem 5 (Condition on an equality-constrained NLP for global minimality)

If f is C^1 and convex, and h is affine, then a point x^* is stationary for the equality constrained NLP if and only if it is a global minimum.

This is similar to the unconstrained result which requires f to be C^1 and convex.

If the objective function is non-convex and/or the constraints are nonlinear, then the above result does not hold. In this case we rely on a theorem giving second order sufficient condition for a stationary point to be a local minimum. First we need some definitions.

Determining Minimality of Stationary Points

Definitions:
Let x^* be a stationary point, with a corresponding Lagrange multiplier η^*. Define $\nabla^2_{xx} L(x^*, \eta^*)$ to be the Hessian of the Lagrangian with respect to x, that is

$$\nabla^2_{xx} L(x^*, \eta^*) \;=\; \nabla^2 f(x^*) + \sum_{j=1}^{q} \eta_j^* \nabla^2 h_j(x^*).$$

Define the *set of feasible directions* at x^* to be

$$\mathcal{C}(x^*) = \{d \in \mathbb{R}^n : d \neq 0, \nabla h(x^*)^T d = 0\}.$$

$\mathcal{C}(x^*)$ is the nullspace (or kernel) of the transpose of the Jacobian $\nabla h(x^*)$.

Determining Minimality of Stationary Points

The next theorem provides a second-order sufficient condition for x^* to be a local minimum.

Theorem 6 (Condition on an equality-constrained NLP for local minimality)

If $\nabla^2_{xx} L(x^*, \eta^*)$ is positive definite on $\mathcal{C}(x^*)$, that is

$$\text{if} \quad 0 \neq d \in \mathbb{R}^n, \ \nabla h(x^*)^T d = 0, \quad \text{and} \quad d^T \nabla^2_{xx} L(x^*, \eta^*) d \ > \ 0,$$

then x^* is a local minimum.

This is analogous to the unconstrained second-order condition which requires $\nabla^2 f(x^*)$ to be positive definite.

Determining Minimality of Stationary Points

Theorem 6 tells us that if the Hessian of the Lagrange function with respect to the x variables is positive definite in directions which maintain feasibility (in a local sense, at least), then the stationary point must be a local optimum (assuming that f and h are C^2).

We will continue to work on Exercise 4.1:
Solve the equality-constrained NLP

$$\begin{array}{lrl}
\min & f(x) = & 2x_1^2 + 2x_2^2 + 4x_1x_2 + x_1x_3 + x_2x_3 \\
\text{s.t.} & h_1(x) = & x_1^2 + x_2^2 \qquad\qquad -1 = 0 \\
 & h_2(x) = & 2x_1 + 2x_2 + x_3 - 1 = 0.
\end{array}$$

Note:
The first sufficient condition, given in Theorem 5, (requiring f to be convex and h to be affine) does not apply, as h is not affine. Hence we will apply Theorem 6.

Solution to Exercise 4.1 [Part 2]

In Part 1, we used Theorem 4 to show that the NLP has two stationary points:

$$x^* = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \\ 1 - 2\sqrt{2} \end{bmatrix} \quad \text{and} \quad x^* = \begin{bmatrix} -\frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \\ 1 + 2\sqrt{2} \end{bmatrix}.$$

Their corresponding Lagrange multipliers:

$$\eta^* = \begin{bmatrix} -\frac{1}{\sqrt{2}} \\ -\sqrt{2} \end{bmatrix} \quad \text{and} \quad \eta^* = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \sqrt{2} \end{bmatrix}$$

respectively.

We also showed that these points satisfy the constraint qualifications.

Solution to Exercise 4.1 [Part 2] – Continued

We now attempt to apply Theorem 6. f and h are obviously C^2. Recall that the Lagrangian is:

$$\begin{aligned} L(x,\eta) &= 2x_1^2 + 2x_2^2 + 4x_1x_2 + x_1x_3 + x_2x_3 \\ &\quad + \eta_1(x_1^2 + x_2^2 - 1) + \eta_2(2x_1 + 2x_2 + x_3 - 1). \end{aligned}$$

Taking the Hessian of this, we have

$$\nabla^2_{xx} L(x,\eta) = \begin{bmatrix} 4+2\eta_1 & 4 & 1 \\ 4 & 4+2\eta_1 & 1 \\ 1 & 1 & 0 \end{bmatrix}.$$

Solution to Exercise 4.1 [Part 2] – Continued

Consider the stationary point $x^* = \begin{bmatrix} -\frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \\ 1 + 2\sqrt{2} \end{bmatrix}$ with Lagrange multipliers $\eta^* = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \sqrt{2} \end{bmatrix}$.

The Hessian of the Lagrangian at this point is

$$\nabla^2_{xx} L(x^*, \eta^*) = \begin{bmatrix} 4 + \sqrt{2} & 4 & 1 \\ 4 & 4 + \sqrt{2} & 1 \\ 1 & 1 & 0 \end{bmatrix}.$$

We wish to know if this is positive definite with respect to the directions which maintain feasibility.

Solution to Exercise 4.1 [Part 2] – Continued

To find these directions (using Theorem 6) we compute the nullspace of the transpose of the Jacobian $\nabla h(x)$.

Recall that the *Jacobian matrix* $\nabla h(x)$ is a 3×2 matrix with the 1st column equal to $\nabla h_1(x)$ and the second column equal to $\nabla h_2(x)$.

Hence, for this exercise,

$$\nabla h(x) = \begin{bmatrix} 2x_1 & 2 \\ 2x_2 & 2 \\ 0 & 1 \end{bmatrix}.$$

Solution to Exercise 4.1 [Part 2] – Continued

Now

$$\nabla h(x^*)^T d = 0 \quad \Rightarrow \quad \begin{bmatrix} -\sqrt{2} & -\sqrt{2} & 0 \\ 2 & 2 & 1 \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\Rightarrow \quad \left.\begin{array}{c} -\sqrt{2}d_1 - \sqrt{2}d_2 = 0 \\ 2d_1 + 2d_2 + d_3 = 0 \end{array}\right\} \quad \Rightarrow \quad d_2 = -d_1 \quad \text{and} \quad d_3 = 0$$

$$\Rightarrow \quad d = \begin{bmatrix} d_1 \\ -d_1 \\ 0 \end{bmatrix}$$

for some $d_1 \in \mathbb{R}$.

Solution to Exercise 4.1 [Part 2] – Continued

Thus the directions of interest are given by

$$\mathcal{C}(x^*) = \{d \in \mathbb{R}^n \;:\; d = \begin{bmatrix} d_1 \\ -d_1 \\ 0 \end{bmatrix},\; d_1 \in \mathbb{R},\; d_1 \neq 0\}.$$

We need to check if $d^T \nabla^2_{xx} L(x^*, \eta^*) d > 0$ for all $d \in \mathcal{C}(x^*)$; if so, we can then apply Theorem 6 to deduce that x^* is a local optimum.

Solution to Exercise 4.1 [Part 2] – Continued

For $d_1 \in \mathbb{R}$, $d_1 \neq 0$, we have

$$\begin{aligned}
&(d_1, -d_1, 0)\nabla^2_{xx} L(x^*, \eta^*) \begin{bmatrix} d_1 \\ -d_1 \\ 0 \end{bmatrix} \\
&= (d_1, -d_1, 0) \begin{bmatrix} 4+\sqrt{2} & 4 & 1 \\ 4 & 4+\sqrt{2} & 1 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} d_1 \\ -d_1 \\ 0 \end{bmatrix} \\
&= (d_1, -d_1, 0) \begin{bmatrix} \sqrt{2}d_1 \\ -\sqrt{2}d_1 \\ 0 \end{bmatrix} \\
&= 2\sqrt{2}d_1^2 \qquad (4.3) \\
&> 0.
\end{aligned}$$

Solution to Exercise 4.1 [Part 2] – Continued

Thus $\nabla^2_{xx} L(x^*, \eta^*)$ is positive definite on $\mathcal{C}(x^*)$, so the second order sufficiency condition does holds. Therefore the point $x^* = \left(-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, 1 + 2\sqrt{2}\right)$ is a local minimum, with Lagrange multipliers $\eta^* = \left(\frac{1}{\sqrt{2}}, \sqrt{2}\right)$.

What about $x^* = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 1 - 2\sqrt{2}\right)$?

Solution to Exercise 4.1 [Part 2] – Continued

The Hessian of the Lagrangian at this point is

$$\nabla^2_{xx} L(x^*, \eta^*) = \begin{bmatrix} 4 - \sqrt{2} & 4 & 1 \\ 4 & 4 - \sqrt{2} & 1 \\ 1 & 1 & 0 \end{bmatrix}.$$

The transpose of the Jacobian at this point is

$$\nabla h(x^*)^T = \begin{bmatrix} \sqrt{2} & \sqrt{2} & 0 \\ 2 & 2 & 1 \end{bmatrix}.$$

Solution to Exercise 4.1 [Part 2] – Continued

Now

$$\nabla h(x^*)^T d = 0 \quad \Rightarrow \quad \begin{bmatrix} \sqrt{2} & \sqrt{2} & 0 \\ 2 & 2 & 1 \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\Rightarrow \quad \left. \begin{array}{c} \sqrt{2}d_1 + \sqrt{2}d_2 = 0 \\ 2d_1 + 2d_2 + d_3 = 0 \end{array} \right\} \quad \Rightarrow \quad d_2 = -d_1 \quad \text{and} \quad d_3 = 0$$

$$\Rightarrow \quad d = \begin{bmatrix} d_1 \\ -d_1 \\ 0 \end{bmatrix}$$

for some $d_1 \in \mathbb{R}$.

Solution to Exercise 4.1 [Part 2] – Continued

Thus the directions of interest are given by

$$\mathcal{C}(x^*) = \{d \in \mathbb{R}^n \ : \ d = \begin{bmatrix} d_1 \\ -d_1 \\ 0 \end{bmatrix}, \ d_1 \in \mathbb{R}, \ d_1 \neq 0\}.$$

We need to check if $d^T \nabla^2_{xx} L(x^*, \eta^*) d > 0$ for all $d \in \mathcal{C}(x^*)$; if so, we can then apply Theorem 6 to deduce that x^* is a local optimum.

Solution to Exercise 4.1 [Part 2] – Continued

For $d_1 \in \mathbb{R}$, $d_1 \neq 0$, we have

$$\begin{aligned}
&(d_1, -d_1, 0)\nabla^2_{xx} L(x^*, \eta^*) \begin{bmatrix} d_1 \\ -d_1 \\ 0 \end{bmatrix} \\
&= (d_1, -d_1, 0) \begin{bmatrix} 4-\sqrt{2} & 4 & 1 \\ 4 & 4-\sqrt{2} & 1 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} d_1 \\ -d_1 \\ 0 \end{bmatrix} \\
&= (d_1, -d_1, 0) \begin{bmatrix} -\sqrt{2}d_1 \\ \sqrt{2}d_1 \\ 0 \end{bmatrix} \\
&= -2\sqrt{2}d_1^2 \qquad (4.4) \\
&< 0.
\end{aligned}$$

Solution to Exercise 4.1 [Part 2] – Continued

Thus $\nabla^2_{xx} L(x^*, \eta^*)$ is certainly *not* positive definite on $\mathcal{C}(x^*)$, so the second order sufficiency condition does *not* hold, and we cannot deduce that this point is a local optimum.
(In fact, $\nabla^2_{xx} L(x^*, \eta^*)$ is *negative* definite on $\mathcal{C}(x^*)$ so x^* is a local maximum.)

Putting this together with the fact that the only other stationary point is a minimum, we deduce that the only local minimum of the function lies at the point $\left(-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, 1 + 2\sqrt{2}\right)$.

Extension of Class Exercise 4.2

We found the stationary point for the following equality-constrained NLP (with $n = 2$):

$$\begin{array}{lll} \min & f(x) = & x_1^2 + x_2^2/4 \\ \text{s.t.} & h(x) = & x_1 + x_2 + 1 = 0, \end{array}$$

namely, $x^* = \begin{bmatrix} -1/5 \\ -4/5 \end{bmatrix}$ with Lagrange multiplier $\eta^* = 2/5$.

Using Theorems 5 or 6, confirm that this stationary point is the global minimum.

Class exercise . slide $\underline{352}$.

Theorem 6

$$\mathcal{L}(x,\eta) = x_1^2 + x_2^2/4 + \eta(x_1 + x_2 + 1)$$

$$\nabla^2_{xx} \mathcal{L}(x,\eta) = \begin{bmatrix} 2 & 0 \\ 0 & 1/2 \end{bmatrix}$$ which is always positive definite.

in particular for : $\nabla h(x)^T \cdot d = 0$

$$\nabla h(x) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} = 0 \Rightarrow$$

$d_1 + d_2 = 0$
direction that maintains feasibility

$$\begin{bmatrix} d_1 \\ -d_1 \end{bmatrix}$$

Theorem 5

f is convex since $\nabla^2 f = \begin{bmatrix} 2 & 0 \\ 0 & 1/2 \end{bmatrix}$ is always positive definite.

h is affine.

Proofs of Theorems 5 and 6

Theorem 5 (Condition on an NLP for global minimality)

If f is C^1 and convex, and h is affine, then a point x^* is stationary for the equality constrained NLP if and only if it is a global minimum.

Theorem 6 (Condition on an NLP for local minimality)

If $\nabla^2_{xx} L(x^*, \eta^*)$ is positive definite on $\mathcal{C}(x^*)$, that is

$$\text{if} \quad 0 \neq d \in \mathbb{R}^n, \ \nabla h(x^*)^T d = 0, \quad \text{then} \quad d^T \nabla^2_{xx} L(x^*, \eta^*) d > 0,$$

then x^* is a local minimum.

Proof of Theorem 5

First note that since f is convex, any local minimum is also a global minimum.
Now, applying proof by contradiction, assume that f is convex and h is affine and the Lagrangian conditions hold at x^*, but x^* is not a minimum. Then there must exist another feasible point x' for which $f(x') < f(x^*)$.

Because h is affine, and both x' and x^* are feasible, the line joining x' to x^* must also be feasible.
By the convexity of f, it must lie below the line joining $(x', f(x'))$ and $(x^*, f(x^*))$. In particular, if we set $d = x' - x^*$, this means that the directional derivative of f at x^* in the direction of d must be less than or equal to

$$\frac{f(x') - f(x^*)}{||x' - x^*||} < 0.$$

Proof of Theorem 5 – Continued

But from the Lagrangian conditions, we know that

$$\nabla f(x^*) + \nabla h(x^*)^T \eta^* = 0$$

and therefore the directional derivative of f at x^* in the direction of d is

$$\begin{aligned}\langle \nabla f(x^*), d\rangle &= \nabla f(x^*)^T d \\ &= -\left[\nabla h(x^*)\eta^*\right]^T d \\ &= -(\eta^*)^T(\nabla h(x^*)^T d).\end{aligned}$$

Proof of Theorem 5 – Continued

Given that $d = x' - x^*$, and $h(x') = h(x^*) = 0$, we can say that d is parallel to each of the level surfaces $h_j(x) = 0$. But $\nabla h_j(x^*)$ is normal to these level surfaces at x^*, which means that $\nabla h_j(x^*)^T d = 0$, which in turn implies

$$\nabla h(x^*)^T d = 0.$$

From our calculations, this shows that the directional derivative of f at x^* in the direction of d must be 0, which contradicts the fact that it is strictly less than 0. Such a contradiction can only imply that there is no such x', i.e., that x^* is indeed a local (and global) minimum.

QED

Proofs of Theorems 5 and 6

Theorem 5 (Condition on an NLP for global minimality)

If f is C^1 and convex, and h is affine, then a point x^* is stationary for the equality constrained NLP if and only if it is a global minimum.

Theorem 6 (Condition on an NLP for local minimality)

If $\nabla^2_{xx} L(x^*, \eta^*)$ is positive definite on $\mathcal{C}(x^*)$, that is

$$\text{if} \quad 0 \neq d \in \mathbb{R}^n,\ \nabla h(x^*)^T d = 0, \quad \text{then} \quad d^T \nabla^2_{xx} L(x^*, \eta^*) d > 0,$$

then x^* is a local minimum.

Proof of Theorem 6

Let d be a feasible direction at x^*, i.e., $d \in \mathcal{C}(x^*)$. Using the Taylor series expansion, for small t,

$$\begin{aligned} f(x^* + td) &= f(x^*) + \langle \nabla f(x^*), td \rangle + \frac{1}{2}\langle td, \nabla^2 f(x^*)(td) \rangle + o(t^2) \\ &= f(x^*) + t \nabla f(x^*)^T d + \frac{1}{2} t^2 d^T \nabla^2 f(x^*) d + o(t^2). \end{aligned}$$

As in the previous argument, we know that d is parallel to each of the level surfaces $h_j(x) = 0$ at x^*. This means that

$$\nabla h(x^*)^T d = 0.$$

Proof of Theorem 6 – Continued

Since the Lagrangian conditions are satisfied, we can use the same argument:

$$\begin{aligned} t\nabla f(x^*)^T d &= -t\left[\nabla h(x^*)\eta^*\right]^T d \\ &= -t(\eta^*)^T \nabla h(x^*)^T d \\ &= 0. \end{aligned}$$

We also assume that the Hessian of the Lagrangian is positive definite on $\mathcal{C}(x^*)$. This means that for all feasible directions d,

$$d^T \left[\nabla^2 f(x^*) + \sum_{j=1}^{q} \eta_j^* \nabla^2 h_j(x^*)\right] d > 0.$$

Proof of Theorem 6 – Continued

It can be shown (but would take too long to do it here) that in fact

$$d^T \nabla^2 f(x^*) d > 0.$$

Since

$$f(x^* + td) = f(x^*) + \frac{1}{2} t^2 d^T \nabla^2 f(x^*) d + o(t^2),$$

and for small t the second term will always dominate the third, this means that as we travel along any feasible direction, the function will increase. Therefore x^* is a local minimum and the theorem is (sort of) proved.
QED

Subsection 3

Lagrange multipliers and sensitivity analysis

Lagrange Multipliers and Sensitivity Analysis

It is not immediately obvious what Lagrange multipliers mean. Each optimal point has one Lagrange multiplier for each constraint, but how can we interpret their values?

Lagrange multipliers can be interpreted as *shadow prices* (economically), or mathematically as rates of change of the optimal f as the level of constraint changes.

To see this, consider each constraint as a resource constraint that we must satisfy.

Lagrange Multipliers and Sensitivity Analysis - Exercise

Consider the following problem:

$$\begin{aligned} \min\ & f(x) = x_1^2 + x_2^2/4 \\ \text{s.t.}\ & h(x) = x_1 + x_2 + 1 = 0 \end{aligned}$$

In Exercise 4.2 we showed this has a global minimum at $x^* = (x_1^*, x_2^*) = (-1/5, -4/5)$, with Lagrange multiplier $\eta^* = 2/5$.

We can think of the constraint as representing some kind of resource, of which we have -1 in total (it is obviously a strange kind of resource!), and of which we use $x_1^* + x_2^*$ units.

Lagrange Multipliers and Sensitivity Analysis - Exercise

Now, if we increase the level of resource that we have available to use by a small amount Δ, so the constraint becomes

$$x_1^* + x_2^* = -1 + \Delta,$$

then we end up with a new optimal solution. This optimal solution is probably only slightly different from the old one, so we say it is $x^* + \Delta x$, where $\Delta x = (\Delta x_1, \Delta x_2)$.
Substituting this value into the new constraint shows us that

$$x_1^* + \Delta x_1 + x_2^* + \Delta x_2 = -1 + \Delta$$

and since x^* is feasible for the original problem, this means that

$$\Delta x_1 + \Delta x_2 = \Delta.$$

Lagrange Multipliers and Sensitivity Analysis - Exercise

Then the new optimal f is given by the first-order Taylor series expansion

$$f(x^* + \Delta x) \approx f(x^*) + \nabla f(x^*)^T \Delta x.$$

Given that $\nabla f(x^*) = (-\frac{2}{5}, -\frac{2}{5}) = -\eta^*(1,1)$, this shows us that

$$\begin{aligned} f(x^* + \Delta x) - f(x^*) \approx & -\eta^*(1,1) \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \end{bmatrix} \\ = & -\eta^*(\Delta x_1 + \Delta x_2) \\ = & -\eta^* \Delta. \end{aligned}$$

So if we change our resource value by a small amount, the value that our optimal function value changes by is proportional to that small amount, and the proportionality constant is $-\eta^*$.

Lagrange Multipliers and Sensitivity Analysis - Exercise

Another way of putting this is

$$\frac{f(x^* + \Delta x) - f(x^*)}{\Delta} \approx -\eta^*.$$

So as we change the amount of resource we have available, the rate of change of our optimal function value is $-\eta^*$.

We call this a *shadow price* because it is the maximum amount we would pay to have another unit of resource available to us (if we have to pay exactly that amount, the cost and the extra 'revenue' balance out).

Lagrange Multipliers and Sensitivity Analysis - Generalisation

In a more general case, suppose that we have to minimise a function $f(x)$ subject to a single linear equality constraint $a^T x = b$. Suppose further that (x^*, η^*) is a local minimum and corresponding Lagrange multiplier for this problem. By Lagrange's condition,

$$\nabla f(x^*) = -\eta^* \nabla h(x^*) = -\eta^* a.$$

Now suppose that the right-hand side of the constraint changes by an amount Δ to $b + \Delta$, and this changes the local minimum to a point $x^* + \Delta x$. Then

$$b + \Delta = a^T(x^* + \Delta x) = a^T x^* + a^T \Delta x.$$

Lagrange Multipliers and Sensitivity Analysis - Generalisation

This implies that

$$\Delta = a^T \Delta x.$$

The amount that the optimal f value is changed because of the change in the constraint is

$$\begin{aligned} f(x^* + \Delta x) - f(x^*) &\approx \nabla f(x^*)^T \Delta x \\ &= -\eta^* a^T \Delta x \\ &= -\eta^* \Delta. \end{aligned}$$

Lagrange Multipliers and Sensitivity Analysis - Generalisation

So the same conclusions can be drawn: a small amount of change in the right-hand side of a constraint results in a proportional change in the optimal function value, and the proportionality constant is $-\eta^*$.

Both these cases are specialised cases of the general theorem, stated below.

Theorem (Sensitivity for equality-constrained NLPs)

Consider the family of equality-constrained nonlinear programs

$$\begin{aligned} &\min_{x} f(x) \\ &\text{s.t. } h(x) = u. \end{aligned}$$

Suppose for $u = 0$, there is a local solution x^* that is a regular point and that, together with its associated Lagrange multiplier η^*, satisfies the second-order sufficient conditions for a strict local minimum. Then, there exists an open sphere S, centred at $u = 0$, such that for every $u \in S$, there is an $x(u)$ and an $\eta(u)$, which are a local minimum-Lagrange multiplier pair of the problem above. Furthermore, $x(u)$ and $\eta(u)$ are continuous functions of $u \in S$ and $x(0) = x^*$ and $\eta(u) = \eta^*$. In addition, for all $u \in S$,

$$\nabla_u f(x(u)) = -\eta(u).$$

Lagrange Multipliers and Sensitivity Analysis - Generalisation

In other words, $-\eta(u)$ is the rate at which the optimal value of f changes, as u is changed.

Subsection 4

KKT Conditions

Karush-Kuhn-Tucker conditions for nonlinear programs

We now focus on NLPs with both inequality and equality constraints. The optimality conditions for such problems were developed first by Karush, and then jointly by Kuhn and Tucker. Recall the problem of interest is the *nonlinear program*,

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{s.t.} \quad & g(x) \leq 0 \\ & h(x) = 0, \end{aligned} \tag{NLP}$$

where the vector inequality $g(x) \leq 0 \in \mathbb{R}^p$ means $g_i(x) \leq 0$ for each $i = 1, \ldots, p$, and the vector equality $h(x) = 0 \in \mathbb{R}^q$ means $h_j(x) = 0$ for each $j = 1, \ldots, q$.

Karush-Kuhn-Tucker conditions for nonlinear programs

The Lagrange function for (NLP) is

$$\begin{aligned} L(x, \lambda, \eta) &= f(x) + \sum_{i=1}^{p} \lambda_i g_i(x) + \sum_{j=1}^{q} \eta_j h_j(x) && (4.5) \\ &= f(x) + \langle \lambda, g(x) \rangle + \langle \eta, h(x) \rangle, && (4.6) \end{aligned}$$

where $\lambda \in \mathbb{R}^p$ is the multiplier corresponding to $g(x)$ and $\eta \in \mathbb{R}^q$ is the multiplier corresponding to $h(x)$. The vectors λ and η are generally called Lagrange or KKT multipliers.

Active and inactive constraints

We introduce the concept of an *active* constraint at a feasible point x^*. These are the constraints that are satisfied with equality.

All the equality constraints $h_j(x^*) = 0$ are active, and each inequality constraint $g_i(x^*) \leq 0$ such that $i \in I(x^*) := \{i \, : \, g_i(x^*) = 0\}$ is also active.

$I(x^*)$ is *the set of active inequality constraint indices*.
Note that if x^* is a local minimum of the inequality constrained problem, then x^* must also be a local minimum for a version of the same problem which has all inactive constraints at x^* discarded.

Active and inactive constraints

It follows that *inactive constraints at x^* don't matter*, and can be ignored in the statement of optimality conditions.

In contrast, at a local minimum, active inequality constraints can be treated to a large extent as equalities. That is, if x^* is a local minimum of the inequality constrained problem (NLP), then x^* is also a local minimum for the equality constrained problem

$$\begin{aligned} \min_x \quad & f(x) \\ \text{s.t.} \quad & h(x) = 0 \quad \text{and} \\ & g_i(x) = 0 \quad \text{for all} \quad i \in I(x^*). \end{aligned}$$

Therefore, treating this temporarily as an equality constrained problem, Theorem 4 suggests that if one of the constraint qualifications holds at x^*, then there should exist Lagrange multipliers $\eta_i^*, i = 1, \ldots, q$, and $\lambda_j^*, j \in I(x^*)$ such that

$$\nabla_x L(x^*, \lambda^*, \eta^*) = \nabla f(x^*) + \sum_{j \in I(x^*)} \lambda_j^* \nabla g_j(x^*) + \sum_{i=1}^{q} \eta_i^* \nabla h_i(x^*) = 0$$

If the Lagrange multipliers corresponding to inactive constraints are set to zero, we can write this as

$$\nabla_x L(x^*, \lambda^*, \eta^*) = \nabla f(x^*) + \sum_{j=1}^{p} \lambda_j^* \nabla g_j(x^*) + \sum_{i=1}^{q} \eta_i^* \nabla h_i(x^*) = 0$$

where we have set $\lambda_j^* = 0$ for $j \notin I(x^*)$. This motivates the following...

Theorem 7 (KKT Conditions)

Let f, g and h be C^1 functions, and assume that one of the constraint qualifications (discussed below) on g and h holds at x^*. If x^* is a local minimum of (NLP), then there exist $\lambda^* \in \mathbb{R}^p$ and $\eta^* \in \mathbb{R}^q$ such that

KKTa $0 = \nabla_x L(x^*, \lambda^*, \eta^*)$, that is

$$0 = \nabla f(x^*) + \nabla g(x^*)\lambda^* + \nabla h(x^*)\eta^*.$$

KKTb
1. $g(x^*) \leq 0$,
2. $\lambda^* \geq 0$, and
3. $\lambda_i^* g_i(x^*) = 0$, $i = 1, 2, \ldots, p$.

KKTc $h(x^*) = 0$.

Karush-Kuhn-Tucker conditions for nonlinear programs

If x^* is a local minimum, the KKT conditions are satisfied; however, the KKT conditions may be satisfied without x^* being a local minimum.

If the KKT conditions are satisfied at (x^*, λ^*, η^*), we say that (x^*, λ^*, η^*) is a *KKT point*. KKT points are basically stationary points of the non-linear program.

Let's investigate the KKT conditions by looking at a one-dimensional problem with a single inequality constraint,

$$\min_{x} f(x) \quad \text{s.t.} \quad x \geq 5. \tag{4.7}$$

Suppose x^* is a local minimum. If $x^* > 5$, then x^* is also a local minimum of the unconstrained problem, $\min f(x)$, hence $\nabla f(x^*) = 0$. Thus x^* is stationary for NLP, taking $\lambda^* = 0$.

If $x^* = 5$, then x^* need not be stationary for the unconstrained problem, but $f'(5) \geq 0$, because if $f'(5) < 0$ then $d = 1$ is a descent direction for f at $x^* = 5$, hence $f(5 + t) < f(5)$ for some small $t > 0$. This would contradict $x^* = 5$ being a local minimum of the constrained problem.

Karush-Kuhn-Tucker conditions for nonlinear programs

Let's compare the above observations with the KKT conditions. Let $g(x) = 5 - x$. We have no equality constraints $h(x)$ (that is $q = 0$). For $\lambda \in \mathbb{R}$,

$$L(x, \lambda) \ = \ f(x) + \lambda(5 - x), \quad \nabla_x L(x, \lambda) \ = \ \nabla f(x) - \lambda.$$

KKTa says $\nabla f(x^*) = \lambda^*$; and KKTb says

1. $5 - x^* \leq 0$,
2. $\lambda^* \geq 0$, and
3. either $5 - x^* = 0$ or $\lambda^* = 0$.

The third condition, KKTc, is vacuous because there are no equality constraints.

Karush-Kuhn-Tucker conditions for nonlinear programs

If $x^* > 5$, KKTb(c) implies that $\lambda^* = 0$, hence $\nabla f(x^*) = 0$ from KKTa. If $x^* = 5$, λ^* is some nonnegative number and we deduce, from KKTa, that $\nabla f(x^*) \geq 0$.

Thus the KKT conditions are just the algebraic statement of our intuitive observations about the gradient of f at a solution point.

We have different "behaviour" depending upon whether the optimal solution is on a "boundary" of an inequality constraint set (constraint is active), or in the "interior" of this set (constraint is inactive).

Summary – KKT conditions for nonlinear programs

Consider the NLP:

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{s.t.} \quad & g(x) \le 0, \\ & h(x) = 0. \end{aligned} \tag{NLP}$$

The Lagrange function for (NLP) is

$$\begin{aligned} L(x, \lambda, \eta) &= f(x) + \sum_{i=1}^{p} \lambda_i g_i(x) + \sum_{j=1}^{q} \eta_j h_j(x) \\ &= f(x) + \langle \lambda, g(x) \rangle + \langle \eta, h(x) \rangle, \end{aligned}$$

where λ and η are the Lagrange or KKT multipliers.

Theorem 7 (KKT Conditions)

Let f, g and h be C^1 functions, and assume that one of the constraint qualifications (discussed below) on g and h holds at x^*. If x^* is a local minimum of (NLP), then there exist $\lambda^* \in \mathbb{R}^p$ and $\eta^* \in \mathbb{R}^q$ such that

KKTa $0 = \nabla_x L(x^*, \lambda^*, \eta^*)$, that is

$$0 = \nabla f(x^*) + \nabla g(x^*)\lambda^* + \nabla h(x^*)\eta^*.$$

KKTb

1. $g(x^*) \leq 0$,
2. $\lambda^* \geq 0$, and
3. for each i, $\lambda_i^* g_i(x^*) = 0$.

KKTc $h(x^*) = 0$.

Summary – KKT conditions for nonlinear programs

We also defined the concept of *active* constraints at a feasible point x^*. These are the constraints that are satisfied with equality.

All the equality constraints $h_j(x^*) = 0$ are active, and each inequality constraint $g_i(x^*) \leq 0$ such that $i \in I(x^*) := \{i : g_i(x^*) = 0\}$ is also active.

$I(x^*)$ is *the set of active inequality constraint indices*.

KKT conditions for nonlinear programs

Homework Exercise.

1. Find all KKT points of the 1-dimensional problem with a single inequality constraint, $\min_x e^x$ subject to $x \geq 5$.

2. Find all KKT points of the problem with the inequality reversed: $x \leq 5$.

3. Find all KKT points of the problem with an equality constraint: $x = 5$.

Homework Exercise - Solution to [1]

The Lagrangian is $L(x, \lambda) = e^x + \lambda(5 - x)$.

Hence,

$$\text{KKTa:} \qquad \nabla_x L(x^*, \lambda^*) = e^{x^*} - \lambda^* = 0$$

$$\text{KKTb:} \qquad 5 - x^* \leq 0, \quad \lambda^* \geq 0, \quad \lambda^*(5 - x^*) = 0$$

There is a unique KKT point: $x^* = 5$ with Lagrange multiplier $\lambda^* = e^5$.

Subsection 5

Constraint qualifications

Constraint qualifications

Theorem 7 requires one of the *constraint qualifications* (also known as "regularity conditions") to be satisfied. We will look at 3 suitable constraint qualifications.

The two simplest constraint qualifications mimic the equality constrained case.

- The first is that all active constraints are affine: i.e., $\{g_i,\ i \in I(x^*)\} \cup \{h_j,\ \text{all } j\}$ are affine functions.
- The second is that the set of all active gradients are linearly independent: i.e., the elements of

$$\{\nabla g_i(x^*) \, : \, i \in I(x^*)\} \cup \{\nabla h_j(x^*) : \ \text{all } j\} \tag{4.8}$$

  are linearly independent.

Constraint qualifications - the Mangasarian-Fromovitz condition

The third constraint qualification we consider is the *Mangasarian-Fromovitz constraint qualification*, which is a weaker condition than the other two.

This requires that the equality constraint gradients $\nabla h_j(x^*)$ are linearly independent, and there exists $d \in \mathbb{R}^n$ such that

1. $\nabla h(x^*)^T d = 0$ and,
2. for $i \in I(x^*)$, $\nabla g_i(x^*)^T d < 0$.

Constraint qualifications - the Mangasarian-Fromovitz condition

The first condition implies that d is orthogonal to the space spanned by the gradients of the equality constraints $\nabla h_j(x^*)$ and the second condition requires that d is a "descent direction" for each of the inequality constraints.

Essentially, it is saying that none of the inequality constraints are implied by the equality constraints at the point x^*.

So Theorem 7 also holds if x^* satisfies the Mangasarian-Fromovitz constraint qualification.

Constraint qualifications

Homework Exercise
For the NLP

$$\begin{array}{ll} \min & x_1 \\ \text{such that} & g_1(x) \stackrel{def}{=} (x_1-1)^2 \;+\; (x_2-1)^2 - 1 \le 0 \\ & g_2(x) \stackrel{def}{=} (x_1-1)^2 \;+\; (x_2+\frac{1}{2})^2 - 1 \le 0. \end{array}$$

sketch the feasible region, and explain, using the sketch, why $x^* = (1 - \frac{\sqrt{7}}{4}, \frac{1}{4})$ is a local (and global) minimum.

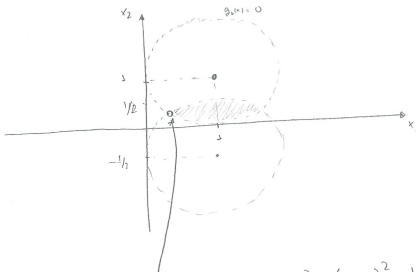
Show that the Mangasarian-Fromowitz condition holds at x^* (i.e., find a suitable $d \in \mathbb{R}^2$). Determine KKT multipliers for x^* and hence show that x^* is a KKT point, as predicted by Theorem 7.

Homework exercise 393

$\min x_1$

$g_1(x) = (x_1-1)^2 + (x_2-1)^2 - 1 \leq 0$

$g_2(x) = (x_1-1)^2 + (x_2+1/2)^2 - 1 \leq 0$



optimal : $(x_1-1)^2 + (x_2-1)^2 = 1$

$(x_1-1)^2 + (x_2+1/2)^2 = 1$

$\Rightarrow (x_2-1)^2 - (x_2+1/2)^2 = 0$

$x_2^2 - 2x_2 + 1 - x_2^2 - x_2 - 1/4 = 0$

$-3x_2 + 3/4 = 0$

$x_2 = 1/4$

$x_1 \Rightarrow$

$(x_1-1)^2 = 1 - (1/4-1)^2$

$(x_1-1)^2 = 7/16$

$x_1 - 1 = \pm \sqrt{7}/4$

$x_1 = \begin{cases} 1+\sqrt{7}/4 \\ 1-\sqrt{7}/4 \end{cases}$

$(x_1, x_2) = \left(1 - \sqrt{7}/4 ,\ 1/4\right)$

Mangarasian - Fromowitz conditions :

* $\nabla h_j(x)$ are linearly independent
* $\exists d \in \mathbb{R}^n$ such that

$$\nabla h(x^*)^T d = 0$$

$$i \in I(x^*), \ \nabla g_i (x^*)^T d < 0$$

* finding x^*

$$\mathcal{L}_{x,\lambda,\eta} = x_1 + \lambda_1\left((x_1-1)^2 + (x_2-1)^2 - 1\right) + \lambda_2\left((x_1-1)^2 + (x_2+1/2)^2 - 1\right)$$

$$\nabla \mathcal{L}_x = \begin{bmatrix} 1 + 2\lambda_1(x_1-1) + 2\lambda_2(x_1-1) \\ 2\lambda_1(x_2-1) + 2\lambda_2(x_2+1/2) \end{bmatrix} = \begin{bmatrix} 1 + 2(\lambda_1+\lambda_2)(x_1-1) \\ 2\lambda_1(x_2-1) + 2\lambda_2(x_2+1/2) \end{bmatrix}$$

For $(x_1, x_2) = (1-\sqrt{7}/4, \ 1/4)$

$$\Rightarrow \begin{cases} 1 + 2(\lambda_1+\lambda_2)\cdot(-\sqrt{7}/4) = 0 \\ 2\lambda_1(-3/4) + 2\lambda_2(3/4) = 0 \Rightarrow \underline{\lambda_1 = \lambda_2} \end{cases}$$

$$1 + 4\lambda_1 \cdot (-\sqrt{7}/4) = 0$$

$$4\lambda_1 = \frac{4}{\sqrt{7}} \Rightarrow \boxed{\lambda_1 = \frac{\sqrt{7}}{7} = \lambda_2}$$

$\cdots h(x)$ (there are no inequality constraints)

Regularity conditions

both g' functions are active :

$$\nabla g(x^*) = \begin{bmatrix} 2(x_1^*-1) & 2(x_2^*-1) \\ 2(x_2^*-1) & 2(x_2^*+1/2) \end{bmatrix} = \begin{bmatrix} -\sqrt{7}/2 & -\sqrt{7}/2 \\ -3/2 & 3/2 \end{bmatrix}$$

rank 2

Mangasarian :

no equality constraints $d \in \mathbb{R}^2$

g_1 $\quad \begin{pmatrix} -\frac{\sqrt{7}}{2} & -\frac{3}{2} \end{pmatrix} \begin{pmatrix} d_1 \\ d_2 \end{pmatrix} < 0 \Rightarrow -\frac{\sqrt{7}}{2} d_1 - \frac{3}{2} d_2 < 0$

g_2 $\quad \begin{pmatrix} -\frac{\sqrt{7}}{2} & +3/2 \end{pmatrix} \begin{pmatrix} d_1 \\ d_2 \end{pmatrix} < 0 \Rightarrow -\frac{\sqrt{7}}{2} d_1 + \frac{3}{2} d_2 < 0$

true for $\begin{pmatrix} d_1 \\ d_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$.

Class Exercise 4.3

For the NLP

$$\begin{array}{ll} \min & (x_1^2 + x_2^2 + x_3^2)/2 \\ \text{s.t.} & x_1 + x_2 + x_3 \leq 3, \end{array}$$

find all KKT points.

Class exercise, week 7

Find all KKT points

$$\text{Min} \quad (x_1^2 + x_2^2 + x_3^2)/2$$

$$x_1 + x_2 + x_3 \le 3$$

$$\mathcal{L} = (x_1^2 + x_2^2 + x_3^2)/2 + \lambda \cdot (x_1 + x_2 + x_3 - 3)$$

$$\nabla \mathcal{L}_x = \begin{bmatrix} x_1 + \lambda \\ x_2 + \lambda \\ x_3 + \lambda \end{bmatrix} = 0 \Rightarrow x_1 = x_2 = x_3 = -\lambda$$

if the constraint is inactive

$$\boxed{\lambda = 0 \Rightarrow x_1 = x_2 = x_3 = 0}$$ single KKT point

if the constraint is active

$$3x_1 = 3 \Rightarrow \boxed{x_1 = x_2 = x_3 = 1 , \quad \lambda = -1}$$

KKT 2 violated

Constraint qualifications

In the following example, we shall see that if the Mangasarian-Fromovitz constraint qualification does *not* hold at a local minimum, and then it may not be possible to find KKT multipliers at this point. That is, we show that Theorem 7 breaks down without the constraint qualification.

Example 4.4

Show that the minimum for the NLP below is not a KKT point:

$$\begin{aligned} \min \quad & x_1 + x_2 + x_3^2 \\ \text{s.t.} \quad & g_1(x) \stackrel{def}{=} (x_1 - 1)^2 \quad + \quad (x_2 - 1)^2 - 1 \leq 0, \\ & g_2(x) \stackrel{def}{=} (x_1 - 1)^2 \quad + \quad (x_2 + 1)^2 - 1 \leq 0. \end{aligned}$$

Solution to Example 4.4

It is not hard to see that the feasible set for this NLP is $\{(1, 0, x_3) \ : \ x_3 \in \mathbb{R}\}$ and thus the optimal solution must be $x^* = (1, 0, 0)$, which must also be a local minimum.

Note that there are no equality constraints, so to check the Mangasarian-Fromowitz constraint qualification, we seek $d \in \mathbb{R}^3$ such that $\nabla g_i(x^*)^T d < 0$ for all i with $g_i(x^*) = 0$.

$$\nabla g_1(x) = (2(x_1 - 1), 2(x_2 - 1), 0)$$
$$\nabla g_2(x) = (2(x_1 - 1), 2(x_2 + 1), 0)$$

So $\nabla g_1(1, 0, 0) = (0, -2, 0)$ and $\nabla g_2(1, 0, 0) = (0, 2, 0)$.

Solution to Example 4.4

Because $g_1(1,0,0) = 0$ and $g_2(1,0,0) = 0$, we know that $I(1,0,0) = \{1,2\}$, so we seek d such that $\nabla g_1(1,0,0)^T d < 0$ and $\nabla g_2(1,0,0)^T d < 0$. These reduce to $-2d_2 < 0$ and $2d_2 < 0$.

Clearly there exists no such d, so the Mangasarian-Fromowitz constraint qualification does not hold.

We now show that it is also impossible to find Lagrange multipliers at this point, that is, the point x^* is not a KKT point (and hence Theorem 7 does not hold).

Solution to Example 4.4

First, observe that the Lagrangian is given by

$$\begin{aligned} L(x,\lambda) &= x_1 + x_2 + x_3^2 + \lambda_1((x_1-1)^2 \\ &\quad + (x_2-1)^2 - 1) + \lambda_2((x_1-1)^2 + (x_2+1)^2 - 1) \end{aligned}$$

and so

$$\nabla_x L(x,\lambda) = \begin{pmatrix} 1 + 2\lambda_1(x_1-1) + 2\lambda_2(x_1-1) \\ 1 + 2\lambda_1(x_2-1) + 2\lambda_2(x_2+1) \\ 2x_3 \end{pmatrix}.$$

Solution to Example 4.4

Now suppose x^* *is* a KKT point. Then there must exist KKT multipliers λ such that $\nabla_x L(x^*, \lambda) = 0$. But

$$\nabla_x L((1,0,0),\lambda) = \left(\begin{array}{c} 1 \\ 1 - 2\lambda_1 + 2\lambda_2 \\ 0 \end{array} \right)$$

which cannot equal $(0,0,0)$. So there cannot be such a λ, and x^* is not a KKT point.

Subsection 6

Sufficient optimality conditions

Sufficient optimality conditions

As we saw with both unconstrained and equality-constrained problems, stationarity of x^* is generally not enough to ensure that x^* is a local minimum. To be able to deduce that a stationarity point is a local minimum, we need more information.

Specifically, we need to know that

- (NLP) is a *convex program*, or
- a second-order condition holds.

Sufficient optimality conditions

The first of these two possibilities is dealt with in the following theorem.

Theorem 8 (Minimality of KKT points for convex NLPs)

Suppose f and each component function g_j of g is C^1 and convex, h is affine, and a constraint qualification holds for g and h at x^*. Then the KKT conditions hold at x^* if and only if x^* is a local minimum of (NLP).

Sufficient optimality conditions

If (NLP) is not a convex program then we need to use second-order information to establish that x^* is a local minimum. Let (x^*, λ^*, η^*) satisfy the KKT conditions.

Define the *critical cone* at (x^*, λ^*) to be the set

$$
\begin{aligned}
\mathcal{C}(x^*, \lambda^*) \;:=\; \{d \in \mathbb{R}^n :& \\
& \langle \nabla g_i(x^*), d\rangle \leq 0 \text{ if } i \in I(x^*),\ \lambda_i^* = 0, && (4.9) \\
& \langle \nabla g_i(x^*), d\rangle = 0 \text{ if } \lambda_i^* > 0, && (4.10) \\
& \langle \nabla h_j(x^*), d\rangle = 0, \forall j\}. && (4.11)
\end{aligned}
$$

Sufficient optimality conditions

If a constraint is an equality constraint (at x^*) or an active inequality constraint (at x^*) with strictly positive KKT multiplier, the critical cone restricts d to the directions which 'move along' the constraint.

On the other hand, if a constraint is an active inequality constraint with zero KKT multiplier (which acts much like an inactive constraint), then the critical cone restricts d to the directions which 'move towards feasibility'.

Basically, if we move along any direction in the critical cone, the 'feasibleness' of the solution does not change.

Sufficient optimality conditions

Now observe that the Hessian with respect to x of the Lagrange function at (x^*, λ^*, η^*) is

$$\nabla^2_{xx} L(x^*, \lambda^*, \eta^*) \ = \ \nabla^2 f(x^*) + \sum_i \lambda_i^* \nabla^2 g_i(x^*) + \sum_j \eta_j^* \nabla^2 h_j(x^*).$$

Remark: If g and h are affine functions, hence $\nabla^2 g(x^*)$ and $\nabla^2 h(x^*)$ are zero, then $\nabla^2_{xx} L(x^*, \lambda^*, \eta^*) = \nabla^2 f(x^*)$.

Sufficient optimality conditions

Let x^* be a KKT point (see Theorem 7). The second-order sufficient condition for x^* to be a local minimum is that $\nabla^2_{xx} L(x^*, \lambda^*, \eta^*)$ is positive definite on the critical cone, that is

$$\text{if} \quad 0 \neq d \in \mathcal{C}(x^*, \lambda^*), \qquad \text{then} \quad d^T \nabla^2_{xx} L(x^*, \lambda^*, \eta^*) d > 0.$$

Of course if $\nabla^2_{xx} L$ is positive definite (over all vectors), then this condition is automatically satisfied!

Sufficient optimality conditions

A slightly weaker second-order sufficient condition is often used. Specifically, x^* is a local minimum if

1. $\lambda_i^* > 0$ for $i \in I(x^*)$, and
2. $\nabla_{xx}^2 L(x^*, \lambda^*, \eta^*)$ is positive definite on the linear subspace

$$\{d \in \mathbb{R}^n : \langle \nabla g_i(x^*), d \rangle = 0 \text{ if } \lambda_i^* > 0, \\ \langle \nabla h_j(x^*), d \rangle = 0, \forall j\}.$$

This condition is much easier to check than the other second-order sufficient condition given above, but is more restrictive since it requires the KKT multipliers λ_i^* associated with active constraints to be strictly positive.

Sufficient optimality conditions

The precise statement of the second order conditions that are sufficient for x^* to be a local minimum is the following:

Theorem 9 (Second Order sufficiency condition)

Suppose f, g and h are C^2 functions, and one of the constraint qualifications holds for g and h at x^*. If x^* is stationary and the second-order sufficient condition holds, then x^* is a local minimum of (NLP).

Class Exercise 4.5

Find a KKT point for the problem $\min_x \, x^2 - 2x$ subject to $x \geq 0$. Do the second-order sufficient conditions hold?

Repeat this exercise with $x \leq 0$ instead of $x \geq 0$.

$$\min_{x \in \mathbb{R}} \quad x^2 - 2x \qquad x \geq 0$$

$$\mathcal{L} = x^2 - 2x + \lambda(-x)$$

$$\nabla \mathcal{L}_x = \left[2x - 2 - \lambda \right] = 0 \quad \Rightarrow \quad x = \frac{\lambda + 2}{2}$$

- constraint active:

$$x = 0 \Rightarrow \lambda = -2 \quad \rightarrow \quad KKT_b \text{ does not hold}$$

- constraint inactive $\underline{\lambda = 0}$

$$\nabla f(x) = 0 \qquad \boxed{\boxed{x = 1} \quad \lambda = 0}$$

Constraint qualifications

no active constraints

Second-order conditions

~~critical cone:~~ no critical cone $\Rightarrow d \in \mathbb{R}$

$$\cancel{\langle x, d \rangle \neq 0} \Rightarrow \cancel{d \neq 0}$$

$$d \cdot \nabla^2_{xx} L \, d = d \cdot (2) \, d > 0$$

Now if $x \leq 0$

$$\nabla \mathcal{L} = (x^2 - 2x) + \lambda x$$

$$\nabla \mathcal{L} = 2x - 2 + \lambda = 0$$

(1) Constraint active : $\boxed{x = 0 \Rightarrow \lambda = 2}$ ✓

Critical cone :

$$\langle \nabla g_i(x^*) \cdot d \rangle = 0$$

$$\langle 0 \cdot d \rangle = 0 \quad \text{any } d$$

$$d^T \nabla_{xx}^* \mathcal{L} \, d = d \cdot 2 \cdot d = 2d^2 \geq 0$$ ✓

(2) constraint inactive : $\lambda = 0 \Rightarrow x = 1$?

KKT b_1 fails

Class Exercise 4.6

Consider the NLP

$$\begin{aligned} \min \quad & x_1^3 + 3x_2^2 + 16x_3 \\ \text{s.t.} \quad & x_1 + x_2 + x_3 = 5, \\ & x_1, x_2, x_3 \geq 1. \end{aligned}$$

Write down the KKT conditions and find all stationary points of the NLP together with their corresponding Lagrange multipliers. At each stationary point, identify the active constraints, write down the critical cone and check a second-order condition. Can you deduce any local minima?

Solution to Exercise 4.6

Firstly, we write out the program to give us our constraint functions:

$$\begin{aligned}
\text{min} \quad & f(x) = x_1^3 + 3x_2^2 + 16x_3 \\
\text{such that} \quad & g_1(x) = 1 - x_1 \leq 0 \\
& g_2(x) = 1 - x_2 \leq 0 \\
& g_3(x) = 1 - x_3 \leq 0 \\
& h(x) = x_1 + x_2 + x_3 - 5 = 0.
\end{aligned}$$

The Lagrangian is

$$\begin{aligned}
L(x, \lambda, \eta) = x_1^3 &+ 3x_2^2 + 16x_3 \\
&+ \lambda_1(1 - x_1) + \lambda_2(1 - x_2) + \lambda_3(1 - x_3) \\
&+ \eta(x_1 + x_2 + x_3 - 5).
\end{aligned}$$

Solution to Exercise 4.6 – Continued

The KKT conditions are

$$\text{KKTa:} \quad \begin{aligned} 3x_1^2 - \lambda_1 + \eta &= 0 \\ 6x_2 - \lambda_2 + \eta &= 0 \\ 16 - \lambda_3 + \eta &= 0 \end{aligned}$$

$$\text{KKTb:} \quad \begin{aligned} &1 - x_1 \leq 0, \quad \lambda_1 \geq 0, \quad \lambda_1(1 - x_1) = 0 \\ &1 - x_2 \leq 0, \quad \lambda_2 \geq 0, \quad \lambda_2(1 - x_2) = 0 \\ &1 - x_3 \leq 0, \quad \lambda_3 \geq 0, \quad \lambda_3(1 - x_3) = 0 \end{aligned}$$

$$\text{KKTc:} \; x_1 + x_2 + x_3 - 5 = 0$$

Solution to Exercise 4.6 – Continued

To solve this, we take all possible combinations of inequality constraints to be active and see which ones solve the KKT system. Because we have 3 inequality constraints, and any can be active or inactive, this gives us 8 possibilities!

First we test if all 3 constraints can be active. This is easy since it tells us what the values of the decision variables are straight away.

1. If $\lambda_1, \lambda_2, \lambda_3 > 0$ then by KKTb, we have $x_1 = x_2 = x_3 = 1$. But then $x_1 + x_2 + x_3 - 5 = -2 \neq 0$, so KKTc is not satisfied.

Solution to Exercise 4.6 – Continued

Next we test if two constraints can be active. This is also simple because we know two of the x's, and the third follows from the equality constraint.

2. If $\lambda_1 = 0$, $\lambda_2, \lambda_3 > 0$, then by KKTb, $x_2 = x_3 = 1$. From KKTc, $x_1 + 1 + 1 - 5 = 0$ so $x_1 = 3$. Then KKTa gives us $\eta = -3x_1^2 = -27$, and $\lambda_2 = 6x_2 + \eta = -21 \ngeq 0$, so KKTb is not satisfied.
3. If $\lambda_2 = 0$, $\lambda_1, \lambda_3 > 0$, then by KKTb, $x_1 = x_3 = 1$. From KKTc, $x_2 = 3$. Then KKTa gives us $\eta = -6x_2 = -18$, and $\lambda_1 = 3x_1^2 + \eta = -15 \ngeq 0$. So KKTb is not satisfied.
4. If $\lambda_3 = 0$, $\lambda_1, \lambda_2 > 0$, then by KKTb, $x_1 = x_2 = 1$. From KKTc, $x_3 = 3$. Then KKTa gives us $\eta = -16$, and $\lambda_1 = 3x_1^2 + \eta = -13 \ngeq 0$. So KKTb is not satisfied.

Solution to Exercise 4.6 – Continued

The case with one active constraint is a bit harder.

5. If $\lambda_1 = \lambda_2 = 0$, $\lambda_3 > 0$, then by KKTb, $x_3 = 1$. Then from KKTa, $\eta = -3x_1^2 = -6x_2$, which implies that $x_1 = \sqrt{-\frac{\eta}{3}}$ and $x_2 = -\frac{\eta}{6}$. Then from KKTc, $\sqrt{-\frac{\eta}{3}} - \frac{\eta}{6} + 1 - 5 = 0$, which is a quadratic in η and solves to $\eta = -12$. Then $x_1 = 2 \geq 1$ and $x_2 = 2 \geq 1$. Finally $\lambda_3 = 16 + \eta = 4 \geq 0$, so the point $(2, 2, 1)$ is a KKT point.
6. If $\lambda_1 = \lambda_3 = 0$, $\lambda_2 > 0$, then by KKTb, $x_2 = 1$. From KKTa, $\eta = -3x_1^2 = -16$ so $x_1 = \frac{4}{\sqrt{3}}$. KKTc now gives $x_3 = 4 - \frac{4}{\sqrt{3}} \geq 1$. Finally $\lambda_2 = 6x_2 + \eta = -10 \not\geq 0$, so KKTb is not satisfied.

Solution to Exercise 4.6 – Continued

7. If $\lambda_2 = \lambda_3 = 0$, $\lambda_1 > 0$, then by KKTb, $x_1 = 1$. From KKTa, $\eta = -6x_2 = -16$, so $x_2 = \frac{8}{3}$. KKTc now gives us $x_3 = \frac{4}{3}$, and KKTa gives $\lambda_1 = 3x_1^2 + \eta = -13 \ngeq 0$, so KKTb is not satisfied.

It turns out that the case with no active constraints is relatively easy to solve.

8. If $\lambda_1 = \lambda_2 = \lambda_3 = 0$, then KKTa gives us $\eta = -3x_1^2 = -6x_2 = -16$, which implies $x_1 = \frac{4}{\sqrt{3}}$ and $x_2 = \frac{8}{3}$. KKTc now gives $x_3 = \frac{7}{3} - \frac{4}{\sqrt{3}} \ngeq 1$, so KKTb is violated.

Therefore the only KKT point is $(2, 2, 1)$, with multipliers $(0, 0, 4)$ and -12.

Solution to Exercise 4.6 – Continued

Next we check that a constraint qualification holds. This is easy because all constraints are affine. But because we will need it later, we also check the second constraint qualification. At $(2, 2, 1)$ the active constraints are the third inequality constraint and the equality constraint. Therefore the active gradients are

$$\nabla g_3(x^*) = (0, 0, -1)$$

and

$$\nabla h(x^*) = (1, 1, 1).$$

It is easy to see that they are linearly independent; furthermore, $d = (0, -1, 1)$ also suffices for the Mangasarian-Fromovitz condition.

Solution to Exercise 4.6 – Continued

To complete our analysis, we show that a sufficient optimality condition holds at $(2, 2, 1)$. Note that this NLP is *not* a convex program because the objective function is not convex. The active gradients give us the critical cone:

$$\mathcal{C}(x^*, \lambda^*) = \{d \in \mathbb{R}^n : \langle (0, 0, -1), d \rangle = \langle (1, 1, 1), d \rangle = 0\}$$

If $d = (d_1, d_2, d_3)$, then this means that $-d_3 = 0$ and $d_1 + d_2 + d_3 = 0$. This leads to $d_3 = 0$ and $d_2 = -d_1$. So we can express the critical cone as $d_1(1, -1, 0)$ for any $d_1 \in \mathbb{R}$.

Solution to Exercise 4.6 – Continued

We want to show that the Hessian of the Lagrangian is positive definite on the critical cone. The Hessian is

$$\nabla^2_{xx} L(x, \lambda, \eta) = \begin{bmatrix} 6x_1 & 0 & 0 \\ 0 & 6 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

and

$$\nabla^2_{xx} L((2,2,1),(0,0,4),-12) = \begin{bmatrix} 12 & 0 & 0 \\ 0 & 6 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

which is *not* positive definite (it is positive semi-definite).

Solution to Exercise 4.6 – Continued

But on the critical cone,

$$
\begin{aligned}
d_1 \begin{bmatrix} 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} 12 & 0 & 0 \\ 0 & 6 & 0 \\ 0 & 0 & 0 \end{bmatrix} d_1 \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}
&= d_1^2 \begin{bmatrix} 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} 12 \\ -6 \\ 0 \end{bmatrix} \\
&= 18 d_1^2 > 0
\end{aligned}
$$

so the Hessian is positive definite on the critical cone.

Final Answer: $(2, 2, 1)$ is the only KKT point and is a local minimum of the NLP, with KKT multipliers $(0, 0, 4)$ and -12.

Interpretation of KKT multipliers

KKT multipliers have the same physical interpretation as Lagrange multipliers — if the right-hand side of a constraint changes, then the optimal function value changes an amount proportional to the constraint change. Furthermore, the constant of proportionality is the KKT multiplier corresponding to the constraint.

We can see how this works for active and inactive inequality constraints. If an inequality constraint is inactive, its KKT multiplier must be 0. So changing the constraint does not change the optimal function value at all, which is as expected since it should not change the optimal solution!

Interpretation of KKT multipliers

On the other hand, active constraints have positive KKT multipliers. So if the right-hand side of a constraint with KKT multiplier λ^* changes by Δ, the optimal function value changes (approximately) by $-\lambda^*\Delta$.

In particular, if Δ is positive, we are enlarging the feasible region (recall that the constraint is $g(x) \le 0$). Such a change must improve the optimal function value, which is why we have $\lambda^* \ge 0$ in the KKT conditions — if it were negative, the optimal function value would increase instead!

Course Lecture Plan

1. Unconstrained Optimisation - Single Variable
   - minimising functions of a single variable without the use of derivatives
   - minimising functions of a single variable with the use of derivatives
2. Unconstrained Optimisation - n Variables
   - Optimality conditions
   - Descent Methods - using first derivatives
   - Newton's Method - using first and second derivatives
3. General Constrained optimisation
   - Karush-Kuhn-Tucker (and related) conditions
   - Penalty Methods for NLPs
4. Convex Constrained optimisation
   - Methods based on Convex Duality

Subsection 7

Penalty methods for Constrained Optimisation

Penalty methods for nonlinear programs

In the previous lectures, we explored analytic methods for locating stationary points of constrained non-linear programs and the deciding whether these stationary points are local minima.

As we saw with both the single variable and multivariable unconstrained cases, it is quite often the case that it is difficult to apply these analytic methods. Usually this is because we cannot solve the equation $\nabla f = 0$ analytically or because, having solved this equation, it is difficult to show analytically that a second-order sufficient condition holds.

Penalty methods for nonlinear programs

Because of this, we resorted to algorithmic methods for solving these problems. For single variable problems, we used the Fibonacci Search, the Golden Section Search or Newton's method, while for multi-variable problems, we used the steepest descent algorithm, Newton's method or the BGFS algorithm.

It is even more difficult to solve the KKT criteria analytically to derive a stationary point and then verify the second order conditions for it to be a minimum when we have a constrained optimisation problem.

Penalty methods for nonlinear programs

It is also more difficult to derive algorithmic methods for constrained problems.

One way of circumventing this is to "convert" a constrained problem to an unconstrained problem. This is usually achieved via the addition of a *penalty function* to the objective function.

The idea is to remove the constraints, but add a large penalty to the objective function, which comes into play if the constraints are not satisfied. The hope is that any local minimum of the modified objective function will be such that the constraints are satisfied.

Penalty methods for nonlinear programs

In particular, let $f(x)$ be the objective function, and let $g_i(x) \leq 0$ be an inequality constraint where f and g_i are both C^1 functions.

Our aim is to replace the inequality constraint by a penalty term added to $f(x)$, such that:

1. The penalty term is 0 when $g_i(x) \leq 0$.
2. The penalty term increases in value when $g_i(x) \leq 0$ is violated, with larger increases as $g_i(x)$ gets larger.
3. The new objective function is C^1.

The l_2 penalty method

We start by illustrating the use of l_2-penalty methods for problems involving C^1 functions with an example.

Example 4.7

Consider

$$\min_x f(x) := x^2 - 2x \quad \text{subject to} \quad g(x) = x \leq 0. \tag{4.12}$$

Solve using a penalty method.

Note: The problem has a unique (global) minimum, $x^* = 0$.

Solution to Example 4.7

We convert the problem to an unconstrained problem by choosing a *penalty parameter* $\alpha > 0$, and define the *penalty function*

$$P_\alpha(x) \;:=\; x^2 - 2x + \tfrac{\alpha}{2}(x_+)^2,$$

where

$$x_+ \;:=\; \max\{x, 0\} \;=\; \begin{cases} x & \text{if } x > 0 \quad \text{(infeasible)}, \\ 0 & \text{if } x \leq 0 \quad \text{(feasible)}. \end{cases}$$

Solution to Example 4.7 – Continued

$P_\alpha(x)$ is the objective function $f(x)$ plus the penalty term $\frac{\alpha}{2}(x_+)^2$.

The penalty term measures the "cost" of infeasibility. The idea is that by letting α get large, in fact letting $\alpha \to \infty$, any infeasible point will become more costly than any feasible point, so that with "$\alpha = \infty$", minimising $P_\alpha(x)$ is the same as solving the original problem.

Solution to Example 4.7 - Continued

For $\alpha > 0$, we have

$$\begin{aligned} \nabla P_\alpha(x) &:= 2x - 2 + \alpha x_+ \\ &= 0 \quad \text{at a local minimum} \;\; x_\alpha. \end{aligned}$$

Suppose $x_\alpha \leq 0$; then $2x_\alpha - 2 = 0$, ie $x_\alpha = 1$, which is a contradiction.

Hence P_α is minimised at a point $x_\alpha > 0$, that satisfies

$$\begin{aligned} 0 &= \nabla P_\alpha(x_\alpha) = 2x - 2 + \alpha x \\ \Longrightarrow x &= 2/(2+\alpha). \end{aligned}$$

Thus $x_\alpha = 2/(2+\alpha) \to 0 = x^*$ as $\alpha \to \infty$.

The l_2 penalty method

This example can be generalised to problems of the form (NLP). For $\alpha > 0$, the l_2 penalty function for (NLP) is

$$P_\alpha(x) = f(x) + \frac{\alpha}{2}\left(\sum_i [g_i(x)_+]^2 + \sum_j h_j(x)^2\right) \qquad (4.13)$$

where

$$g_i(x)_+ := \max\{g_i(x), 0\} = \begin{cases} g_i(x) & \text{if } g_i(x) > 0 \quad \text{(infeasible)}, \\ 0 & \text{if } g_i(x) \le 0 \quad \text{(feasible)}. \end{cases}$$

The l_2 penalty method

To see why the name "l_2" is used, let $g(x)_+ := (g_1(x)_+, \ldots, g_p(x)_+)$, then

$$\sum_i [g_i(x)_+]^2 + \sum_j h_j(x)^2 \;=\; \|g(x)_+\|^2 + \|h(x)\|^2,$$

That is, the penalty term uses the square of the Euclidean or l_2 norm.

The l_2 penalty method

The following convergence theorem shows the usefulness of the penalty function approach.

Theorem 10 (Convergence of minima for penalty functions)

Let f, g and h be C^1 functions. Assume a constraint qualification on h and g holds at x^*.
Suppose x^k minimises P_{α_k} for each k, where $\alpha_k \to \infty$. If $\{x^k\}$ has a cluster point x^* and a constraint qualification holds at x^*, then x^* is (feasible and) stationary for (NLP).

Class Exercise 4.8

Consider the nonlinear program

$$\begin{aligned} \text{min} \quad & x_1^2 + x_2^2 + x_1 - x_2 \\ \text{s.t.} \quad & x_1 \geq 1 \\ & x_2 \geq 0. \end{aligned}$$

1. Write down the l_2-penalty function $P_{\alpha_k}(x)$ with penalty parameter $\alpha_k = k$.
2. Show that stationary points for $P_{\alpha_k}(x)$ only occur when $x_1 < 1$ and $x_2 > 0$.
3. Hence, find a stationary point $x^k = (x_1^k, x_2^k)$ for $P_{\alpha_k}(x)$. Write down the limit $x^* = \lim_{k \to \infty} x^k$.

Solution to Class Exercise 4.8

1. $P_{\alpha_k}(x) \;=\; x_1^2 + x_2^2 + x_1 - x_2 + \frac{k}{2}(1 - x_1)_+^2 + \frac{k}{2}(-x_2)_+^2.$
2. For the stationary points we first compute the grad:

$$\nabla P_{\alpha_k}(x) = \begin{bmatrix} 2x_1 + 1 - k(1 - x_1)_+ \\ 2x_2 - 1 - k(-x_2)_+ \end{bmatrix} = 0. \qquad (4.14)$$

If $x_1 \geq 1$ then the first line of (4.14) becomes $2x_1 + 1 = 0 \;\Rightarrow\; x_1 = 1/2$, giving a contradiction. Hence $x_1 < 1$.

If $x_2 \leq 0$ then the second line of (4.14) becomes $2x_2 - 1 + kx_2 = 0 \;\Rightarrow\; x_2 = 1/(k+2)$, giving a contradiction (for $k > 0$). Hence $x_2 > 0$.

Solution to Class Exercise 4.8 – Continued

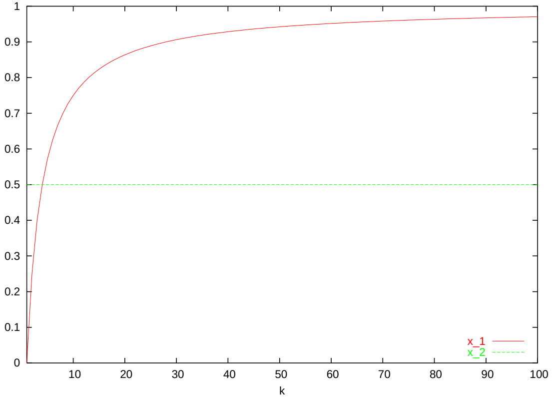
3. From Part 2 the grad at a stationary point is now:

$$\nabla P_{\alpha_k}(x) = \begin{bmatrix} 2x_1 + 1 - k(1 - x_1) \\ 2x_2 - 1 \end{bmatrix} = 0. \qquad (4.15)$$

By the first line of (4.15)
$2x_1^k + 1 - k(1 - x_1^k) = 0 \;\Rightarrow\; x_1^k = (k-1)/(k+2)$.

By the second line of (4.15) $2x_2^k - 1 = 0 \;\Rightarrow\; x_2^k = 1/2$.

As $k \to \infty$ we have $(x_1^k, x_2^k) \to (1, 1/2) = x^*$.


1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
10
20
30
40
50
60
70
80
90
100
k
x_1
x_2

The l_2 penalty method

Let's look at the first-order necessary condition for x^k to minimise P_α. Suppose f, g and h are C^2 functions.

Q: Is P_α C^1? Is it C^2?
A: From its definition, it can be shown that P_α is differentiable, and using the chain rule

$$\begin{aligned}\nabla P_\alpha(x) &= \nabla f(x) + \frac{\alpha}{2}\left(\sum_i \nabla_x [g_i(x)_+]^2 + \sum_j \nabla_x [h_j(x)]^2\right) \\ &= \nabla f(x) + \alpha\left(\sum_i (g_i(x)_+)\nabla g_i(x) + \sum_j h_j(x)\nabla h_j(x)\right)\end{aligned}$$

The l_2 penalty method

Hence P_α is continuously differentiable.

However P_α is not C^2 because it is not generally twice differentiable if $g_i(x) = 0$ for some i.

For example, for $x \in \mathbb{R}$, $(x_+)^2$ is C^1 but not C^2 at 0.

The l_2 penalty method

In the situation of the above theorem, suppose that $\{x^k\}$ converges to x^*.

By choice of x^k, we have $0 = \nabla P_{\alpha_k}(x^k)$.

Therefore if $\{\alpha_k g_i(x^k)_+\}_k$ is convergent for each i, that is $\{\alpha_k g(x^k)_+\}$ converges to some $\lambda^* \in \mathbb{R}^p$; and if $\{\alpha_k h_j(x^k)\}_k$ is convergent for each j, that is $\{\alpha_k h(x^k)\}$ converges to some $\eta^* \in \mathbb{R}^q$, then

The l_2 penalty method

$$\begin{aligned} 0 &= \nabla P_{\alpha_k}(x^k) \\ &= \nabla f(x^k) + \sum_i (\alpha_k g_i(x^k)_+) \nabla g_i(x^k) + \sum_j (\alpha_k h_j(x^k)) \nabla h_j(x^k) \\ &\to \nabla f(x^*) + \sum_i \lambda_i^* \nabla g_i(x^*) + \sum_j \eta_j^* \nabla h_j(x^*). \end{aligned}$$

This is KKTa!

In other words, just as x^k approximates an optimal solution x^* of the constrained problem, $\alpha_k g_i(x^k)_+$ and $\alpha_k h_j(x^k)$ approximate the respective optimal multipliers λ_i^* and η_j^*.

The l_2 penalty method

In fact we can go a little bit further. Given $\lambda^* = \lim_{k\to\infty} \alpha_k g(x^k)_+$ as above, the fact that $\alpha_k > 0$ and $g_i(x^k)_+ \geq 0$ by definition means that $\lambda_i^* \geq 0$. This is part of KKTb.

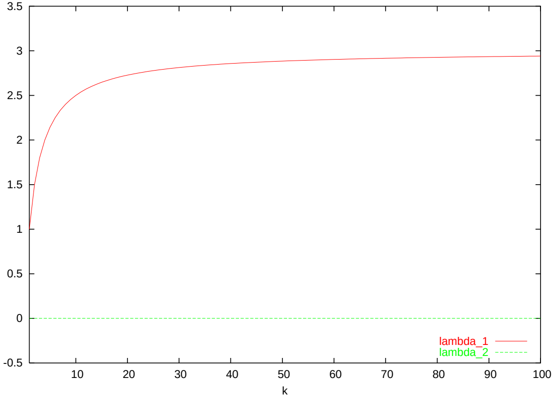
Also, if $x^k \to x^*$, and $g_i(x^*) < 0$, then for large enough k, $\alpha_k g_i(x^k)_+ = 0$ and therefore $\lambda_i^* = 0$. On the other hand if $\lambda_i^* > 0$ then $g_i(x^*)$ cannot possibly be strictly negative, and therefore $g_i(x^*) = 0$. This is another way of expressing $\lambda_i^* g_i(x^*) = 0$, which is also part of KKTb.

Extension of Class Exercise 4.8

$$\begin{aligned} \min \quad & x_1^2 + x_2^2 + x_1 - x_2 \\ \text{s.t.} \quad & x_1 \geq 1 \\ & x_2 \geq 0. \end{aligned}$$

1. Write down the l_2-penalty function $P_k(x)$ with penalty parameter $\alpha_k = k$.
2. Show that stationary points for $P_k(x)$ only occur when $x_1 < 1$ and $x_2 > 0$.
3. Hence, find a stationary point $x^k = (x_1^k, x_2^k)$ for $P_k(x)$. Write down the limit $x^* = \lim_{k \to \infty} x^k$.
4. Write down an estimate of λ^k, the optimal multiplier vector, and find the limit $\lambda^* = \lim_{k \to \infty} \lambda^k$.

$$
\begin{aligned}
\lambda_1^* &= \lim_{k\to\infty} \lambda_1^k = \lim_{k\to\infty} k g_1(x^k)_+ \\
&= \lim_{k\to\infty} k \left(1 - \frac{k-1}{k+2}\right)_+ \\
&= \lim_{k\to\infty} \frac{3k}{k+2} \\
&= 3 \\
\lambda_2^* &= \lim_{k\to\infty} \lambda_2^k = \lim_{k\to\infty} k g_2(x^k)_+ \\
&= \lim_{k\to\infty} k \left(-1/2\right)_+ \\
&= \lim_{k\to\infty} k \cdot 0 \\
&= 0.
\end{aligned}
$$



3.5
3
2.5
2
1.5
1
0.5
0
-0.5
10
20
30
40
50
60
70
80
90
100
k
lambda_1
lambda_2

Proof of Theorem 10

We next present a proof of the penalty function theorem in the case with only equality constraints. The case with inequality constraints as well is similar, but more technical.

So, the theorem we will prove is the following:

Theorem 10 (Equality Constrained Version)

Suppose we have an equality constrained NLP where f and h are C^1 functions, and a constraint qualification on h holds at x^*. Suppose x^k minimises P_{α_k} for each k, where $\alpha_k \to \infty$. If $\{x^k\}$ has a cluster point x^*, then x^* is (feasible and) stationary for (NLP).

Proof of Theorem 10

Suppose x' solves the non-linear program, so for all x such that $h(x) = 0$,

$$f(x') \leq f(x).$$

From the statement of the theorem, x^k minimises the penalty function $P_{\alpha_k}(x)$. Therefore we know that

$$P_{\alpha_k}(x^k) \leq P_{\alpha_k}(x').$$

Proof of Theorem 10 – Continued

Expanding this out gives

$$
\begin{aligned}
f(x^k) + \frac{\alpha_k}{2}\sum_{j=1}^{q}[h_j(x^k)]^2 &\leq f(x') + \frac{\alpha_k}{2}\sum_{j=1}^{q}[h_j(x')]^2 \\
&= f(x') \qquad (4.16)
\end{aligned}
$$

where (4.16) holds because x' is feasible for the NLP.

Rearranging gives

$$
\sum_{j=1}^{q}[h_j(x^k)]^2 \leq \frac{2}{\alpha_k}(f(x') - f(x^k)).
$$

Proof of Theorem 10 – Continued

If x^* is the limit point of x^k, then we can take the limit on both sides as $k \to \infty$ to get

$$\begin{aligned}\sum_{j=1}^{q}[h_j(x^*)]^2 &\leq \lim_{k\to\infty}\frac{2}{\alpha_k}(f(x') - f(x^k)) \\ &= 0 \end{aligned} \tag{4.17}$$

The limit above follows because $f(x') - f(x^k)$ is finite and $\alpha_k \to \infty$ as $k \to \infty$.

This tells us that $h_j(x^*) = 0$ for all j, and therefore x^* is feasible.

Proof of Theorem 10 – Continued

To prove that x^* is optimal, we note that from (4.16),

$$f(x^k) + \frac{\alpha_k}{2} \sum_{j=1}^{q} [h_j(x^k)]^2 \leq f(x').$$

Taking the limit as $k \to \infty$ gives us

$$f(x^*) \leq f(x^*) + \lim_{k\to\infty} \frac{\alpha_k}{2} \sum_{j=1}^{q} [h_j(x^k)]^2 \leq f(x').$$

Since x' is an optimal solution for the NLP, and x^* has an equal or lower objective function value, we conclude that x^* is also an optimal solution for the NLP.
QED

The l_2 penalty method

In practice, it is usually not possible to take the exact limit as $k \to \infty$, and so we would solve the unconstrained penalty problem by numerical methods. What we would normally do is to progressively set the penalty parameter higher and higher, and when the algorithm stops, take the nearest feasible point as a solution.

The l_2-penalty method is just one of the many types of penalty methods that we can apply to constrained problems, and like most other methods it has advantages and disadvantages.

One of the more obvious disadvantages is that the l_2 penalty function is not C^2, so we cannot apply Newton's method to solve the unconstrained penalty problem.
We can overcome this problem by using a smoother function.

Subsection 8

Log Barrier penalty methods

The log barrier penalty method

The *log barrier* penalty method does this by taking the penalty function

$$P_\alpha(x) = f(x) - \frac{1}{\alpha}\sum_i \log(-g_i(x)) + \frac{\alpha}{2}\sum_j [h_j(x)]^2.$$

For the logarithm to be defined, this requires that $-g_i(x) > 0$, which means $g_i(x) < 0$, i.e., the point is strictly feasible.

Now as $g_i(x) \to 0$, $-\frac{1}{\alpha}\log(-g_i(x)) \to \infty$, so the boundary of the feasible region acts as a barrier preventing the solution from going past it — hence the name!

The log barrier penalty method

The only problem with this is that this does not allow $g_i(x)$ to be 0, which we want, because as $g_i(x) \to 0$, $-\frac{1}{\alpha}\log(-g_i(x)) \to \infty$. However, as $\alpha \to \infty$, this term becomes smaller and smaller, so with "$\alpha = \infty$", we can have $g_i(x) = 0$.

But since all points leading up to the limit must have been feasible because of the barrier, the limit point must be feasible too!

It turns out that the log-barrier method has a similar convergence result to the l_2 penalty method theorem.

The log barrier penalty method

Theorem 11 (Convergence for Log barrier Method)

Let f, g and h be C^1. Suppose that x^* is a KKT point of the NLP which is a local minimum. Then there exists a sequence of points x^k such that x^k is a local minimum of $P_{\alpha_k}(x)$, and $x^k \to x^*$ as $k \to \infty$ (and therefore $\alpha_k \to \infty$).

As before, we can find estimates of the KKT multipliers by taking the limit of certain quantities in the log-barrier problem.

The log barrier penalty method – KKT Multipliers

$$
\begin{aligned}
\nabla P_\alpha(x) &= \nabla f(x) - \sum_i \frac{1}{\alpha}\nabla(\log(-g_i(x))) + \sum_j \frac{\alpha}{2}\nabla[h_j(x)]^2 \\
&= \nabla f(x) - \sum_i \frac{1}{\alpha g_i(x)}\nabla g_i(x) + \sum_j \alpha h_j(x)\nabla h_j(x) \\
&= 0
\end{aligned}
$$

when the penalty function is minimised.
By comparing with KKTa as before, we can see that if we let

$$
-\frac{1}{\alpha g_i(x^k)} \to \lambda_i^* \quad \text{and} \quad \alpha h_j(x^k) \to \eta_j^*
$$

as $k \to \infty$, then λ^* and η^* give the KKT multipliers.

The log barrier penalty method – KKT Multipliers

Again, we can look a bit closer to uncover some of KKTb. Because $\alpha > 0$ and $g_i(x^k) < 0$ (remember all the x^k are strictly feasible), we can say that $\lambda_i^* \geq 0$.

Furthermore, if $g_i(x^*) < 0$, the α term in the denominator of λ_i^* will go to ∞ and ensure that $\lambda_i^* = 0$. On the other hand, if $\lambda_i^* > 0$, then this means that $\alpha g_i(x^k)$ cannot tend to ∞, and the only way this can happen is if $g_i(x^k) \to 0$ — in other words, $g_i(x^*) = 0$. As before, this is encapsulated in $\lambda_i^* g_i(x^*) = 0$.

Class Exercise 4.9

$$\begin{aligned} \min \quad & x_1^2 + x_2^2 + x_1 - x_2 \\ \text{s.t.} \quad & x_1 \geq 1 \\ & x_2 \geq 0. \end{aligned}$$

1. Write down the log barrier penalty function $P_{\alpha_k}(x)$ with penalty parameter $\alpha_k = k$.
2. Write down $\nabla P_{\alpha_k}(x)$, and solve $\nabla P_{\alpha_k}(x) = 0$ to find a stationary point $x^k = (x_1^k, x_2^k)$ for $P_{\alpha_k}(x)$.
3. Find the limit $x^* = \lim\limits_{k \to \infty} x^k$.
4. Write down an estimate of λ^k, the optimal multiplier vector, and find the limit $\lambda^* = \lim\limits_{k \to \infty} \lambda^k$.

Solution to Class Exercise 4.9

1. $P_{\alpha_k}(x) \; = \; x_1^2 + x_2^2 + x_1 - x_2 - \frac{1}{k}\log(x_1 - 1) - \frac{1}{k}\log(x_2)$.
2. We first compute the grad:

$$\nabla P_{\alpha_k}(x) = \begin{bmatrix} 2x_1 + 1 + 1/(k(1-x_1)) \\ 2x_2 - 1 + 1/(kx_2) \end{bmatrix} = 0 \qquad (4.18)$$

At a stationary point the first line of (4.18) becomes $2kx_1^2 - kx_1 - (k+1) = 0$, which has roots $x_1^k = (k \pm \sqrt{k^2 + 8k(k+1)})/4k$.

At a stationary point the second line of (4.18) becomes $2kx_2^2 - kx_2 - 1 = 0$, which has roots $x_2^k = (k \pm \sqrt{k^2 + 8k})/4k$.

For feasibility we choose the *positive* root in each case.

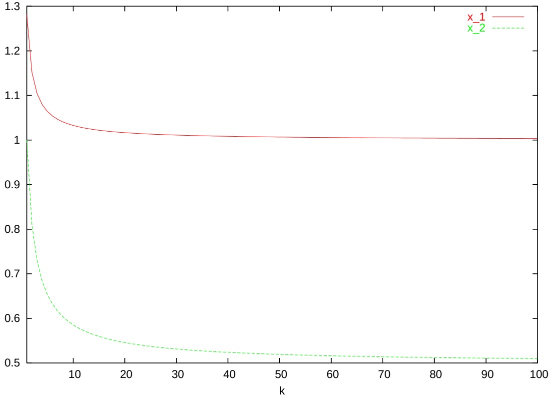
Solution to Class Exercise 4.9 – Continued

3. From Part 2 we have:

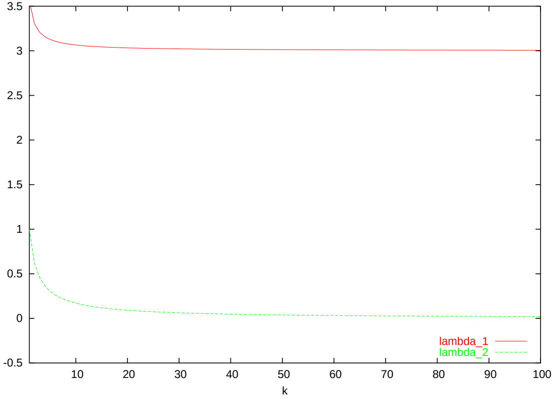
$$
\begin{aligned}
x^* &= \lim_{k\to\infty}((k+\sqrt{k^2+8k(k+1)})/4k, (k+\sqrt{k^2+8k})/4k) \\
&= \lim_{k\to\infty}((1+\sqrt{1+8+8/k})/4, (1+\sqrt{1+8/k})/4) \\
&= (1, 1/2)
\end{aligned}
$$

4. For the KKT multipliers λ^*:

$$
\begin{aligned}
\lambda^* &= \lim_{k\to\infty}(-1/(k(1-x_1)), -1/(-kx_2)) \\
&= \lim_{k\to\infty}(4/(-3k+\sqrt{k^2+8k(k+1)}), 4/(k+\sqrt{k^2+8k})) \\
&= (3, 0)
\end{aligned}
$$



x_1
x_2
1.3
1.2
1.1
1
0.9
0.8
0.7
0.6
0.5
10
20
30
40
50
60
70
80
90
100
k



3.5
3
2.5
2
1.5
1
0.5
0
-0.5
10
20
30
40
50
60
70
80
90
100
k
lambda_1
lambda_2

Subsection 9

Exact penalty methods

Exact Penalty Methods

A disadvantage of smooth penalty methods like the l_2 method and the log-barrier method is that the penalty parameter α_k must diverge to ∞, which is hard to replicate in a numerical-methods setting. Furthermore, even setting α to be very large but finite has its own difficulties, because when we do this, rounding errors can be a problem.

We can get around this by use of *exact penalty methods*. The idea behind this is that instead of finding the local minimum only as $\alpha_k \to \infty$, we find it whenever $\alpha \geq \bar{\alpha}$ for some $\bar{\alpha}$.

Exact Penalty Methods

More formally, we choose $P_\alpha(x)$ so that for sufficiently large α, any local minimum of the non-linear program is also a local minimum of $P_\alpha(x)$, and any feasible point of the non-linear program that is a local minimum of $P_\alpha(x)$ is also a local minimum of the NLP.

A well-known exact penalty function is the l_1 penalty function:

$$P_\alpha(x) = f(x) + \alpha(\|g(x)_+\|_1 + \|h(x)\|_1),$$

where

$$\|y\|_1 = \sum_{i=1}^{p} |y_i|.$$

Exact Penalty Methods

This is analogous to the l_2 penalty method, except we replace the 2-norm by the 1-norm. The advantage to this method is that we only need to minimise the penalty function once, if we have set α to be high enough.

However, there is a big disadvantage to this method — $P_\alpha(x)$ is not even C^1, let alone C^2! This means that any unconstrained methods requiring the gradient will not (necessarily) work. This includes all of the methods that we have studied in this course.

Subsection 10

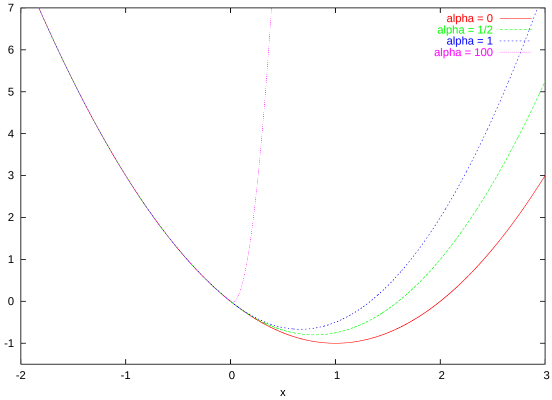
Comparison of penalty methods

Comparison of Penalty Methods

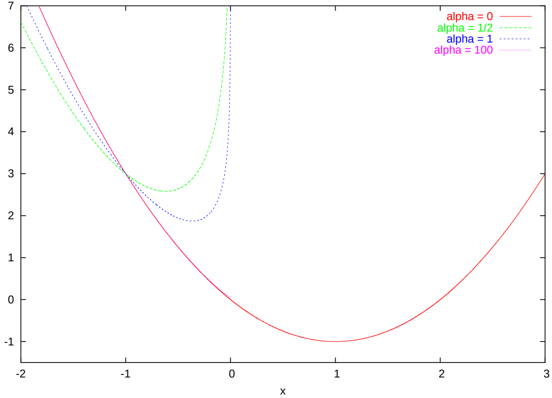
To compare the effects of various penalties, we go back to an earlier example:

$$\min f(x) = x^2 - 2x \text{ s.t. } x \leq 0.$$

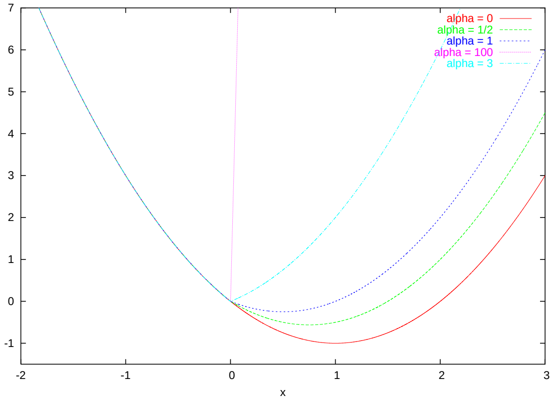
Observe how, as α increases, all of the penalty functions start to look like $f(x)$ for $x \leq 0$ and ∞ for $x > 0$. This is what we want!



$$P_\alpha(x) = x^2 - 2x + \frac{\alpha}{2}(x_+)^2$$



$$P_\alpha(x) = x^2 - 2x + \frac{1}{\alpha}\log(-x)$$



$$P_\alpha(x) = x^2 - 2x + \alpha|x_+|$$

Section 5

Convex Optimisation

- Stationary points and global minima
- Convex duality
- Lagrangian duality
- Wolfe duality

Convex Nonlinear Programs

In the next section we look at some of the further properties of NLPs when they are convex.

Recall that a function $f(x)$ is *convex* if for all $\alpha \in [0, 1]$,

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

A non-linear program

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

is defined to be a *convex program* if the functions $f(x)$ and $g_i(x)$ are convex functions, and $h(x)$ is affine.

Convex Nonlinear Programs

Note that the reason for restricting $h(x)$ to be affine can be seen by separating the equality constraint $h(x) = 0$ into two inequality constraints, $h(x) \leq 0$ and $-h(x) \leq 0$.

The only way that both $h(x)$ and $-h(x)$ can be convex is if for all α,

$$h(\alpha x + (1 - \alpha)y) = \alpha h(x) + (1 - \alpha)h(y).$$

This only happens when $h(x)$ is affine.

Convex Nonlinear Programs

We already know that if a non-linear program is a convex program, then it enjoys some useful properties. In particular, any stationary point of a convex program is a global minimum (and therefore the solution to the program).

In previous sections, we have seen that this applies under various restrictions as well — when there are no constraints we just want $f(x)$ to be convex, and when we have equality-only constraints we want f to be convex and h to be affine.

We will show that there exists a useful duality framework for convex programs, much like linear programming duality.

Properties of Convex Functions

Lemma (Convex Functions)

Let $f : \mathbb{R}^n \to \mathbb{R}$.

1. If f is C^1 then f is convex if and only if $\forall x, y \in \mathbb{R}^n$,
$$f(x) + \nabla f(x)^T (y - x) \leq f(y).$$
2. If f is C^2 then f is convex if and only if the Hessian $\nabla^2 f(x)$ is positive semidefinite for all x.
3. If $f_i : \mathbb{R}^n \to \mathbb{R}$ is convex and $\lambda_i \geq 0$, then $g(x) = \sum_i \lambda_i f_i(x)$ is a convex function.

Properties of Convex Functions

Proof of Lemma.

1. If f is convex then it lies underneath the line joining $(x, f(x))$ to $(y, f(y))$. In particular, the directional derivative of f in the direction $y - x$ must be less than the gradient of this line. In other words,

$$\begin{aligned} & \langle \nabla f(x), \frac{y-x}{\|y-x\|} \rangle \leq \frac{f(y)-f(x)}{\|y-x\|} \\ \Leftrightarrow \quad & \nabla f(x)^T (y-x) \leq f(y) - f(x) \\ \Leftrightarrow \quad & f(x) + \nabla f(x)^T (y-x) \leq f(y). \end{aligned}$$

This gives the required implication in one direction.

Proof of Lemma (1) – continued

For the reverse implication, if the inequality holds then

$$\langle \nabla f(x), \frac{y-x}{\|y-x\|} \rangle \leq \frac{f(y)-f(x)}{\|y-x\|}.$$

Therefore the directional derivative of f at x in the direction of $y-x$ is less than the gradient of the line joining $(x, f(x))$ to $(y, f(y))$. So initially $f(x)$ will lie underneath that line.

Proof of Lemma (1) – continued

If at some point $f(x)$ crosses that line (at z say), then the directional derivative of f at z in the direction of $y - z$ must be greater than the gradient of the line joining $(z, f(z))$ to $(y, f(y))$. But this contradicts

$$\langle \nabla f(z), \frac{y - z}{\|y - z\|} \rangle \leq \frac{f(y) - f(z)}{\|y - z\|},$$

which we derive by switching x for z. Therefore f stays below the line joining $(x, f(x))$ to $(y, f(y))$ and it is convex.

Proof of Lemma – continued

2. This is a standard result (see Notes p.9).
3. For any $\alpha \in [0, 1]$,

$$
\begin{aligned}
g(\alpha x + (1-\alpha)y) &= \sum_i \lambda_i f_i(\alpha x + (1-\alpha)y) \\
&\leq \sum_i \lambda_i \left[\alpha f_i(x) + (1-\alpha) f_i(y)\right] \\
&= \alpha \sum_i \lambda_i f_i(x) + (1-\alpha) \sum_i \lambda_i f_i(y) \\
&= \alpha g(x) + (1-\alpha) g(y).
\end{aligned}
$$

Hence $g(x)$ is convex.
QED

Properties of Convex Functions

Part (1) of the Lemma states that if you 'linearize' a function f by replacing it with the first-order Taylor series approximation,

$$f(y) \approx f(x) + \nabla f(x)^T (y - x),$$

then the linearization lies underneath the function f if and only if f is convex.

Properties of Convex Functions

In fact, this also applies to part (2) of the Lemma. Replacing f by its second-order Taylor series approximation, we get

$$
\begin{aligned}
f(y) &\approx f(x) + \nabla f(x)^T (y - x) + \frac{1}{2}(y - x)^T \nabla^2 f(x)(y - x), \\
&\geq f(x) + \nabla f(x)^T (y - x)
\end{aligned}
$$

if $\nabla^2 f(x)$ is positive semidefinite.

Again, this approximation lies underneath the function f if and only if f is convex (from the definition of positive semi-definiteness).

Properties of Convex Functions

An easy corollary of the Lemma applies to quadratic functions:

Corollary (Convexity of quadratic function)

A quadratic function $\frac{1}{2}x^T Ax + b^T x + c$, where $A \in \mathbb{R}^{n \times n}$ is symmetric, is convex if and only if A is positive semidefinite.

Proof. This is easy to prove — direct differentiation shows that A is the Hessian of the function.

Class Exercise 5.1

Using the previous corollary determine whether or not the following quadratic function is convex.

$$\frac{3}{2}x_1^2 - 4x_1x_2 + 3x_2^2 - 12x_2 + 2$$

Subsection 1

Stationary points and global minima

Stationary Points and Global Minima

The next result says that if a constraint qualification holds then stationary points of a convex program are equivalent to global minimizers.

Theorem (Global minima for convex programs)

Suppose (NLP) is a convex program.

If x^* is a stationary point of (NLP), i.e., satisfies the KKT conditions, then x^* is a global minimizer.

Conversely, if x^* is a local or global minimizer of (NLP) and a constraint qualification holds then x^* is also stationary.

Proof of Global Minima Theorem

The "converse" part is immediate from the basic theorem on stationarity, Theorem 7.
We now show that stationary points are global minimisers.
Suppose we have the general non-linear program

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

and suppose we know that it is a convex program. Now let x^* be a KKT point of the program, with multipliers λ^* and η^*. (We often say that (x^*, λ^*, η^*) is the KKT point.)

Proof of Global Minima Theorem – Continued

Define the function

$$\phi(x) = L(x, \lambda^*, \eta^*) = f(x) + \sum_i \lambda_i^* g_i(x) + \sum_j \eta_j^* h_j(x).$$

This differs from the Lagrangian because we fix the KKT multipliers λ^* and η^*.

Because f and g are convex and $\lambda_i^* \geq 0$, and h is affine (so both h and $-h$ are convex), we know that $\phi(x)$ is a convex function from the above Lemma.

Proof of Global Minima Theorem – Continued

So for any feasible point x,

$$\begin{aligned} f(x) &\geq L(x, \lambda^*, \eta^*) = \phi(x) \quad \text{since } x \text{ is feasible} \\ &\geq \phi(x^*) + \nabla\phi(x^*)^T(x - x^*) \quad \text{from the Lemma} \\ &= \phi(x^*) \quad \text{since } \nabla\phi(x^*) = \nabla_x L(x^*, \lambda^*, \eta^*) = 0 \text{ from KKTa} \\ &= L(x^*, \lambda^*, \eta^*) = f(x^*) \quad \text{from KKTb and KKTc.} \end{aligned}$$

Therefore x^* is a global minimiser of the non-linear program.
QED

Stationary Points and Global Minima

From this proof, we have the following corollary.

Corollary (Minimum of the Lagrangian)

If (x^*, λ^*, η^*) is a KKT point of a convex program, then the Lagrange function $L(x, \lambda^*, \eta^*)$, when considered as a function of x only, is also minimised at $x = x^*$.

Stationary Points and Global Minima

We illustrate this result with an example.

Example 5.2

Consider the nonlinear program

$$\begin{aligned} \min \quad & x_1^2 + x_2^2 \\ \text{s.t.} \quad & -x_1 - x_2 + 4 \leq 0 \\ & x_1, x_2 \geq 0. \end{aligned}$$

Verify the previous corollary (Minimum of the Lagrangian).

Minimum of the Lagrangian for Example 5.2

It can be seen by inspection that the NLP has a unique KKT point $x^* = (2, 2)$ with multipliers $\lambda^* = (4, 0, 0)$.

Now

$$\begin{aligned}
\phi(x) &= L(x, \lambda^*) \\
&= x_1^2 + x_2^2 + 4(-x_1 - x_2 + 4) + 0(-x_1) + 0(-x_2) \\
&= x_1^2 + x_2^2 - 4x_1 - 4x_2 + 16.
\end{aligned}$$

To find the minimum of $\phi(x)$ over all x, we simply take $\nabla\phi(x) = 0$.

$$\nabla\phi(x) = (2x_1 - 4, 2x_2 - 4) = (0, 0).$$

This is solved by $x = (2, 2)$.

Minimum of the Lagrangian for Example 5.2

Checking that $x = (2, 2)$ is in fact a minimum:

$$\nabla^2 \phi(x) = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$

which is obviously positive definite.

Therefore $(2, 2)$ minimises $L(x, \lambda^*)$, and we see that it is indeed equal to x^*, as predicted by the previous corollary.

Stationary Points and Global Minima

Given the previous result, we can ask: if we want to solve a constrained problem by converting it to an unconstrained problem, why can't we just minimise the Lagrangian as an unconstrained problem?

We could — if we knew the optimal KKT multipliers! But we usually don't.

Subsection 2

Convex duality

The Lagrangian Saddlepoint

We have seen that given specific KKT multipliers λ^* and η^*, we can consider the Lagrangian as a function of x alone. In the next theorem, we also do the opposite: given a specific point x^*, we can consider the Lagrangian as a function of λ and η!

Theorem 12 (Lagrangian Saddlepoint)

The triple (x^*, λ^*, η^*) is a KKT point of a convex program if and only if, for all $\lambda \geq 0$ and all x and η,

$$L(x^*, \lambda, \eta) \leq L(x^*, \lambda^*, \eta^*) \leq L(x, \lambda^*, \eta^*).$$

The Lagrangian Saddlepoint

This inequality is known as the Saddle Inequality and the theorem as the Saddlepoint Theorem.

The Saddlepoint theorem indicates that, just as the optimal value $f(x^*)$ of a convex nonlinear program can be written as the minimum, with respect to x, of the Lagrangian $L(x, \lambda^*, \eta^*)$, it is also the maximum, with respect to $\lambda \geq 0$ and η, of $L(x^*, \lambda, \eta)$. This will be explored later as Lagrangian duality.

The Lagrangian Saddlepoint

To better understand the implications of this Theorem, we verify it using Example 5.2:

$$\begin{aligned} \min \quad & x_1^2 + x_2^2 \\ \text{s.t.} \quad & -x_1 - x_2 + 4 \leq 0 \\ & x_1, x_2 \geq 0. \end{aligned}$$

The Lagrangian is

$$L(x, \lambda) = x_1^2 + x_2^2 + \lambda_1(-x_1 - x_2 + 4) + \lambda_2(-x_1) + \lambda_3(-x_2).$$

The Lagrangian Saddlepoint for Example 5.2

We have already seen that there is a KKT point at $x^* = (2, 2), \lambda^* = (4, 0, 0)$, and furthermore, $(2, 2)$ is a minimiser for the unconstrained function $L(x, (4, 0, 0))$.

Now consider the Lagrangian at x^*, considered as a function of λ only:

$$L(x^*, \lambda) = 8 - 2\lambda_2 - 2\lambda_3.$$

Maximising this over all $\lambda_1, \lambda_2, \lambda_3 \geq 0$ is easy: clearly $\lambda_2 = \lambda_3 = 0$, and λ_1 can take any value.

The Lagrangian Saddlepoint for Example 5.2

This does not give us a value for λ_1, so we cannot solve for the KKT multipliers from this problem alone. However we note that for any value of λ_1, $L(x^*, \lambda)$ does not change, so it follows that

$$L(x^*, \lambda) \leq L(x^*, (4, 0, 0)),$$

i.e., the Saddle Inequality holds.

Note, from the Theorem, the fact that we have shown both sides of the Saddle Inequality holds actually proves that $((2, 2), (4, 0, 0))$ is a KKT point!

Proof of Theorem 12

We now prove the Lagrangian Saddlepoint Theorem.

Theorem 12 (Lagrangian Saddlepoint)

The triple (x^*, λ^*, η^*) is a KKT point of a convex program if and only if, for all $\lambda \geq 0$ and all x and η,

$$L(x^*, \lambda, \eta) \leq L(x^*, \lambda^*, \eta^*) \leq L(x, \lambda^*, \eta^*).$$

Proof of Theorem 12

Part 1 ($\Rightarrow$): Consider the forward implication: *if* (x^*, λ^*, η^*) *is a KKT point, then the Saddle Inequality holds*.
The second inequality is just the Corollary above, so we already know it to be true.

Note that $L(x^*, \lambda^*, \eta^*) = f(x^*) + (\lambda^*)^T g(x^*) + (\eta^*)^T h(x^*)$.

Since (x^*, λ^*, η^*) is a KKT point, from KKTb and KKTc we know that

$$(\lambda^*)^T g(x^*) = 0$$

and

$$(\eta^*)^T h(x^*) = 0.$$

Hence,

$$L(x^*, \lambda^*, \eta^*) = f(x^*). \tag{5.1}$$

Proof of Theorem 12 – Continued

Now consider any $\lambda \geq 0$ and any η. Since $g(x^*) \leq 0$ by KKTb, we have:

$$\lambda^T g(x^*) \leq 0$$

and from KKTc we have:

$$\eta^T h(x^*) = 0.$$

Therefore

$$\begin{aligned} L(x^*, \lambda, \eta) &= f(x^*) + \lambda^T g(x^*) + \eta^T h(x^*) \\ &\leq f(x^*) \\ &= L(x^*, \lambda^*, \eta^*) \quad \text{by (5.1).} \end{aligned}$$

This is the left inequality of the theorem, completing Part (1).

Proof of Theorem 12 – Continued

Part 2 ($\Leftarrow$): Consider the backward implication: *if the Saddle Inequality holds then* (x^*, λ^*, η^*) *is a KKT point.*
We must show that the Saddle Inequality implies that all the KKT conditions hold for (x^*, λ^*, η^*).

The right-hand side of the Saddle Inequality says that x^* minimises $L(x, \lambda^*, \eta^*)$ over all x. Therefore, x^* must be a stationary point of $L(x, \lambda^*, \eta^*)$, considered as a function of x, and therefore at $x = x^*$ we have

$$\nabla_x L(x, \lambda^*, \eta^*) = 0.$$

This is KKTa.

Proof of Theorem 12 – Continued

The left-hand side of the Saddle Inequality says that (λ^*, η^*) maximises $L(x^*, \lambda, \eta)$ (considered as a function of (λ, η)) over all $\lambda \geq 0$ and all η. If you consider the KKT multipliers as variables themselves, we see that this is another optimisation problem:

$$\begin{aligned} \min_{\lambda,\ \eta} \quad & -L(x^*, \lambda, \eta) \\ \text{s.t.} \quad & \lambda \geq 0. \end{aligned}$$

We denote this problem by (P).

Proof of Theorem 12 – Continued

This problem must also have a Lagrange function, which we denote by $L^P(\lambda, \eta, \alpha)$, where α is the vector of KKT multipliers:

$$\begin{aligned} L^P(\lambda, \eta, \alpha) &= -L(x^*, \lambda, \eta) + \alpha^T(-\lambda) \\ &= -L(x^*, \lambda, \eta) - \alpha^T \lambda. \end{aligned} \tag{5.2}$$

Since (λ^*, η^*) solves (P), there must exist a KKT multiplier α^* such that the KKT conditions for (P) hold for $(\lambda^*, \eta^*, \alpha^*)$. Condition KKTa for (P) tells us that

$$\nabla_\lambda L^P(\lambda^*, \eta^*, \alpha^*) = 0 \tag{5.3}$$

and

$$\nabla_\eta L^P(\lambda^*, \eta^*, \alpha^*) = 0. \tag{5.4}$$

Proof of Theorem 12 – Continued

Also, Condition KKTb for (P) gives us

$$-\lambda^* \leq 0,\ \alpha^* \geq 0,\ (\alpha^*)^T \lambda^* = 0. \tag{5.5}$$

Now from the KKTa condition of (P) [Eq. (5.3)],

$$\begin{aligned}\nabla_\lambda L^P(\lambda^*, \eta^*, \alpha^*) &= -\nabla_\lambda L(x^*, \lambda^*, \eta^*) - \alpha^* \\ &= -g(x^*) - \alpha^* \\ &= 0\end{aligned} \tag{5.6}$$

(where the first equality follows from Eq. (5.2)).

Therefore

$$g(x^*) = -\alpha^* \leq 0 \qquad \text{by Eq. (5.5)}$$

which is the first KKTb condition for the original program.

Proof of Theorem 12 – Continued

Also, by Eq. (5.6), we have

$$(\alpha^*)^T \lambda^* = -(g(x^*))^T \lambda^* = -(\lambda^*)^T g(x^*) = 0$$

which is the third KKTb condition.

The second KKTb condition is clearly fulfilled by the first part of Eq. (5.5), since

$$-\lambda^* \leq 0 \quad \Rightarrow \quad \lambda^* \geq 0.$$

Proof of Theorem 12 – Continued

All that remains is to show that the last KKT condition holds. From the KKTa condition of (P) [Eq. (5.4)],

$$\nabla_\eta L^P(\lambda^*, \eta^*, \alpha^*) = -\nabla_\eta L(x^*, \lambda^*, \eta^*) = -h(x^*) = 0$$

(where the first equality follows from Eq. (5.2)), and so KKTc holds true too, for the original NLP.

We have seen that if the Saddle Inequality holds at a point (x^*, λ^*, η^*), then all the KKT conditions hold for that triple. Thus (x^*, λ^*, η^*) is a KKT point and the theorem is proved.

QED

The Lagrangian Saddlepoint

Example 5.3

$$\begin{array}{ll} \min & x_1^2 + x_2 \\ \text{s.t.} & x_2 \geq 0. \end{array}$$

Use the Saddle Inequality to find the KKT points for this NLP.

Solution to Example 5.3

The Lagrangian is

$$L(x_1, x_2, \lambda) = x_1^2 + x_2 + \lambda(-x_2) = x_1^2 + (1 - \lambda)x_2.$$

We look for all possible values which can solve the Saddle Inequality.

If λ^* has any other value besides 1, minimising the Lagrangian with respect to x_1 and x_2 will be an unbounded problem. So for the Saddle Inequality to hold, we need $\lambda^* = 1$.

Solution to Example 5.3 – Continued

Now we know that $L(x_1, x_2, \lambda^*) = x_1^2$. This clearly has a minimum at $x_1^* = 0$.

We also know that $\lambda^* = 1$ is the maximum of $L(x^*, \lambda)$, which is a linear function of λ, subject to $\lambda \geq 0$. The only way this can occur is if the coefficient of λ is itself 0 (note that this isn't the case if $\lambda^* = 0$). Therefore $-x_2 = 0$, which means $x_2^* = 0$.

Solution to Example 5.3 – Continued

Therefore the only point which satisfies the Saddle Inequality is $((0, 0), 1)$. From the Theorem, this is a KKT point, and also is the only KKT point.

Considering that solving the Saddle Inequality for even such a simple nonlinear program requires a couple of deductive reasonings (which would be hard to do in a general case), it is obvious that solving the Saddle Inequality directly is not a very good way to find the KKT points.

However, it does lead on to some interesting results, which make things much easier!

The Slater constraint qualification

We now note another constraint qualification that can be applied to the KKT conditions to be sure that they hold.

Recall that one of the constraint qualifications that we already know about requires $g(x)$ and $h(x)$ to be affine. For a convex program, this can be relaxed to the following

Definition: The *Slater constraint qualification* requires that the program be a convex program (in particular, $g(x)$ convex and $h(x)$ affine), and that there must exist a strictly feasible point, i.e., a point x' such that $h(x') = 0$ and $g(x') < 0$.

The Slater constraint qualification

Let's take a closer look at the Slater constraint qualification. Since $h(x)$ is affine, we can express it as

$$h(x) = Ax + a$$

where A is a $q \times n$ matrix and a is a vector of length q (q being the number of equality constraints).

In that case, for any point x, set $d = x' - x$. We know that

$$\nabla h(x)^T d = A(x' - x) = -a + a = 0$$

since both x and x' are feasible.

Now consider any constraint i that is active for x, i.e., $i \in I(x)$. We know that x lies on the line $g_i(x) = 0$. Because $g_i(x)$ is convex, the vector joining x to x' must lie entirely in the feasible region. This means that it is a descent direction for $g_i(x)$, i.e.,

$$\nabla g_i(x)^T d = \nabla g_i(x)^T (x' - x) < 0.$$

The Slater constraint qualification

These two conditions are just stating that the Mangasarian-Fromovitz qualification holds at every point. While this is not necessary for the Mangasarian-Fromovitz to hold (we just need it to hold at the supposed KKT point), it is certainly sufficient. Also the Slater qualification is easier to check!

This is just another demonstration of how having a convex program makes life easier for us.

Subsection 3

Lagrangian duality

Lagrangian duality

In the proof of the Lagrangian saddlepoint theorem, we noted that maximising the Lagrangian with respect to λ and η at the optimal x^* was equivalent to the following non-linear program:

$$\begin{array}{ll} \max\limits_{\lambda,\ \eta} & L(x^*, \lambda, \eta) \\ \text{s.t.} & \lambda \geq 0. \end{array}$$

We note that the objective function of this problem can be re-written as

$$L(x^*, \lambda, \eta) = f(x^*) + g(x^*)^T \lambda + h(x^*)^T \eta.$$

Lagrangian duality

Now we want to maximise this quantity over all $\lambda \geq 0$ and all η. But the only time that λ appears in this quantity is in the second term, and the only time that η appears in is the third term. Therefore we can write the problem as

$$\max_{\lambda \geq 0, \eta} L(x^*, \lambda, \eta) = f(x^*) + \max_{\lambda \geq 0} g(x^*)^T \lambda + \max_{\eta} h(x^*)^T \eta.$$

If we can solve this, then we will know the optimal KKT multipliers. The problem, as pointed out before, is that we do not know x^*, and thus cannot solve it directly.

Lagrangian duality

Let us look at the problem when we do not know x^*, so we replace x^* by any given x.

The third term is $\max_\eta h(x)^T\eta$, which asks us to maximise a linear function in η. As we saw in previous examples, this is an unbounded problem unless the coefficient of η is 0, i.e., $h(x) = 0$.

Lagrangian duality

The second term is $\max_{\lambda \geq 0} g(x)^T \lambda$, which again is a linear function. However, this time we have the constraint $\lambda \geq 0$. This will be an unbounded problem if any of the λ_i's have a positive coefficient.

Since the coefficient of λ_i is $g_i(x)$, the only way this problem will be bounded is if $g_i(x) \leq 0$ for all i, i.e., if $g(x) \leq 0$. In this case, the optimum is at $\lambda = 0$ and so $g(x)^T \lambda = 0$.

Lagrangian duality

We see from the above that the program is unbounded if and only if x is an infeasible point. Therefore we write the program as a function of x:

$$\phi(x) = \max_{\lambda \geq 0, \eta} L(x, \lambda, \eta) = \begin{cases} f(x) & \text{if } x \text{ is feasible for (NLP)} \\ \infty & \text{otherwise.} \end{cases}$$

This is actually a penalty function!

Lagrangian duality

In fact, looking closer reveals that it is the *ideal* penalty function: when x is feasible, it takes the value of $f(x)$, and when x is infeasible, it is ∞, which excludes any minimiser from choosing an infeasible point.

Therefore, we can solve the original non-linear program by minimising this function, in an unconstrained manner:

$$\min_x \phi(x) = \min_x \max_{\lambda \geq 0, \eta} L(x, \lambda, \eta).$$

Lagrangian duality

In other words, we can maximise the Lagrangian with respect to λ and η (ensuring the left-hand side of the Saddle Inequality), while keeping x arbitrary, and then minimise the maximum with respect to x (ensuring the right-hand side of the Saddle Inequality) to obtain the solution.

This gives us another idea: we can do this in reverse! We can minimise the Lagrangian with respect to x, keeping the multipliers arbitrary, and then maximise the minimum with respect to λ and η to obtain the solution.

Lagrangian duality

In other words, we exchange the minimum and the maximum:

$$\min_{x} \max_{\lambda \geq 0, \eta} L(x, \lambda, \eta) = \max_{\lambda \geq 0, \eta} \min_{x} L(x, \lambda, \eta) = \max_{\lambda \geq 0, \eta} \psi(\lambda, \eta)$$

where the last equality is the definition of $\psi(\lambda, \eta)$.

Lagrangian duality

This gives us two new problems which, ideally, should have the same solution as the original problem — the penalty formulation of the original NLP,

$$\min_{x} \phi(x),$$

and the *Lagrangian dual* problem,

$$\max_{\lambda \geq 0, \eta} \psi(\lambda, \eta).$$

Lagrangian duality – Example

We now examine this formulation with respect to Example 5.2 from the previous lecture:

Example 5.4

For the following NLP

$$\begin{array}{ll} \min & x_1^2 + x_2^2 \\ \text{s.t.} & -x_1 - x_2 + 4 \leq 0 \\ & x_1, x_2 \geq 0, \end{array}$$

find the unconstrained penalty formulation of this problem and determine the Lagrangian dual problem, where the objective function is written as an algebraic expression.

Solution to Example 5.4

First we look at the penalty formulation of the primal problem:

$$\begin{aligned}
\phi(x) &= \max_{\lambda\geq 0,\eta} L(x,\lambda,\eta) \\
&= \max_{\lambda\geq 0} x_1^2 + x_2^2 + \lambda_1(-x_1 - x_2 + 4) + \lambda_2(-x_1) + \lambda_3(-x_2) \\
&= x_1^2 + x_2^2 + \max_{\lambda_1\geq 0}(-x_1 - x_2 + 4)\lambda_1 + \max_{\lambda_2\geq 0} -x_1\lambda_2 + \max_{\lambda_3\geq 0} -x_2\lambda_3.
\end{aligned}$$

If any of the constraints are violated, this is an unbounded problem, so $\phi(x) = \infty$ for any infeasible point. Apart from that, however, it is not immediately obvious how to solve the maximisation problem to express $\phi(x)$ in terms of x only (so that we can apply a unconstrained minimiser).

Solution to Example 5.4 – Continued

Now consider the Lagrangian dual:

$$
\begin{aligned}
\psi(\lambda) &= \min_x L(x, \lambda, \eta) \\
&= \min_x (x_1^2 + x_2^2 + \lambda_1(-x_1 - x_2 + 4) + \lambda_2(-x_1) + \lambda_3(-x_2)) \\
&= \min_x (4\lambda_1 + (x_1^2 - (\lambda_1 + \lambda_2)x_1) + (x_2^2 - (\lambda_1 + \lambda_3)x_2)) \\
&= 4\lambda_1 + \min_{x_1}(x_1^2 - (\lambda_1 + \lambda_2)x_1) + \min_{x_2}(x_2^2 - (\lambda_1 + \lambda_3)x_2).
\end{aligned}
$$

Because this is not a linear function in x, we can actually solve it.

Solution to Example 5.4 – Continued

We note that for any a, the function $y^2 - ay$ is a quadratic function, and its minimum occurs at the turning point $y = \frac{a}{2}$. Therefore its minimum value is $\left(\frac{a}{2}\right)^2 - a\frac{a}{2} = -\frac{1}{4}a^2$. We substitute this in the above to derive

$$\psi(\lambda) = 4\lambda_1 - \frac{1}{4}(\lambda_1 + \lambda_2)^2 - \frac{1}{4}(\lambda_1 + \lambda_3)^2.$$

Solution to Example 5.4 – Continued

Therefore the Lagrangian dual problem is

$$\begin{array}{ll} \max & 4\lambda_1 - \frac{1}{4}(\lambda_1 + \lambda_2)^2 - \frac{1}{4}(\lambda_1 + \lambda_3)^2 \\ \text{s.t.} & \lambda \geq 0. \end{array}$$

This is not an unconstrained problem (so we cannot apply unconstrained methods), but it does have very simple constraints, making it easy to solve. The solution can be seen to be $\lambda^* = (4, 0, 0)$, our optimal KKT multipliers.

Notes on Example 5.4

Furthermore, we note that at the point $\lambda^* = (4, 0, 0)$, the Lagrangian dual problem has the optimal function value

$$4 \times 4 - \frac{1}{4}(4+0)^2 - \frac{1}{4}(4+0)^2 = 8,$$

and at the point $x^* = (2, 2)$, the original program has the optimal function value

$$2^2 + 2^2 = 8.$$

Notes on Example 5.4 – Continued

This shows that the two problems have identical optimal function values.

However, the original program is a minimising problem, and the Lagrangian dual is a maximising problem. In particular, this means that for any feasible point x, $f(x) \geq 8$, and for any feasible λ for the Lagrangian dual, $\psi(\lambda) \leq 8$. Therefore, for any feasible pair of points x, λ, we know that

$$\psi(\lambda) \leq f(x).$$

Lagrangian duality

Example 5.4 demonstrates the following duality theorem.

Theorem (Lagrangian duality)

Let (NLP) be a convex program, and let $\phi(x)$ and $\psi(\lambda, \eta)$ be defined as previously.

1. (Weak Lagrangian duality) Let x be feasible for (NLP) and the multipliers λ, η be such that $\lambda \geq 0$. Then $\psi(\lambda, \eta) \leq \phi(x)$.
2. (Strong Lagrangian duality) A triple (x^*, λ^*, η^*) is a KKT point of (NLP) if and only if $\lambda^* \geq 0$ and $\psi(\lambda^*, \eta^*) = \phi(x^*)$.

Summary – Lagrangian duality

The NLP should have the same solution as *the penalty formulation* of the NLP,

$$\min_x \phi(x)$$

where

$$\phi(x) = \max_{\lambda \geq 0, \eta} L(x, \lambda, \eta) = \begin{cases} f(x) & \text{if } x \text{ is feasible for (NLP)} \\ \infty & \text{otherwise.} \end{cases}$$

and the *Lagrangian dual* problem,

$$\max_{\lambda \geq 0, \eta} \psi(\lambda, \eta).$$

where $\psi(\lambda, \eta) = \min_x L(x, \lambda, \eta)$.

Summary – Lagrangian duality

With the above definitions, we have the following duality theorem.

Theorem 13 (Lagrangian duality)

Let (NLP) be a convex program, and let $\phi(x)$ and $\psi(\lambda, \eta)$ be defined as previously.

1. (Weak Lagrangian duality) Let x be feasible for (NLP) and the multipliers λ, η be such that $\lambda \geq 0$. Then $\psi(\lambda, \eta) \leq \phi(x)$.
2. (Strong Lagrangian duality) A triple (x^*, λ^*, η^*) is a KKT point of (NLP) if and only if $\lambda^* \geq 0$ and $\psi(\lambda^*, \eta^*) = \phi(x^*)$.

Proof of Theorem 13

1. From the definitions,

$$\psi(\lambda, \eta) = \min_{x'} L(x', \lambda, \eta) \leq L(x, \lambda, \eta) \leq \max_{\lambda' \geq 0, \eta'} L(x, \lambda', \eta') = \phi(x).$$

2. ($\Rightarrow$) Suppose (x^*, λ^*, η^*) is a KKT point. Then the Saddle Inequality holds. Therefore

$$\psi(\lambda^*, \eta^*) = \min_{x} L(x, \lambda^*, \eta^*) = L(x^*, \lambda^*, \eta^*)$$

from the Saddle Inequality.

Proof of Theorem 13 – Continued

Furthermore

$$\phi(x^*) = \max_{\lambda \geq 0, \eta} L(x^*, \lambda, \eta) = L(x^*, \lambda^*, \eta^*)$$

again from the Saddle Inequality. This means that $\psi(\lambda^*, \eta^*) = \phi(x^*)$.

($\Leftarrow$) Now suppose that $\psi(\lambda^*, \eta^*) = \phi(x^*)$. From the proof of part 1,

$$\psi(\lambda^*, \eta^*) \leq L(x^*, \lambda^*, \eta^*) \leq \phi(x^*),$$

so all three quantities are equal.

Proof of Theorem 13 – Continued

Therefore

$$\min_{x} L(x, \lambda^*, \eta^*) = L(x^*, \lambda^*, \eta^*)$$

which implies that x^* minimises $L(x, \lambda^*, \eta^*)$. Similarly

$$\max_{\lambda \geq 0, \eta} L(x^*, \lambda, \eta) = L(x^*, \lambda^*, \eta^*)$$

which implies that λ^* and η^* maximise $L(x^*, \lambda, \eta)$ (subject to $\lambda \geq 0$).

However, this is the Saddle Inequality, so (x^*, λ^*, η^*) is a KKT point and the theorem is proved.

Class Exercise 5.5

Consider the non-linear program

$$\begin{array}{ll} \min & x_1^2 + x_2 \\ \text{s.t.} & x_2 \geq 2. \end{array}$$

1. Sketch the feasible region and some level curves for this NLP, and hence graphically determine the global minimum x^*.
2. Write down the penalty function version of this NLP, and the objective function for the Lagrangian dual.
3. Given that the optimal Lagrange multiplier at the point x^* you found in (1) is $\lambda^* = 1$, use the Strong Duality Theorem to verify that (x^*, λ^*) is a KKT point of the NLP.

Solution to Class Exercise 5.5

The Lagrangian is

$$L(x, \lambda) = x_1^2 + x_2 + \lambda(2 - x_2).$$

The penalty function is

$$\phi(x) = \max_{\lambda \geq 0} x_1^2 + x_2 + \lambda(2 - x_2).$$

The objective function for the Lagrangian dual is

$$\psi(\lambda) = \min_{x} x_1^2 + x_2 + \lambda(2 - x_2).$$

Note: for any feasible x and $\lambda \geq 0$, weak duality tells us that

$$\psi(\lambda) \leq \phi(x).$$

Solution to Class Exercise 5.5 – Continued

Now consider the point $(x^*, \lambda^*) = ((0, 2), 1)$. The two functions are

$$\phi(0, 2) = \max_{\lambda \geq 0} 0^2 + 2 + 0 \cdot \lambda = 2,$$

and

$$\psi(1) = \min_{x} x_1^2 + x_2 + 2 - x_2 = \min_{x_1} x_1^2 + 2 = 2.$$

By the Strong Duality Theorem, this verifies that $((0, 2), 1)$ is a KKT point of the program.

Notes on Class Exercise 5.5 – Continued

Note that from weak Lagrangian duality,

$$\max_{\lambda \geq 0, \eta} \psi(\lambda, \eta) \leq \min_{x} \phi(x),$$

so if the conditions for strong duality hold, it is easy to see that (x^*, λ^*, η^*) is not just a KKT point, but also a global minimum. This also follows from the fact that the program is convex.

We can also see from the above inequality that in this case, (λ^*, η^*) is a global maximum for the Lagrangian dual.

Lagrangian duality - Corollaries

Weak and strong duality also lead to some other interesting results.

Corollary 3 (NLP lower bound)

If we set z to be the optimal function value of the non-linear program

$$z = \min_x \{f(x) : g(x) \leq 0, h(x) = 0\},$$

then if the program is a convex program and $\lambda \geq 0$, we have

$$z \geq \psi(\lambda, \eta)$$

for any η.

Lagrangian duality - Corollary 3

This tells us that the optimal function value of the original program is bounded below by the objective of the Lagrangian dual, evaluated at any feasible point.

This is useful, because we can then evaluate the Lagrangian dual at any feasible point to gain a lower bound for the original program, even if we do not know what the solution either program is. By evaluating the dual at different feasible points, we can (hopefully) get closer and closer to the actual optimum value for the primal program.

Lagrangian duality - Corollary 3 – Proof

From weak duality, we know that at any feasible (for the Lagrangian dual) λ and η,

$$\min_{x} \phi(x) \geq \psi(\lambda, \eta).$$

But $\min_x \phi(x)$ is the optimal objective function of the original program (since $\phi(x)$) is the ideal penalty function.
QED

Lagrangian duality - Corollary 3

Example 5.6

Going back to our first example 5.2:

$$\begin{aligned} \min \quad & x_1^2 + x_2^2 \\ \text{s.t.} \quad & -x_1 - x_2 + 4 \leq 0 \\ & x_1, x_2 \geq 0. \end{aligned}$$

The Lagrangian dual objective function is

$$\psi(\lambda) = 4\lambda_1 - \frac{1}{4}(\lambda_1 + \lambda_2)^2 - \frac{1}{4}(\lambda_1 + \lambda_3)^2.$$

Example 5.6 for Corollary 3 - Continued

Evaluating this function at any feasible λ gives a lower bound for the objective function of the primal. So if the optimal value of the primal is z, we can say, for example, that

$$\begin{aligned} z &\geq \psi(0,0,0) = 0 \\ z &\geq \psi(1,1,1) = 2 \\ z &\geq \psi(1,2,3) = -\frac{9}{4} \\ z &\geq \psi(4,0,0) = 8. \end{aligned}$$

We already know that $z = 8$, and so can see that these inequalities are valid.

Lagrangian duality - Corollary 4

Corollary 4 (Conditions for KKT point)

If x is feasible for a convex program, $\lambda \geq 0$, and $f(x) = \psi(\lambda, \eta)$, then (x, λ, η) is a KKT point and x is a global minimum of the program.

This works in much the same way as strong duality, except that we need merely to evaluate the original objective function (and the Lagrangian dual).

Lagrangian duality - Corollary 4 – Proof

If x is feasible, then

$$\begin{aligned}
\phi(x) &= \max_{\lambda \geq 0, \eta} L(x, \lambda, \eta) \\
&= \max_{\lambda \geq 0, \eta} f(x) + g(x)^T \lambda + h(x)^T \eta \\
&= f(x) + \max_{\lambda \geq 0} g(x)^T \lambda \\
&= f(x).
\end{aligned}$$

The last equation follows because $g(x) \leq 0$. Therefore (x, λ, η) satisfies the conditions for strong duality and is a KKT point. Since the program is convex, it is also a global minimum.
QED

Lagrangian duality - Corollary 4

Example 5.7

In the previous example, we know that $\psi(4,0,0) = 8$. Evaluating the objective function at the point $(2,2)$ gives

$$f(2,2) = 2^2 + 2^2 = 8$$

which demonstrates that $(2,2)$ is the global minimum of the original program.

Subsection 4

Wolfe duality

Wolfe duality

The Lagrangian dual has some good properties but it does have a drawback. Recall that the objective function is

$$\psi(\lambda, \eta) = \min_{x} L(x, \lambda, \eta).$$

This involves minimising the Lagrangian just to calculate the objective function. If the Lagrangian is complex, this can be difficult and undesirable.

Wolfe duality

The Wolfe dual gets around this difficulty by observing that since the above problem is an unconstrained problem, at the minimum of the Lagrangian its gradient must be 0 with respect to the decision variables:

$$\nabla_x L(x, \lambda, \eta) = 0.$$

At this point, $\psi(x, \lambda, \eta)$ is equal to the Lagrangian, which allows us to convert the Lagrangian dual into the Wolfe dual.

Wolfe duality

Definition: The Wolfe dual is

$$\begin{aligned}
\max_{x,\lambda,\eta} \quad & L(x, \lambda, \eta) \\
\text{s.t.} \quad & \lambda \geq 0 \\
& \nabla_x L(x, \lambda, \eta) = 0.
\end{aligned}$$

Note that this is a (constrained) optimisation problem in all three variable sets, x, λ, and η. As with the Lagrangian dual, it is a maximisation problem.

Wolfe duality

We make two further observations. Firstly, we cannot assume that $\nabla_x L(x, \lambda, \eta)$ is affine, and therefore the Wolfe dual may not be a convex program.

Also, note that a feasible point in the primal has to satisfy $g(x) \leq 0$, $h(x) = 0$, but these constraints are not applied to x in the Wolfe dual. The equality constraint on x in the dual is not in the primal either, so there is no relationship between primal and dual feasible points.

The Wolfe dual of an LP

Let us look at the Wolfe dual of a linear program:

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & x \geq 0 \\ & Ax = b. \end{aligned}$$

If we take $g(x) = -x$ and $h(x) = b - Ax$, the Lagrangian is

$$\begin{aligned} L(x, \lambda, \eta) &= c^T x - \lambda^T x + \eta^T (b - Ax) \\ &= (c - \lambda - A^T \eta)^T x + \eta^T b. \end{aligned}$$

The Wolfe dual of an LP

This gives us $\nabla_x L(x, \lambda, \eta) = c - \lambda - A^T \eta$, so the Wolfe dual of this program is

$$\begin{aligned} \max_{x,\lambda,\eta} \quad & (c - \lambda - A^T \eta)^T x + \eta^T b \\ \text{s.t.} \quad & \lambda \geq 0, \\ & c - \lambda - A^T \eta = 0. \end{aligned}$$

Now, the equality constraint shows that the coefficient of x in the objective function is 0. This removes all of the x's from the dual. Furthermore, if we write the equality constraint as $\lambda = c - A^T \eta$, we can then substitute this into the inequality constraint to remove λ.

The Wolfe dual of an LP

This results in a simplified Wolfe dual of

$$\begin{aligned} \max_{\eta} \quad & b^T \eta \\ \text{s.t.} \quad & A^T \eta \leq c. \end{aligned}$$

This is the standard dual of the original linear program. (Hence the Wolfe dual generalises the standard dual for an LP).

Wolfe duality - Example

Example 5.8

We return to our example from the previous lectures:

$$\begin{aligned} \min \quad & x_1^2 + x_2^2 \\ \text{s.t.} \quad & -x_1 - x_2 + 4 \leq 0 \\ & x_1, x_2 \geq 0. \end{aligned}$$

Construct and simplify the Wolfe dual for the above NLP.

Solution to Example 5.8

The Lagrangian of this program is

$$L(x, \lambda) = x_1^2 + x_2^2 + \lambda_1(-x_1 - x_2 + 4) - \lambda_2 x_1 - \lambda_3 x_2.$$

Therefore the Wolfe dual is

$$\begin{array}{ll} \max & x_1^2 + (-\lambda_1 - \lambda_2)x_1 + x_2^2 + (-\lambda_1 - \lambda_3)x_2 + 4\lambda_1 \\ \text{s.t.} & \lambda \geq 0 \\ & 2x_1 - \lambda_1 - \lambda_2 = 0 \\ & 2x_2 - \lambda_1 - \lambda_3 = 0. \end{array}$$

Again, the Wolfe dual can be simplified somewhat because of the equality constraints.

Solution to Example 5.8 – Continued

From the equality constraints, we know that $-\lambda_1 - \lambda_2 = -2x_1$ and $-\lambda_1 - \lambda_3 = -2x_2$. This eliminates λ_2 and λ_3 from the dual, which gives:

$$\begin{aligned}
\max \quad & -x_1^2 - x_2^2 + 4\lambda_1 \\
\text{s.t.} \quad & \lambda_1 \geq 0 \\
& 2x_1 - \lambda_1 \geq 0 \\
& 2x_2 - \lambda_1 \geq 0.
\end{aligned}$$

Note: This can be solved using the KKT conditions to generate the optimal solution $x_1 = 2, x_2 = 2, \lambda_1 = 4$. These are the optimal values for the primal as well, and back substitution gives $\lambda_2 = 2x_1 - \lambda_1 = 0$ and $\lambda_3 = 2x_2 - \lambda_1 = 0$.

Class Exercise 5.9

Consider the non-linear program

$$\begin{aligned} \min \quad & x_1 + x_2^2 \\ \text{s.t.} \quad & x \geq 0, \\ & x_1 + x_2 \geq 2. \end{aligned}$$

1. Write down the Wolfe dual for this NLP.
2. Simplify your answer in (1) so that the Wolfe Dual is an NLP in 2 variables.
3. [Homework] Solve the NLP in (2) using KKT conditions. How do you know that the solution you find is globally optimal?

Class exercise 2. s. 565

$$\min \; x_1 + x_2^2$$

s.t. $x_1, \; x_2 \geq 0$

$x_1 + x_2 \geq 2$

$\boxed{1}$ $\mathcal{L} = x_1 + x_2^2 - \lambda_1 x_1 - \lambda_2 x_2 + \lambda_3 (2 - x_1 - x_2)$

Wolfe dual

$$\max_{x,\lambda} \; x_1 + x_2^2 - \lambda_1 x_1 - \lambda_2 x_2 + \lambda_3 (2 - x_1 - x_2)$$

$\lambda^T \geq 0$

$1 - \lambda_1 - \lambda_3 = 0$

$2x_2 - \lambda_2 - \lambda_3 = 0$

$\boxed{2}$

$$\max_{x,\lambda} \; \cancelto{0}{(1 - \lambda_1 - \lambda_3)} x_1 + \cancelto{0}{(2x_2 - \lambda_2 - \lambda_3)} x_2 - x_2^2 + 2\lambda_3$$

s.t.

$2x_2 - \lambda_3 = \lambda_2 \Rightarrow$

$2x_2 - \lambda_3 \geq 0$

$\lambda_3 \geq 0 \qquad \lambda_3 \leq 1$

$\alpha_1 = 0 \qquad \alpha_2 > 0$

$x_2 = 0$ KKT point $x_2^* = 0$

$\alpha_2 = 2$ $\lambda_3^* = 2$

$\lambda_3 = 0$ $\alpha_2^* = 2$

second constraint not at equality :

$\alpha_1 > 0 \qquad \alpha_2 = 0$

$\alpha_1 = -2$ (violation)

KKT point $x_2^* = 0, \ \lambda_3^* = 2, \ \alpha_2^* = 2$

it's globally

Weak and Strong Wolfe Duality Properties

Theorem 14 (Wolfe duality)

Suppose (NLP) is a convex program.

1. (Weak Wolfe duality) For any feasible points x of (NLP) and (x', λ, η) of the Wolfe dual, we have $L(x', \lambda, \eta) \leq f(x)$.
2. (Strong Wolfe duality) A triple (x^*, λ^*, η^*) is a KKT point of the primal program if and only if x^* is primal feasible, (x^*, λ^*, η^*) is feasible for the Wolfe dual and $L(x^*, \lambda^*, \eta^*) = f(x^*)$. In this case, x^* is a global minimiser for the primal and (x^*, λ^*, η^*) is a global maximiser for the Wolfe dual.

Proof of the Wolfe Duality Theorem

1. Because the original non-linear program is a convex program, we know that f and g are convex and h is affine. Since $\lambda \geq 0$ from feasibility of the dual point,

$$L(x, \lambda, \eta) = f(x) + \sum_i \lambda_i g_i(x) + \sum_j \lambda_j h_j(x)$$

is a convex function.

Proof of the Wolfe Duality Theorem – Continued

Because (x', λ, η) is feasible for the Wolfe dual,

$$\nabla_x L(x', \lambda, \eta) = 0.$$

Since L is convex, this implies that x' is a global minimiser for the Lagrangian considered as a function of x only.

Proof of the Wolfe Duality Theorem – Continued

Therefore

$$
\begin{aligned}
L(x', \lambda, \eta) &\leq L(x, \lambda, \eta) && (5.7)\\
&= f(x) + \sum_i \lambda_i g_i(x) + \sum_j \eta_j h_j(x) && (5.8)\\
&\leq f(x). && (5.9)
\end{aligned}
$$

The last inequality comes about because x is feasible for the primal, and so $g(x) \leq 0$ and $h(x) = 0$.

Proof of the Wolfe Duality Theorem – Continued

2. ($\Rightarrow$) Let (x^*, λ^*, η^*) be a KKT point of the original program. KKTa gives us

$$\nabla_x L(x^*, \lambda^*, \eta^*) = 0$$

and KKTb gives us

$$\lambda^* \geq 0,$$

so it is clear that (x^*, λ^*, η^*) is feasible for the Wolfe dual.

Proof of the Wolfe Duality Theorem – Continued

Now from KKTb, we know that $(\lambda^*)^T g(x) = 0$, and from KKTc, $(\eta^*)^T h(x) = 0$. Therefore

$$L(x^*, \lambda^*, \eta^*) = f(x^*) + (\lambda^*)^T g(x) + (\eta^*)^T h(x) = f(x^*).$$

But from weak duality, it is impossible for any feasible point of the Wolfe dual to have a higher objective function value than this. Therefore (x^*, λ^*, η^*) is a global maximum for the Wolfe dual with optimal function value $f(x^*)$. Since the original program is convex, any KKT point must be a global minimiser.

Proof of the Wolfe Duality Theorem – Continued

($\Leftarrow$) Now suppose that (x^*, λ^*, η^*) is a feasible point for the Wolfe dual which satisfies $L(x^*, \lambda^*, \eta^*) = f(x^*)$. From weak duality, no feasible point can have a higher Wolfe dual objective than $f(x^*)$, so (x^*, λ^*, η^*) is a global maximiser of the Wolfe dual.

Again from weak duality, no feasible point of the original program can have a smaller objective function than $L(x^*, \lambda^*, \eta^*)$, and therefore x^* is a global minimiser for the original program (and also a KKT point). This proves the theorem.

Weak and Strong Wolfe Duality Properties

Note that for strong duality it is insufficient simply to have a (primal) feasible point x which satisfies $L(x, \lambda, \eta) = f(x)$ for some λ and η. For example, by setting $\lambda = \eta = 0$ we derive

$$L(x, \lambda, \eta) = f(x) + 0^T g(x) + 0^T h(x) = f(x)$$

for any feasible x. However not all feasible x are optimal points!

We also require that (x, λ, η) is feasible for the Wolfe dual (i.e., that it satisfies the KKTa condition).

Wolfe Duality Properties – Example

Example 5.10

We return once more to the problem we have been using as an example to verify Theorem 14:

$$\begin{aligned} \text{min} \quad & x_1^2 + x_2^2 \\ \text{s.t.} \quad & -x_1 - x_2 + 4 \le 0, \\ & x_1, x_2 \ge 0. \end{aligned}$$

Wolfe Duality Properties – Example 5.10

We have shown previously that the Wolfe dual can be simplified to:

$$\begin{aligned} \max \quad & -x_1^2 - x_2^2 + 4\lambda_1 \\ \text{s.t.} \quad & \lambda_1 \geq 0 \\ & 2x_1 - \lambda_1 \geq 0 \\ & 2x_2 - \lambda_1 \geq 0. \end{aligned}$$

The constraints can be written: $0 \leq \lambda_1 \leq \min\{2x_1, 2x_2\}$. Weak duality gives us lower bounds on the optimal objective function of the primal (which we denoted by z).

Wolfe Weak Duality – Example 5.10

For instance, $(x_1, x_2, \lambda_1) = (0, 0, 0)$, $(1, 1, 1)$, $(1, 1, 2)$ and $(2, 2, 4)$ are all feasible for the (reduced) Wolfe dual. Therefore

$$z \geq -0^2 - 0^2 + 4 \times 0 = 0 \tag{5.10}$$
$$z \geq -1^2 - 1^2 + 4 \times 1 = 2 \tag{5.11}$$
$$z \geq -1^2 - 1^2 + 4 \times 2 = 6 \tag{5.12}$$
$$z \geq -2^2 - 2^2 + 4 \times 2 = 8. \tag{5.13}$$

Again we know beforehand that $z = 8$, so we can see that these inequalities are true.

Wolfe Strong Duality – Example 5.10

We know that $(2,2)$ is feasible for the primal program, $((2,2),(4,0,0))$ is feasible for the full Wolfe dual, and that

$L((2,2),(4,0,0)) = 2^2 + 2^2 + 4(-2-2+4) + 0(-2) + 0(-2) = 8$

and

$$f(2,2) = 2^2 + 2^2 = 8.$$

Strong duality now tells us that the original non-linear program has a global minimiser at $(2,2)$, with optimal KKT multipliers $(4,0,0)$. Furthermore the Wolfe dual has a global maximum at $((2,2),(4,0,0))$.