# Course Notes
# for
# MS4327
# Optimisation

J. Kinsella

March 6, 2013

# Contents

# 1 Organisation of Course

- Lectures will be held
  - Mondays at 13:00.
  - Tuesdays at 10:00.
- From Week 2 on we'll use the Tuesday 12:00 class as a tutorial class based on the work of the previous week.
- All classes will be held in C2-062.
- Office Hours: 15:00-17:00 Mondays.
- These Notes are available at `http://jkcray.maths.ul.ie/ms4327/Slides.pdf`

- **The Appendices contain the solutions to many of the Exercises and also some advanced material included for reference.**
- **Not all of the material in the Notes will be used in class or examined at the end of the Semester**
- Other material for the course available at `http://jkcray.maths.ul.ie/ms4327.html`

- Attendance at lectures and tutorials will be recorded.
- The course will be assessed with an end of semester written exam for 80%.
- A Matlab-based project will be assigned for 20% during the semester.
- You should review the work done in a particular week and attempt the problems assigned **before** the Monday 13:00 tutorial.
- Matlab will be the language used for programming examples.
- See `http://jkcray.maths.ul.ie/ms4327.html` for an introduction to Matlab.
- The main reference text for the course is “Numerical Optimization” by Nocedal & Wright.

# 2 Introduction

The word “optimisation ” (often spelled “optimization”) means “selecting the best option from a range of choices”.

- Airlines organise their crew & aircraft schedules to minimise costs.
- Investors seek to create portfolios that maximise the rate of return while avoiding excessive risk. (See Example 10.1.)
- The Electricity Regulator adjusts prices paid to electricity generators so as to minimise the magnitude of the adjustment subject to the constraint that the generators’ costs are met. (See Example 10.2.)
- Students (try to) maximise their performance in a module subject to the constraint of limited study time.

The same is true in Nature.

- Physical systems tend to a state of minimum energy.
- Rays of light follow paths that minimise their travel time.

Optimisation is an important tool in management science, the mathematics of finance & in physical mathematics. The procedure consists of first identifying some **objective** — a quantitative measure of the performance of the system under study — for example the profit from an investment or the potential energy of a physical system. The objective depends on certain characteristics of the system, the **variables** or unknowns. Our goal is to find values of the variables which optimise (maximise or minimise) the objective. Often the variables are restricted or **constrained** in some way.

Once the physical (or financial) description of the problem has been translated into mathematical notation — a **mathematical model** — an optimisation algorithm can be used to find a solution. For all but the simplest problems, an exact solution cannot be calculated directly. Instead a suitable algorithm must be chosen which will approximate the solution as closely as required. Often a set of **optimality conditions** can be applied to the final values returned by the algorithm to check that they yield a solution to the problem.

## 2.1 Mathematical Formulation

Mathematically speaking, optimisation is the maximisation or minimisation of a function subject to constraints on its variables. We use the following notation:

- $x$ is the vector of **variables**: $x \in \mathbb{R}^n$.
- $f$ is the **objective function** , a real-valued function of $x$ that we want to maximise or minimise: $f : \mathbb{R}^n \to \mathbb{R}$.
- $c$ is the vector of **constraints** that the unknowns must satisfy. This is a vector function of the variables $x$. The number of components in $c$ is the number of individual restrictions that we place on the variables: $c_i : \mathbb{R}^n \to \mathbb{R}$ where $i$ is an index labelling each constraint.

- The general optimisation problem can then be written as:

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{subject to} \begin{cases} c_i(x) = 0, & i \in \mathcal{E}, \\ c_i(x) \geq 0, & i \in \mathcal{I}. \end{cases} \tag{2.1}$$

- Here $f$ and each $c_i$ are real-valued functions of the variables $x$. The sets $\mathcal{E}, \mathcal{I}$ are “index sets” — $\mathcal{E}$ and $\mathcal{I}$ contain the indices of the equality constraints and inequality constraints respectively.

**Example 2.1** *Consider the problem:*

$$\textit{Minimise } (x_1 - 2)^2 + (x_2 - 1)^2 \textit{subject to } \begin{cases} x_1^2 - x_2 \le 0, \\ x_1 + x_2 \le 2. \end{cases}$$

*We can write this problem in the form 2.1 by defining*

$$f(x) = (x_1 - 2)^2 + (x_2 - 1)^2, \quad x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix},$$

$$c(x) = \begin{bmatrix} c_1(x) \\ c_2(x) \end{bmatrix} = \begin{bmatrix} -x_1^2 + x_2 \\ -x_1 - x_2 + 2 \end{bmatrix}, \mathcal{I} = \{1, 2\}, \mathcal{E} = \{\}$$

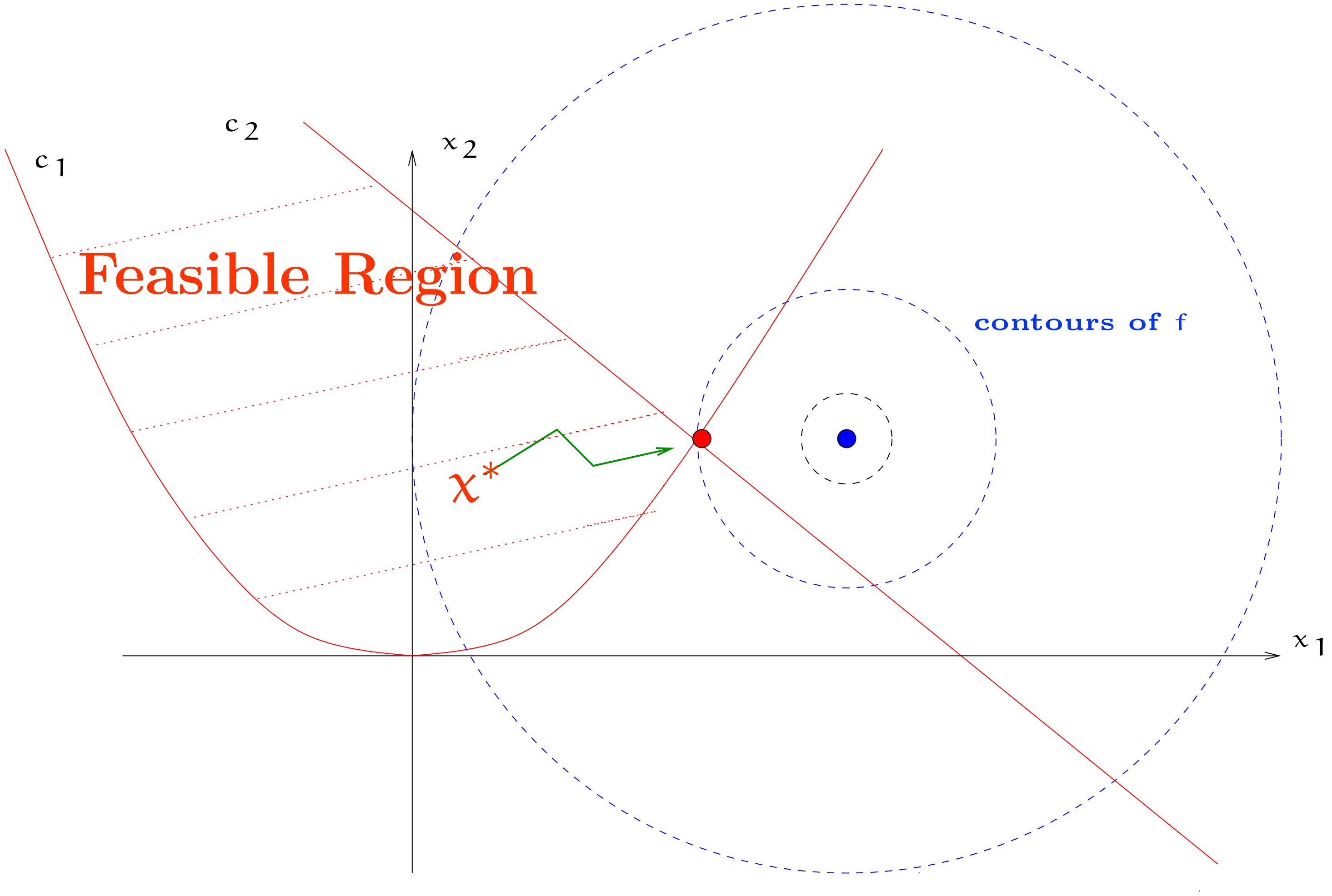


Figure 1: Geometrical representation of Example 2.1

- Figure 1 shows the **contours** of the objective function, the set of points for which $f(x)$ has a constant value.
- In two dimensions ($n = 2$) these are curves in the plane — in general we refer to **level surfaces** in $\mathbb{R}^n$.
- It also shows the **feasible region** — the set of points satisfying all the constraints — and the optimal point $x^*$, the solution of the problem.
- Clearly the problem could be transformed into a **maximisation** problem by simply replacing $f$ by $-f$.

## 2.2 Some Definitions

Most of the following definitions are obvious and/or familiar and are listed here for convenience. I'll state them as either/ors..

### 2.2.1 Linear vs. NonLinear Optimisation

- Most students are familiar with Linear Programming (LP) where the objective function and constraints are all linear functions of $x$; $f(x) = c^T x$ and $c_i(x) = a_i^T x - b_i$, $i \in \mathcal{I} \cup \mathcal{E}$.
- A particular Linear Programming problem is called a Linear Program (also abbreviated to LP!).
- LP's are studied in Ch. 9.

- On the other hand Nonlinear Optimisation (often called Non Linear Programming — NLP) allows for the objective function and the constraints (if any) to be general non-linear functions of $x$ as in Example 2.1 above.
- A particular Non Linear Programming problem is called a Non Linear Program ( abbreviated to NLP).
- If the objective function is quadratic and the constraints are linear then the NLP reduces to a Quadratic Program; studied in Ch. 10 and abbreviated to QP.

### 2.2.2 Continuous vs. Discrete Optimisation

- In some problems the variables make sense only if they take on integer values — e.g. the problem of scheduling U.L.'s timetable could be stated as "Minimise the number of clashes subject to the rule set".
- The rule set would include rules such as "a student may have a maximum of three classes in succession".
- A clash might be defined as a physical impossibility such as a student (or lecturer) having two classes in different places at the same time.
- Discrete Optimisation is often referred to as Integer Programming, usually restricted to the Linear case.
- In this course we focus on **Continuous Optimisation** problems.

### 2.2.3 Constrained vs. Unconstrained Optimisation

- Problems with the general form 2.1 can be classified according to the nature of the objective function and constraints (linear, non-linear, convex), the number of variables (large or small), the smoothness of the functions (differentiable or non-differentiable).
- The most important distinction is between problems with constraints on the variables and those which do not.
- Unconstrained problems are much easier to solve.
- Most of this course will be devoted to developing algorithms for solving unconstrained problems.

- Constrained problems are solved using either
  - using special techniques (discussed in Ch. 10 at the end of this course in the context of QP's)
  - or sometimes by restating them (Def. 8.4) as unconstrained problems with penalty terms in the objective function which discourage constraint violation.

### 2.2.4 Global vs. Local Optimisation

- Most optimisation algorithms find only a local solution, a point where the objective function is smaller than at all other feasible points in its vicinity.
- They will rarely find the best of all such minima, the **global solution**.
- In this course we will concentrate on searching for local solutions.

### 2.2.5 Convex vs. NonConvex Problems

The term convex is defined in App. A.1.

- Informally, convexity is roughly equivalent to the function having a non-negative second derivative or, if multivariate, a positive semi-definite Hessian.
- Again informally, the problem of minimising a convex function is usually easier.
- The problem will always have a unique global minimum.
- Most problems are not convex.

## 2.3 Exercises

1. Which of the following real-valued functions are convex/concave/neither?

   (a) $f(x) = 3x + 4$

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

   (c) $f(x) = -x^{1/2}$ if $x \geq 0$

   (d)
   $$f(x) = \begin{cases} -x - 1, & x \leq -1 \\ 0, & -1 \leq x \leq 1 \\ x - 1, & x \geq 1 \end{cases}$$

   (e) $f(x) = |x + 3|$

2. Which of the following multivariate functions are convex/concave/neither? (You'll need to refer to Defs. 3.5 and 3.6 below.)

   (a) $f(x) = \|x\|$

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

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

   (d) $f(x_1, x_2) = 2x_1^2 + x_2^2 - 2x_1x_2$

   (e) $f(x_1, x_2, x_3) = x_1^4 + 2x_2^2 + 3x_3^2 - 4x_1 - 4x_1x_2$

   (f) $f(x, y) = (x-1)^2 + 4(x+y)^4$

# 3 Fundamentals of Unconstrained Optimisation

- In unconstrained optimisation I minimise an objective function which depends on real variables — with no restriction on the values of these variables.
- The generic problem is simply:

$$\min_{x \in \mathbb{R}^n} f(x) \tag{3.1}$$

- When $n = 1$ I can plot the function and approximately locate the minima before "setting $dy/dx = 0$ and solving for $x$"..
- When $n = 2$ I can (using a program like `matlab` ) draw a contour plot or a 3D surface plot $z = f(x, y)$. Again, examining the plot will often allow us find the approximate location of minima.

- In non-trivial problems I usually won't be able to visualise the problem — for $n > 1$ I do not have an $x$–$y$ graph, for $n > 2$ I do not have a contour plot.
- In general I will have no "road map", just a "compass" — the ability to calculate the values of $f$ and perhaps its derivatives at any points required.
- Whatever algorithm I use should choose these points so as to move rapidly towards the minimum — I'll show you that the derivative (gradient) plays the role of the "compass" that points towards the minimum.

**Example 3.1 (Rosenbrock's Function:)** *This function*

$$f(x) = (x_1 - 1)^2 + (x_2 - x_1^2)^2 \tag{3.2}$$

*is often used in optimisation to test algorithms as while it should be obvious that the (global) minimum is the point* $(1, 1)$, *simplistic optimisation algorithms will often fail on this problem.*

**Exercise 3.1** *Use Matlab to sketch the contours of Rosenbrock's function. See Section 3.5.6.*

**Exercise 3.2** *Can you generalise Rosenbrock's Function to a function on* $\mathbb{R}^n$*?*

## 3.1 Solutions

- In an ideal world, I would like to find a **global minimiser** of $f$, a point where the function attains its least value.

- Formally:

  **Definition 3.1** *A point* $x^*$ *is a* **global minimiser** *if* $f(x^*) \leq f(x)$ *for all* $x \in \mathbb{R}^n$ *— i.e.* $f(x)$ *at* $x^*$ *is lower than at any other point.*

- Sadly, the global minimiser is difficult to find in general — we usually only have information from the points where the function has been evaluated (hopefully a small number) and the function could dip dramatically in a region as yet unsampled.

- If — on the other hand — I know that the objective function is convex, then I may be able to make global statements, see Theorem 3.6 below.

- Most algorithms can only find a **local** minimiser.
- If I define the term “neighbourhood” by:

  **Definition 3.2** *A neighbourhood $\mathcal{N}(x^*)$ is just an open ball radius $\delta$ (say) centred at $x^*$ :*

  $$\mathcal{N}(x^*) = \{x \in \mathbb{R}^n \quad \text{s.t.} \quad \|x - x^*\| < \delta\}.$$

  *so the boundary of the ball is excluded.*

  then:

  **Definition 3.3** *A point $x^*$ is a* **local minimiser** *if there is a neighbourhood $\mathcal{N}(x^*)$ of $x^*$ such that $f(x^*) \leq f(x)$ for all $x \in \mathcal{N}(x^*)$.*
- A sloppy version of the definition is that $x^*$ is a local minimum for a function $f$ if $f$ takes a smaller value at $x^*$ than at any nearby point.

## 3.2 Optimality Conditions

- First some notation:

**Definition 3.4** *The* **gradient** *of $f$ is the vector of partial derivatives — written $\nabla f(x)$ or often $g(x)$ for short;*

$$\nabla f(x) = \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} \end{bmatrix} \tag{3.3}$$

- In `matlab`, a vector is input as a list:

  ```
  >> a = [1 2 3]
  a = 1 2 3
  ```

  a row vector. To input a column vector in `matlab`, use

  ```
  >> a = [1; 2; 3]
  a =
   1
   2
   3
  ```

- So `matlab`'s default vector format is a **row vector**.
- In mathematics and physics, on the other hand, the convention is that vector quantities like the gradient are represented by column vectors, as above. To save space, I will often write column vectors as row vectors transposed — so

$$\nabla f(x) \equiv g(x) = \left[\frac{\partial f(x)}{\partial x_1}, \quad \ldots, \quad \frac{\partial f(x)}{\partial x_n}\right]^T \tag{3.4}$$

**Definition 3.5** *The* **Hessian** *of* $f$ *is the* $n \times n$ *matrix of second partial derivatives;*

$$\nabla^2 f(x) = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f(x)}{\partial x_2 \partial x_1} & \cdots & \frac{\partial^2 f(x)}{\partial x_2 \partial x_n} \\ \vdots & \cdots & \vdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f(x)}{\partial x_n^2} \end{bmatrix} \tag{3.5}$$

**Example 3.2** *For Rosenbrock's function (3.2), the gradient is*

$$\nabla f(x) \equiv g(x) = \left[2(x-1) - 4x(y-x^2), \quad -4x(y-x^2)\right]^T$$

*and the Hessian is (the lines are for readability and are not normally used)*

$$\nabla^2 f(x) = \left[\begin{array}{c|c} 2 + 12x^2 - 4y & -4x \\ \hline -4x & 2 \end{array}\right]$$

- It is worth noting that the Hessian matrix is always symmetric (as the order of the partial derivatives makes no difference when the function is $C^2$).
- It is also (for all problems of interest) a real matrix.
- Real symmetric matrices have perpendicular "orthogonal" eigenvectors and real eigenvalues — more later.
- In principle, to confirm that $x^*$ is a local minimum, I need to check **all** the neighbouring points — obviously an impossible task.
- Fortunately, when a function is smooth, knowing the function and its derivatives at a point allows us to deduce its shape **near** the point.
- Taylor's Theorem (Thm. 3.1) is exactly what I need.

**Theorem 3.1 (Taylor's)** *Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable ($C^1$) and that $p \in \mathbb{R}^n$.*

*Then I have that:*

- $f(x + p)$ *can be approximated by* $f(x)$ *plus an* **error term** *that depends on the gradient:*

  $$f(x+p) = f(x) + \mathbf{p}^{\mathbf{T}}\nabla \mathbf{f}(\mathbf{x} + \xi\mathbf{p}), \tag{3.6}$$

  *for some* **unknown** $\xi \in (0, 1)$.

- *In addition, if* $f$ *is* $C^2$, *then the* **gradient** *at* $x + p$ *can be approximated by the gradient at* $x$ *plus an* **error term** *that depends on the Hessian:*

  $$\nabla f(x+p) = \nabla f(x) + \int_{\mathbf{0}}^{\mathbf{1}} \nabla^{\mathbf{2}}\mathbf{f}(\mathbf{x} + \mathbf{tp})\,\mathbf{p}\,\mathbf{dt}. \tag{3.7}$$

  *(I cannot usually evaluate the integral over* $t$ *in the error term but may be able to find an upper bound for its norm.)*

- *Finally, again if* $f$ *is* $C^2$, $f(x+p)$ *can be approximated by* $f(x)$ *plus a "gradient" or "first-order" term evaluated at* $x$ *plus an* **error term** *that depends on the Hessian:*

$$f(x+p) = f(x) + p^T \nabla f(x) + \mathbf{\frac{1}{2} p^T \nabla^2 f(x + \xi p)\, p}, \qquad (3.8)$$

*for some* **unknown** $\xi \in (0,1)$*.*

**Proof:** See any multivariate calculus text. ■

- I can derive **necessary conditions** for optimality by assuming that $x^*$ is a local minimiser and then proving facts about $\nabla f(x^*)$ and $\nabla^2 f(x^*)$ — in other words what conditions must the derivatives satisfy if $x^*$ is a local minimiser?

- The term **first-order** below refers to the order of the derivative — similarly **second-order** conditions refer to properties of the Hessian matrix.

  **Theorem 3.2 (First-order Necessary Conditions)** *If $x^*$ is a local minimiser and $f$ is $C^1$ in an open neighbourhood of $x^*$ , then $\nabla f(x^*) = 0$.*

- **Note:** A stronger — i.e. not assuming that the gradient is continuous — version of this result is given in Appendix A.6. I make the assumption that $f$ is $C^1$ as it will always be true in this course as all our algorithms require it. In addition the proof below is more "geometrical" than that in the Appendix.
- **Note:** "$A$ is necessary for B" is equivalent to "$B \Rightarrow A$". So the Theorem is stating that "For $x^*$ to be a local minimiser it is necessary that $\nabla f(x^*) = 0$."

**Proof:** Suppose that the Theorem is false. Then $x^*$ is a local minimiser and $\nabla f(x^*) \neq 0$. Let $p = -g(x^*)(= -\nabla f(x^*))$ and note that $p^T g(x^*) = -\|g(x^*)\|^2 < 0$. As $\nabla f$ is continuous near $x^*$, $p^T g(x)$ "stays negative near $x^*$" so for some $T > 0$ (sufficiently small.

$$p^T g(x^* + \alpha p) < 0 \quad \text{for all } \alpha \in [0, T]. \tag{3.9}$$

Now, I have by Taylor's Theorem (3.6) that

$$f(x^* + p) = f(x^*) + \mathbf{p}^{\mathbf{T}}\mathbf{g}(\mathbf{x}^* + \xi\mathbf{p})$$

$$\text{for some } \xi \text{ where } 0 < \xi < 1.$$

Replacing $p$ in the latter equation by $tp$ $(0 < t < T)$ I have

$$f(x^* + tp) = f(x^*) + \mathbf{tp^T g(x^* + t\xi p)}$$

for some $\xi$ where $0 < \xi < 1$ so if $\alpha = t\xi$ then $0 < \alpha < t < T$.

So $\alpha$ satisfies the condition in Eq 3.9 and so the term in **blue** is negative so $f(x^* + tp) < f(x^*)$ for all $t \in (0, T]$.

I have found a direction $p$ leading away from $x^*$ along which $f$ decreases, so $x^*$ cannot be a local minimum. ■

(This proof may seem over-technical but all the steps are needed.)

- I call $x^*$ a **stationary point** if $g(x^*) = 0$.
- So, by Theorem 3.2, any local minimiser is a stationary point.
- Before deriving second-order necessary conditions, I need the following definition:

  **Definition 3.6 (Positive (Semi)Definite)** *An* $n \times n$ *matrix* $B$ *is* **positive definite** *if* $p^T B p > 0$ *for all* $p \in \mathbb{R}^n, p \neq 0$ *and* **positive semidefinite** *if* $p^T B p \geq 0$ *for all* $p \in \mathbb{R}^n$*.*

- The following Theorem relates this important property of a matrix to whether its eigenvalues are positive/zero/negative.

  **Theorem 3.3** *For a real symmetric matrix* $M$ *with eigenvalues* $\lambda_1, \lambda_2, \ldots, \lambda_n$*;*

  – $M$ *is positive definite* $\Leftrightarrow \lambda_i > 0$ *for* $i = 1, \ldots, n$.

  – $M$ *is positive semidefinite* $\Leftrightarrow \lambda_i \geq 0$ *for* $i = 1, \ldots, n$.

**Proof:** First remember the standard results from Linear Algebra 2:

- If a square (real) matrix $M$ is symmetric, then $M = U\Lambda U^T$.
- The columns $u_i$ of the matrix $U$ are the eigenvectors of $M$.
- So the matrix $U$ is orthogonal $U^T U = I$ which is equivalent to the vectors $u_i$ say being orthonormal ($u_i^T u_j = \delta_{ij}$).
- So $U^T u_i = e_i$ where $e_i$ is a vector which is all $0$ except for a $1$ in the $i^{th}$ row.
- The matrix $\Lambda$ is diagonal with real diagonal elements $\lambda_1, \ldots, \lambda_n$, the eigenvalues of $M$.

$\rightarrow$:Assume that $M$ is positive definite so $p^T M p > 0$ for all non-zero $p$. Choose $p = u_i$, then

$$\begin{aligned} p^T M p &= u_i^T U \Lambda U^T u_i \\ &= (U^T u_i)^T \Lambda U^T u_i = e_i^T \Lambda e_i = \lambda_i, \end{aligned}$$

So $\lambda_i > 0$ for each $i$.

$\leftarrow$: Now suppose that all the eigenvalues $\lambda_i > 0$. Any vector $p$ can be written $p = \sum_{i=1}^{n} p_i u_i$ ( as the eigenvectors $u_i$ form an orthonormal basis for $\mathbb{R}^n$). So $U^T p = \sum p_i e_i$. Then

$$\begin{aligned} p^T M p &= p^T U \Lambda U^T p \\ &= (U^T p)^T \Lambda (U^T p) \\ &= \sum_{i,j} p_i p_j e_i^T \Lambda e_j \\ &= \sum_i \lambda_i p_i^2 \end{aligned}$$

which is certainly positive provided that the vector $p$ is non-zero. ■

**Theorem 3.4 (Second-Order Necessary Conditions)** *If* $x^*$ *is a local minimiser of* $f$ *and* $\nabla^2 f$ *is continuous in an open neighbourhood of* $x^*$ *, then* $g(x^*) = 0$ *and* $\nabla^2 f(x^*)$ *is positive semidefinite .*

**Proof:** Let $x^*$ be a local minimiser of $f$. I know from Theorem 3.2 that $g(x^*) = 0$. Assume the Theorem is false. So there must be a vector $p$ such that $p^T \nabla^2 f(x^*) p < 0$ and as $f$ is $C^2$ at $x^*$ , there must be a (sufficiently small) scalar $T > 0$ such that $p^T \nabla^2 f(x^* + tp) p < 0$ for all $t \in [0, T]$.

Pick any $t$ such that $0 < t \leq T$. By expanding $f$ in a Taylor series around $x^*$ , (see 3.8) I have

$$f(x^* + tp) = f(x^*) + \cancel{tp^T g(x^*)} + \frac{1}{2} t^2 p^T \nabla^2 f(x^* + \xi p)\, p < f(x^*).$$

for some unknown $\xi \in (0, t]$. Again I have found a "descent direction" from $x^*$ so $x^*$ cannot be a local minimiser. ∎

- I now give **sufficient conditions** — guaranteeing that $x^*$ is a local minimum.

**Theorem 3.5 (Second-Order Sufficient Conditions)** *Provided that* $\nabla^2 f$ *is continuous in an open neighbourhood of* $x^*$ *;*

*if*

- $g(x^*) = 0$ *and*
- $\nabla^2 f(x^*)$ *is positive definite*

*then* $x^*$ *is a strict local minimiser of* $f$*.*

*(A* **strict** *minimiser is a point* $x^*$ *where the function takes a value* **strictly** *less than at neighbouring points.)*

**Proof:** As the **Hessian** $\nabla^2 f$ is continuous and positive definite at $x^*$, I can choose a radius $R > 0$ such that $\nabla^2 f(x)$ remains positive definite for all $x$ in the neighbourhood $\mathcal{N} = \{x \mid \|x - x^*\| < R\}$. Pick any nonzero vector $p$ with $\|p\| < R$, then $x^* + p \in \mathcal{N}$ and so

$$\begin{aligned} f(x^* + p) &= f(x^*) + \sout{p^T g(x^*)} + \frac{1}{2} p^T \nabla^2 f(x) p \\ &= f(x^*) + \frac{1}{2} p^T \nabla^2 f(x) p, \end{aligned}$$

where $x = x^* + \xi p$ for some $\xi \in (0, 1)$. Since $x \in \mathcal{N}$, we have $p^T \nabla^2 f(x) p > 0$ and so $f(x^* + p) > f(x^*)$.

So $f$ increases when I move away from $x^*$. This means that $x^*$ is a strict local minimiser of $f$. ■

- Note that the second-order sufficient conditions of Theorem 3.5 are not **necessary**.
- A point $x^*$ may be a strict local minimiser and yet fail to satisfy the sufficient conditions.

  **Example 3.3** *Consider* $f(x) = x^4$. *The point* $x^* = 0$ *is a strict local minimiser at which the Hessian matrix* $\nabla^2 f(x) = 12x^2$ *vanishes and so is not positive definite.*

- I can easily show that, when the objective function is convex, local minimum points are global minimum points.

**Theorem 3.6** *When $f$ is convex, any local minimiser $x^*$ is a global minimiser of $f$. If in addition $f$ is differentiable, then any stationary point $x^*$ is a global minimiser of $f$.*

**Proof:** Suppose that $x^*$ is a local but not a global minimiser. Then I can find a point $z \in \mathbb{R}^n$ with $f(z) < f(x^*)$. Let $x$ be any point on the line segment that joins $x^*$ to $z$, so $x$ satisfies:

$$x = \alpha z + (1-\alpha)x^*, \quad \text{for some } \alpha \text{ between } 0 \text{ and } 1. \tag{3.10}$$

By the the convexity property of $f$, (Def. A.1) I can write

$$f(x) \leq \alpha f(z) + (1-\alpha)f(x^*) < f(x^*). \tag{3.11}$$

Any neighbourhood $\mathcal{N}$ of $x^*$ contains a portion of the line segment 3.10, so there will always be points $x \in \mathcal{N}$ at which 3.11 is satisfied. So $x^*$ cannot be a local minimiser.

For the second part of the theorem, suppose that $x^*$ is a local but not a global minimiser and choose $z$ as above with $f(z) < f(x^*)$. Then from convexity and using the Chain Rule I have;

$$\begin{aligned}
(z - x^*)^T g(x^*) &= \frac{d}{d\alpha}(f(x^* + \alpha(z - x^*))|_{\alpha=0} \\
&= \lim_{\alpha \to 0} \frac{f(x^* + \alpha(z - x^*)) - f(x^*)}{\alpha} \\
&= \lim_{\alpha \to 0} \frac{f(\alpha z + (1 - \alpha)x^*) - f(x^*)}{\alpha} \\
&\leq \lim_{\alpha \to 0} \frac{\alpha f(z) + (\cancel{1} - \alpha)f(x^*) \cancel{- f(x^*)}}{\alpha} \\
&= f(z) - f(x^*) < 0.
\end{aligned}$$

So $g(x^*) \neq 0$ and $x^*$ is not a stationary point . But I assumed that $x^*$ was a local minimum point. **Contradiction**. So a local minimiser must be a global one. ■

- These results provide the foundations for unconstrained optimisation algorithms.
- All algorithms seek a point where $g(x)$ vanishes and some try to ensure that the Hessian is positive semidefinite.

## 3.3 Rates of Convergence

- One of the key measures of performance of an algorithm is its rate of convergence.
- I now define the key terms which I will use in later
- (The prefix Q is often used — as in Q-linear, Q-superlinear, etc.)

**Definition 3.7 (linear)** *Let $\{x_k\}$ be a sequence in $\mathbb{R}^n$ that converges to $x^*$. I say that the convergence is* **linear** *if there is a constant $r$ with $0 < r < 1$ such that*

$$\frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} \leq r, \quad \textit{for all } k \textit{ sufficiently large.} \tag{3.12}$$

This means that the **error** (the distance to the solution $x^*$) decreases at each iteration by at least a constant factor.

**Example 3.4** *The sequence $1 + (0.5)^k$ converges linearly to $1$.*

**Definition 3.8 (superlinear)** *The convergence is* **superlinear** *if*

$$\lim_{k\to\infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} = 0. \tag{3.13}$$

**Exercise 3.3** *Show that the sequence* $1 + k^{-k}$ *converges superlinearly to 1.*

**Definition 3.9 (quadratic)** *The convergence is* **quadratic** *if*

$$\frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|^2} \leq M, \quad \textit{for all } k \textit{ sufficiently large.} \tag{3.14}$$

*(*$M$ *is a positive constant, not necessarily less than* $1$*.)*

**Exercise 3.4** *Can you show that quadratic convergence implies that the error* $\epsilon_k \equiv \|x_k - x^*\|$ *is less than or equal to* $\frac{1}{M}\left(M\epsilon_0\right)^{2^k}$ *where* $\epsilon_0$ *is the starting value of the error?*

*So if* $M < 1/\epsilon_0$ *what can I say?*

- I will show later that
  - Steepest Descent Methods converge linearly
  - Quasi-Newton Methods converge super-linearly
  - Newton's Method converges quadratically.

### 3.3.1 Order Notation

For convenience, I summarise so-called "big-oh" and "little-oh" notation here. Given two non-negative infinite sequences of scalars $\{\eta_k\}$ and $\{\nu_k\}$, I write:

**Definition 3.10 ("Big Oh")**

$$\eta_k = O(\nu_k)$$

*if there is a positive constant* $C$ *such that* $\eta_k \le \nu_k$ *for all* $k$ *suficiently large.*

**Definition 3.11 ("Little Oh")**

$$\eta_k = o(\nu_k)$$

*if the sequence of ratios* $\{\eta_k / \nu_k\}$ *goes to zero, that is, if:*

$$\lim_{k \to \infty} \frac{\eta_k}{\nu_k} = 0.$$

## 3.4 Overview of Algorithms

- In this section I preview the algorithms to be studied in more detail in later Chapters.
- I will not go into detail at this stage.
- There are two fundamental strategies for updating the current point $x_k$ to a new point $x_{k+1}$:

1. Search Direction Methods: Sec. 3.5
   - In the **Search Direction** strategy, the algorithm chooses a **search direction** $p_k$ then performs a **line search**; i.e. searches along this direction from the current point for a new point with a lower function value.
   - The distance to move along $p_k$ can be found by solving the following **one-dimensional** minimisation problem to find a step length $\alpha_k$:

$$\min_{\alpha>0} f(x_k + \alpha p_k) \tag{3.15}$$

   - In practice I try to find an **approximate** solution to 3.15 that is "good enough" — this topic, Line Search, is discussed in Sec. 4.2.
   - Go to Sec. 3.5 to read an introduction to the four most important Search Direction Methods.

2. Trust Region Methods: Sec. 3.6
   - **Trust Region Methods** are very different to Search Direction Methods — I search a region rather than searching along a certain direction.
     - ∗ The strategy uses information available about $f$ to construct a **model function** $m_k$ whose behaviour near the current point $x_k$ is similar to that of the actual objective function $f$.
     - ∗ Because the (often quadratic) model $m_k$ may not be a good approximation to $f$ far from $x_k$, I restrict the search for a minimiser of $m_k$ to some region (the **trust region**).

## 3.5 Introduction to Search Direction Methods

In this section I briefly describe some methods for choosing the **search direction** — these methods are discussed in much greater detail in Chapter 4 and also in Chapters 5 and 6. The following four methods are introduced below;

- The Steepest Descent Method — Sec. 3.5.1
- Newton's Method — Sec. 3.5.3
- Quasi-Newton Methods — Sec. 3.5.4
- The Conjugate Gradient Method — Sec. 3.5.5

### 3.5.1 Steepest Descent Method

- The **steepest-descent direction** $p = -g_k$ is the obvious choice of search direction for a Search Direction method as it is the direction along which f decreases most rapidly — at least for small steps from the start point.

- Surprisingly, it turns out not to be the best choice — as I will see.

- Let's derive the method first.

- I use Taylor's Theorem (Theorem 3.1) to write (see Eq. 3.8):

$$f(x_k + \alpha p) = f(x_k) + \alpha p^T g_k + \frac{1}{2}\alpha^2 p^T \nabla^2 f(x_k + \alpha t\, p), \quad \text{for some } t \in (0, \alpha) \qquad (3.16)$$

- If I ignore the quadratic remainder term, the rate of change in $f$ along the direction $p$ at $x_k$ is simply the coefficient of $\alpha$, namely $p^T g_k$.

- It follows that the unit vector $p$ along which $f$ decreases fastest is the solution to the problem

$$\min_p p^T g_k, \quad \text{subject to } \|p\| = 1. \tag{3.17}$$

- Since $p^T g_k = \|p\| \|g_k\| \cos\theta$, where $\theta$ is the angle between $p$ and $g_k$, I have (as $\|p\| = 1$) that $p^T g_k = \|g_k\| \cos\theta$, so the objective in Eq. 3.17 is minimised when $\cos\theta$ takes its minimum value of $-1$ at $\theta = \pi$ radians.

- So, the solution to Eq. 3.17 is

$$p = -g_k/\|g_k\|.$$

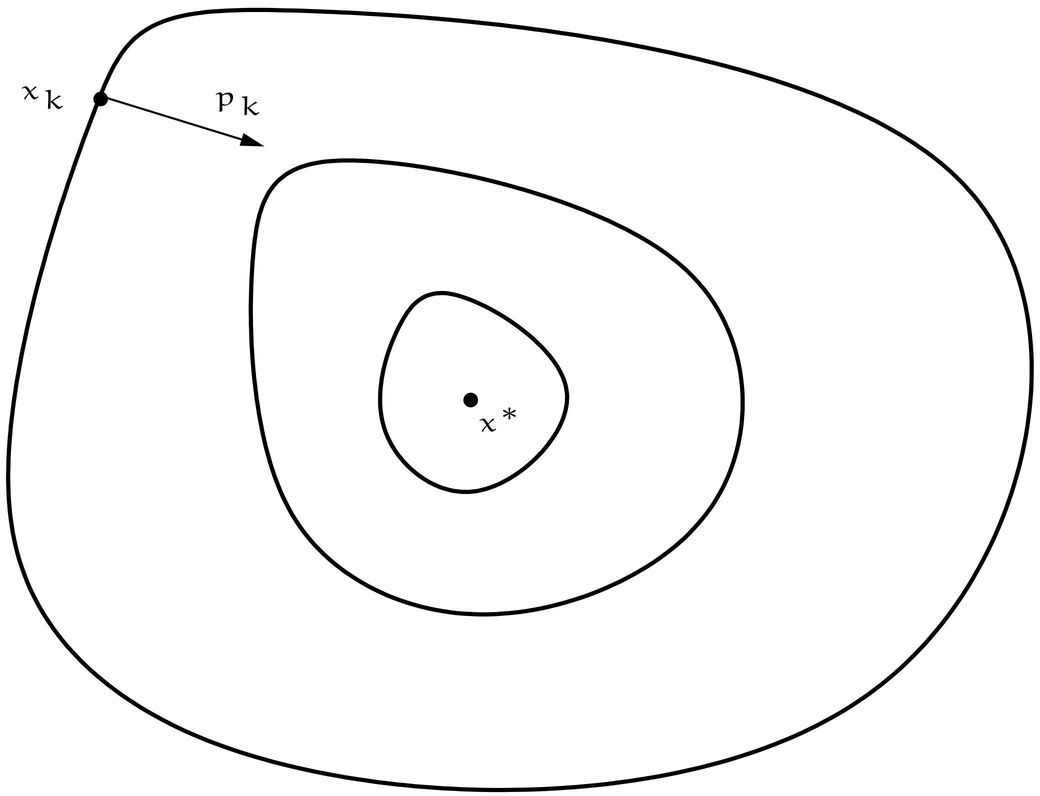


Figure 2: Steepest descent direction in 2D.

- As Figure 2 illustrates, this direction $p$ is perpendicular to the tangent to the contour of the function that passes through the current point — can you explain why?

  Click A.7 ← here for an answer.

- The steepest descent method is a Search Direction method that moves along $p_k = -g_k$ at every step. I will show in Chapter 4 that the steepest descent method is usually so slow at each step that it is useless in practice!

- In Chapter 4 I will show how to choose the step length $\alpha_k$.

### 3.5.2 A Comment on Search Direction Methods in General

- Line search methods may use search directions other than the steepest descent direction.

- In general, any **descent direction** — one which makes an acute angle with $-g_k$ — is guaranteed to produce a decrease in $f$, provided that the step length is sufficiently short.

- See Figure 3 below.

- I can verify this claim by using Taylor's Theorem.

- From Eq. 3.8 I have that

$$f(x_k + \epsilon p_k) = f(x_k) + \epsilon {p_k}^T g_k + O(\epsilon^2).$$

- (See Section 3.3.1 for a summary of "big-oh" notation.)

- When $p_k$ is a descent direction, the angle $\theta_k$ **between $p_k$ and $-g_k$** has $\cos\theta_k > 0$ and so

$$p_k{}^T g_k = (-1)p_k{}^T(-g_k) = (-1)\|p_k\|\,\|g_k\|\cos\theta_k < 0$$

and so $f(x_k + \epsilon p_k) < f(x_k)$ provided $\epsilon$ is sufficiently small & positive — so that the second derivative term cannot affect the sign of the change in $f$.

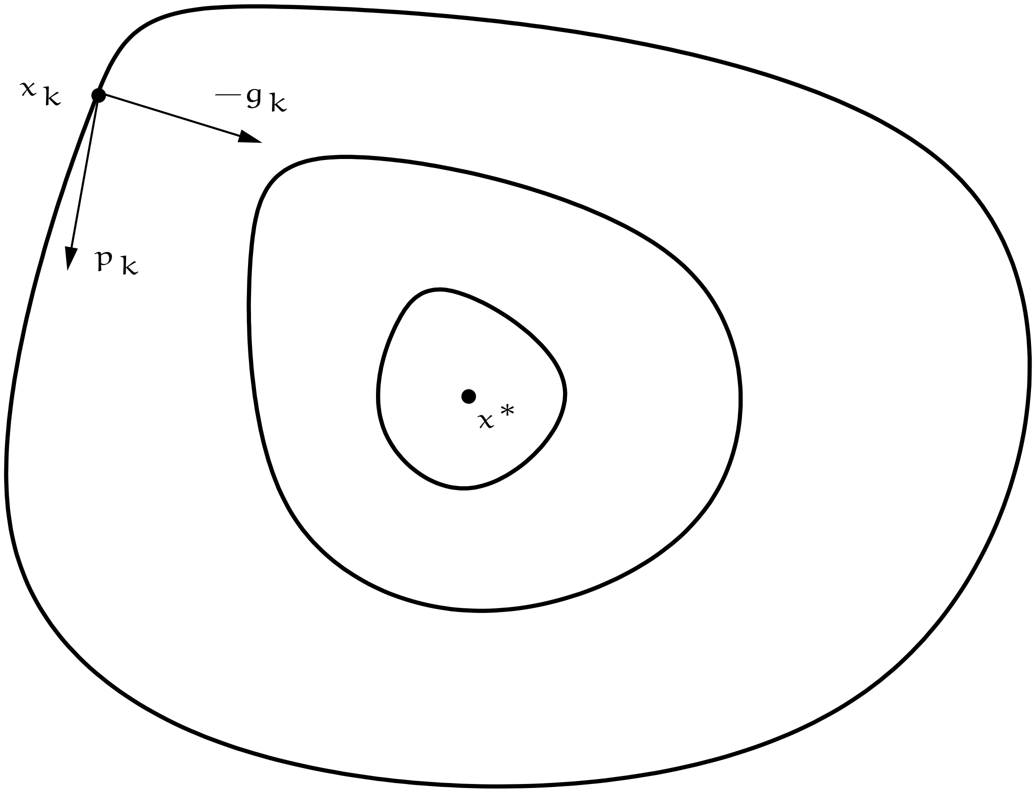


Figure 3: A descent direction $p_k$

### 3.5.3 Newton's Method

- Another important search direction — the one to which others are often compared — is the **Newton direction**.
- This is derived from the second-order Taylor Series approximation to $f(x_k + p)$, which is:

$$f(x_k + p) \approx f_k + p^T g_k + \frac{1}{2} p^T \nabla^2 f_k \ p \equiv m_k(p) \text{ say.} \tag{3.18}$$

- Assuming for the moment that $\nabla^2 f_k$ is positive definite (and therefore invertible), I get the Newton direction by finding the vector $p$ that minimises $m_k(p)$.
- By setting the gradient of $m_k(p)$ with respect to $p$ equal to zero, I find the formula

$$p_k^N = -(\nabla^2 f_k \ )^{-1} g_k \,. \tag{3.19}$$

- The Newton direction $p_k^N$ can be used in a Search Direction method when $\nabla^2 f_k$ is positive definite as in this case I have

$$g_k{}^T p_k^N = -g_k{}^T (\nabla^2 f_k)^{-1} g_k \le -\sigma_k \| g_k \|^2$$

for some positive $\sigma_k$ (can you say what $\sigma_k$ is?).

- Unless the gradient $g_k$ (and therefore the Newton step $p_k^N$) is zero, I have that $p_k^N g_k{}^T < 0$, so the Newton direction is a descent direction.
- It is interesting to note that (unlike the steepest descent method ) there is a natural step length of 1 associated with the Newton direction.
- Implementations of Newton's method use the step $\alpha = 1$ when possible — reducing it only if when necessary.

- When $\nabla^2 f_k$ is not positive definite, the Newton direction may not be defined.
- Even if it is, $p_k^N$ may not be a descent direction.
- One way around this is to change $\nabla^2 f_k$ to make it positive definite.
- For example: at every iteration, replace $\nabla^2 f_k$ by $(\nabla^2 f_k + \alpha I)$, for some $\alpha \in (-\lambda_-, -2\lambda_-)$, where $\lambda_-$ is the most negative eigenvalue of $\nabla^2 f_k$ .
- This choice of $\alpha$ ensures that $\nabla^2 f_k + \alpha I$ is positive definite.
- Methods based on the Newton direction have a fast rate of convergence — typically quadratic (see Def. 3.9).
- The main drawback of such methods is the necessity of computing the Hessian $\nabla^2 f_k$ , which is usually computationally expensive.

### 3.5.4 Quasi-Newton Methods

- **Quasi-Newton** search directions (to be studied in Chapter 6) provide a useful alternative to Newton's method in that they do not require computation of the Hessian and yet attain a superlinear convergence rate.
- In place of the true Hessian, $\nabla^2 f_k$ , they use an approximation $B_k$, which is updated after each step to take account of the additional information gained at each step.
- The updates make use of the fact that changes in the gradient $g = g_k$ provide information about the second derivative of $f$ along the search direction .

- If I define

$$s_k = x_{k+1} - x_k, \quad y_k = g_{k+1} - g_k\,.$$

then I will show in Chapter 6 that successive "approximate Hessians" can be computed using the **BFGS formula** (named after its inventors Broyden, Fletcher, Goldfarb & Shanno), defined by

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}. \tag{3.20}$$

- This update rule preserves the symmetry of the approximation.
- I will show in Chapter 6 that the BFGS rule generates positive definite updates provided the initial approximation $B_0$ is positive definite and $s_k^T y_k > 0$ at each iteration.

- The quasi-Newton search direction  is found by using $B_k$ in place of the exact Hessian in Eq. 3.19., that is;

$$p_k = -B_K^{-1} g_k \,. \tag{3.21}$$

- To avoid having to solve the linear system Eq. 3.21, I will show in Chapter 6 that (using a clever transformation), an update formula can be found for the inverse approximation $H_k \equiv B_k^{-1}$,

$$H_{k+1} = \left(I - \gamma_k s_k y_k^T\right) H_k \left(I - \gamma_k y_k s_k^T\right) + \gamma_k s_k s_k^T, \quad \gamma_k = \frac{1}{y_k^T s_k}. \tag{3.22}$$

- Now the quasi-Newton direction $p_k$ can be calculated using the formula $p_k = -H_k g_k$ .

### 3.5.5 Non-Linear Conjugate Gradient Methods

- The last search direction method in this course (to be studied in Chapter 5) is the class of **non-linear conjugate gradient methods**.
- They take the form

$$p_k = -g_k + \beta_k p_{k-1},$$

  where $\beta_k$ is a scalar that ensures that $p_k$ is a descent direction.
- These methods are much better than steepest descent methods & almost as easy to compute.
- They are not as fast as Newton or quasi-Newton methods, but do not need the exact or approximate Hessian matrix to be stored — a major consideration for large problems.

### 3.5.6 Matlab Tools

Having a look at some/all of the following will give you useful insights into the methods that I will examine later.

- In `http://jkcray.maths.ul.ie/ms4327/m-files/` you will find a simple contour utility and a (two-dim, obviously) version of Rosenbrock's function to try it out on.
- (An alternative way to generate contour plots in the Matlab built-in function `ezcontour`.)
- See the Matlab demo files in `http://jkcray.maths.ul.ie/ms4327/m-files/SimpleLineSearch/`.
- You should read the file `http://jkcray.maths.ul.ie/ms4327/m-files/SimpleLineSearch/Readme.txt` first!

- The file `http://jkcray.maths.ul.ie/ms4327/m-files/SimpleLineSearch/sls.m` is a simple Matlab implementation of the backtracking line search algorithm given on Slide 106 below.

## 3.6 Introduction to Trust Region Methods

- The alternative **trust region** strategy uses information available about $f$ to construct a **model function** $m_k$ whose behaviour near the current point $x_k$ is similar to that of the actual objective function $f$.
- Because the (often quadratic) model $m_k$ may not be a good approximation to $f$ far from $x_k$, I restrict the search for a minimiser of $m_k$ to some region (the **trust region**).
- In other words, I find the candidate step $p$ by finding an approximate solution to the sub-problem:

$$\min_p m_k(x_k + p), \quad \text{where } x_k + p \text{ lies inside the trust region.} \tag{3.23}$$

- If the candidate solution does not produce a sufficient decrease in $f$, I conclude that the trust region is too large and shrink it & re-solve 3.23.
- Usually, the trust region is a ball defined by $\|p\|_2 \leq \Delta$, where the scalar $\Delta > 0$ is called the trust-region radius.
- The model $m_k$ is usually defined to be a quadratic function of the form:

$$m_k(x_k + p) = f_k + p^T \nabla f_k + \frac{1}{2} p^T B_k p, \qquad (3.24)$$

  where $f_k$, $\nabla f_k$ and $B_k$ are a scalar, vector & matrix, respectively.
- The matrix $B_k$ is either the Hessian $\nabla^2 f_k$ or some approximation to it.

**Example 3.5** *Suppose that the objective function is given by Rosenbrock's function:* $f(x) = (x_1 - 1)^2 + 10(x_2 - x_1^2)^2$. *At the point* $x_k = (0, 1)$, *its gradient & Hessian are:*

$$\nabla f_k = \begin{bmatrix} -2 \\ 20 \end{bmatrix} \quad \text{and} \quad \nabla^2 f_k = \begin{bmatrix} -38 & 0 \\ 0 & 20 \end{bmatrix}$$

*Figure 4 shows two possible trust regions (blue circles) and their corresponding steps (the red arrows* $p_k^{(1)}$ *and* $p_k^{(2)}$*). Note that changing the trust region radius will usually change the direction as well as the magnitude of the step* $p_k$. *The step* $p_k^{(2)}$ *is "better" as the Figure shows. The actual contours of* $f$ *are drawn in red and those of the quadratic model are drawn in purple.*

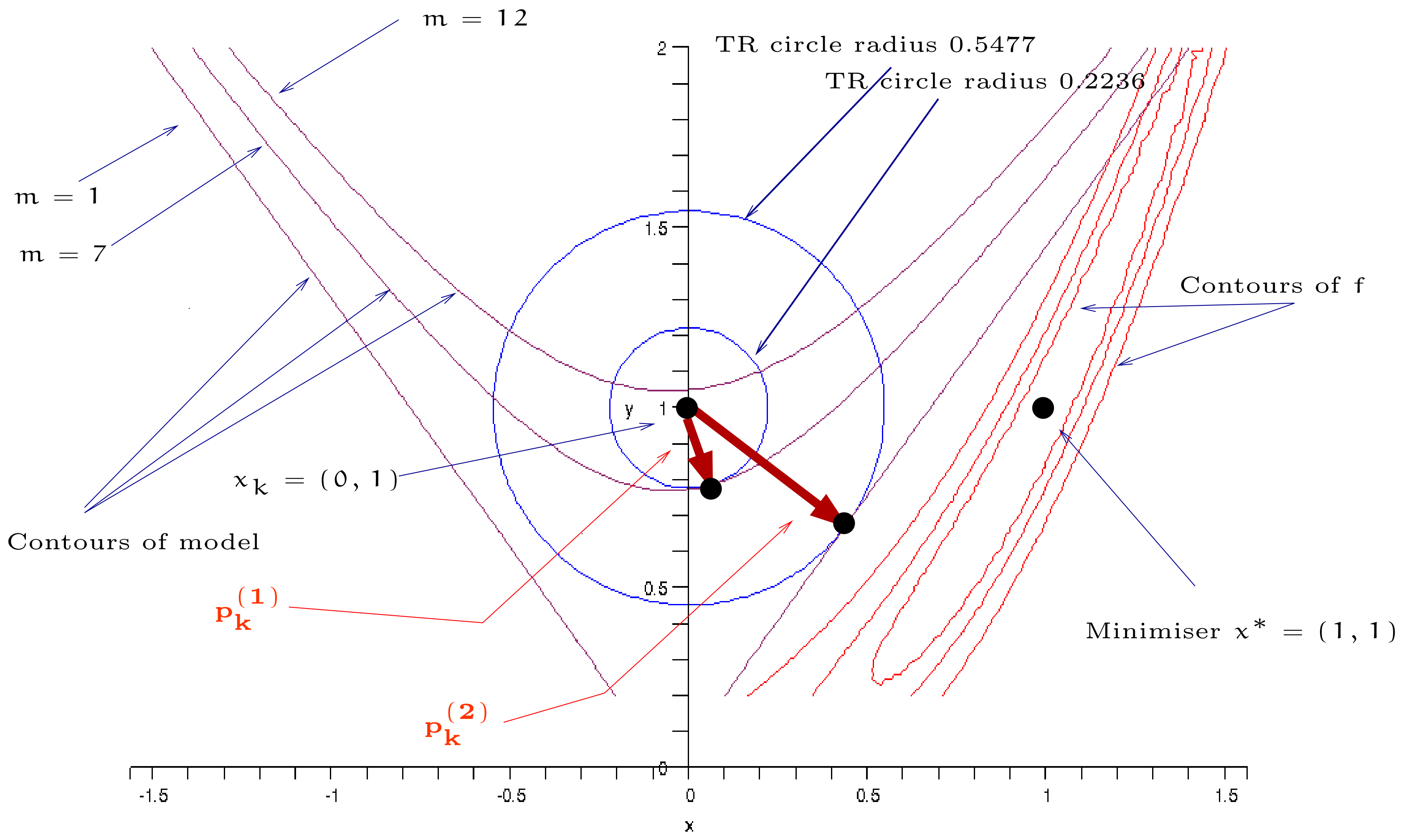


*Figure 4: Trust Region Example*

## 3.7 Contrasting Search Direction and Trust Region Methods

- One way of distinguishing between Search Direction and trust region methods is the order in which they choose the **direction** and **distance** to the next iterate.
- A line search starts by fixing the direction $p_k$ and then finding a suitable distance — the step length $\alpha_k$.
- On the other hand, a trust region method first chooses a maximum distance — the trust region radius $\Delta_k$ — and then seeks a direction and step that yield the best improvement possible subject to this distance constraint.
- If this step proves unsatisfactory, I reduce the radius $\Delta_k$ and try again.
- Chapter 7 discusses the trust region approach in detail.

## 3.8 Exercises

1. Show that the sequence $x_k = 1/k$ is not linearly convergent, though it does convege to zero.
2. Show that the sequence $x_k = 1 + (0.5)^{2^k}$ is quadratically convergent to $1$.
3. Show that the sequence $x_0 + A^{b^k}$ converges superlinearly to $x_0$ if $0 < A < 1$ and $1 < b < 2$.
4. Show that the sequence $x_0 + A^{b^k}$ converges quadratically to $x_0$ if $0 < A < 1$ and $b = 2$.
5. What happens if $b > 2$?
6. Does the sequence $1/(k!)$ converge superlinearly or quadratically?

7. Compute the gradient & Hessian of Rosenbrock's function

$$f(x) = (x_1 - 1)^2 + 10(x_2 - x_1^2)^2. \tag{3.25}$$

   Show that $x^* = (1,1)^T$ is the only minimiser and that the Hessian $H(x,y)$ is positive definite there.

8. Can you say for which points $H(x,y)$ is positive definite & for which it is not?

9. Figure 4 shows the effect on the update direction of changing the trust region radius when using a trust region method . Can you explain all the features of the plot?

10. As Figure 2 illustrates, the steepest descent direction $p = -\nabla f$ is perpendicular to the contours of the function — can you explain why? (Click A.7 ← here for an answer.)

11. Show that $f(x) = 8x_1 + 12x_2 + x_1^2 - 2x_2^2$ has only one stationary point and that it is neither a maximum nor a minimum but is a saddle point. Sketch (or use Maple/Matlab to sketch) the contours of $f$.

12. Let $a$ be a fixed vector in $\mathbb{R}^n$and let $A$ be a fixed $n \times n$ symmetric matrix. Compute the gradient & Hessian of $f(x) = a^T x + \frac{1}{2} x^T A x$.

13. Consider the function $f(x_1, x_2) = (x_1 + x_2^2)^2$. At the point $x^T = (1, 0)$ let the search direction be $p^T = (-1, 1)$. Show that $p$ is a descent direction and find all minimisers of 3.15.

14. I'll say that a function $f$ is **poorly scaled** if changes to $x$ in a certain direction produce much larger variations in the value of $f$ than do changes to $x$ in another direction. An example would be $f(x) = 10^9 x_1^2 + x_2^2$. Suppose that a function $f$ of two real variables is poorly scaled at the solution $x^*$ . Write two Taylor expansions of $f$ around $x^*$ — one along each coordinate direction — and use them to show that the Hessian $\nabla^2 f(x^*)$ is ill-conditioned.

# 4 Search Direction Methods

- As I have shown, each step of a search direction method computes a search direction $p_k$ and then decides how far to move along that direction.
- The iteration is given by

$$x_{k+1} = x_k + \alpha_k p_k, \tag{4.1}$$

  where the positive scalar $\alpha_k$ is called the **step length**.
- Most line search algorithms require that $p_k$ be a descent direction (${p_k}^T g_k < 0$).

- The search direction often takes the form

$$p_k = -B_k^{-1} g_k \,, \tag{4.2}$$

where $B_k$ is symmetric and non-singular — usually an approximation to the Hessian.

- When $p_k$ is defined by 4.2 and $B_k$ is positive definite, then it is easy to check that $p_k$ is a descent direction.
- So the steepest descent method is equivalent to taking $B_k = I$.

- In this Chapter I first discuss in Section 4.2 a set of conditions, the Wolfe conditions, that allow us to decide how far to step along a search direction.
- Next I show in Section 4.3 that any method satisfying a set of conditions (based on the Wolfe conditions) will converge to a stationary point (zero gradient).
- Then in Section 4.4 I try to measure the **rate** of convergence of some commonly used line search methods.
- In Section 4.5 I look at the tricky question of when to stop generating new search directions.

## 4.1 Do Not Read This!

- For the interested student, detailed material on practical line search algorithms may be found in Appendices A.13 and A.14.
- In Appendix A.13 I first discuss some preliminary material then present a step length selection method which will converge under very general conditions.
- Appendix A.14 also includes proofs that the algorithm converges to step lengths that satisfies the Wolfe conditions.

## 4.2 Line Search — How Far Should I Go Along The Search Direction?

- Suppose that I have selected a search direction $p$ (say using Newton's Mathod or a Quasi-Newton Method).
- So I will "move"(i.e. update) $x$ to $x + p$?
- Or could I do better to update $x$ to $x + \alpha p$ for a "good" choice of the **step-size** $\alpha$?
- The "obvious" way to choose $\alpha$ is to seek the global minimiser of the **one-dimensional** function $\phi(\cdot)$ defined by

$$\phi(\alpha) = f(x + \alpha p), \quad \alpha > 0, \tag{4.3}$$

but this is usually too computationally expensive.

- Instead I use an **inexact line search** which systematically generates a sequence of candidate values for $\alpha$, stopping to accept one of them when a set of conditions is satisfied.
- In the following I will work with $\phi$ and its derivative $\phi'$.
- Note that $\phi'(\alpha)$ is just the directional derivative at $x_k + \alpha p_k$, i.e. (using the Chain Rule)

  $$\phi'(\alpha) = {p_k}^T \nabla f(x_k + \alpha p_k) \tag{4.4}$$

  and so

  $$\phi'(0) = {p_k}^T \nabla f(x_k). \tag{4.5}$$
- The line search is often done in two stages; first a **bracketing phase** finds an interval containing suitable step lengths, then a **refinement phase** which computes a good step length within this interval.

- First, I consider appropriate stopping conditions for the line search algorithm.
- A simple (and reasonable) condition on $\alpha$ is that $\phi(\alpha) < \phi(0)$.
- This is certainly necessary but does not ensure satisfactory progress.
- I need to ensure **sufficient reduction** in $\phi$ — or equivalently in $f$ along $p_k$.

### 4.2.1 The Wolfe Conditions

- One widely used inexact line search condition requires that $\alpha_k$ should give **sufficient reduction** in $f$, as measured by:

  $$\phi(\alpha) \leq \phi(0) + c_1 \alpha \phi'(0), \tag{4.6}$$

  for some constant $c_1 \in (0, 1)$.
- This "sufficient decrease condition" is illustrated in Fig. 5.

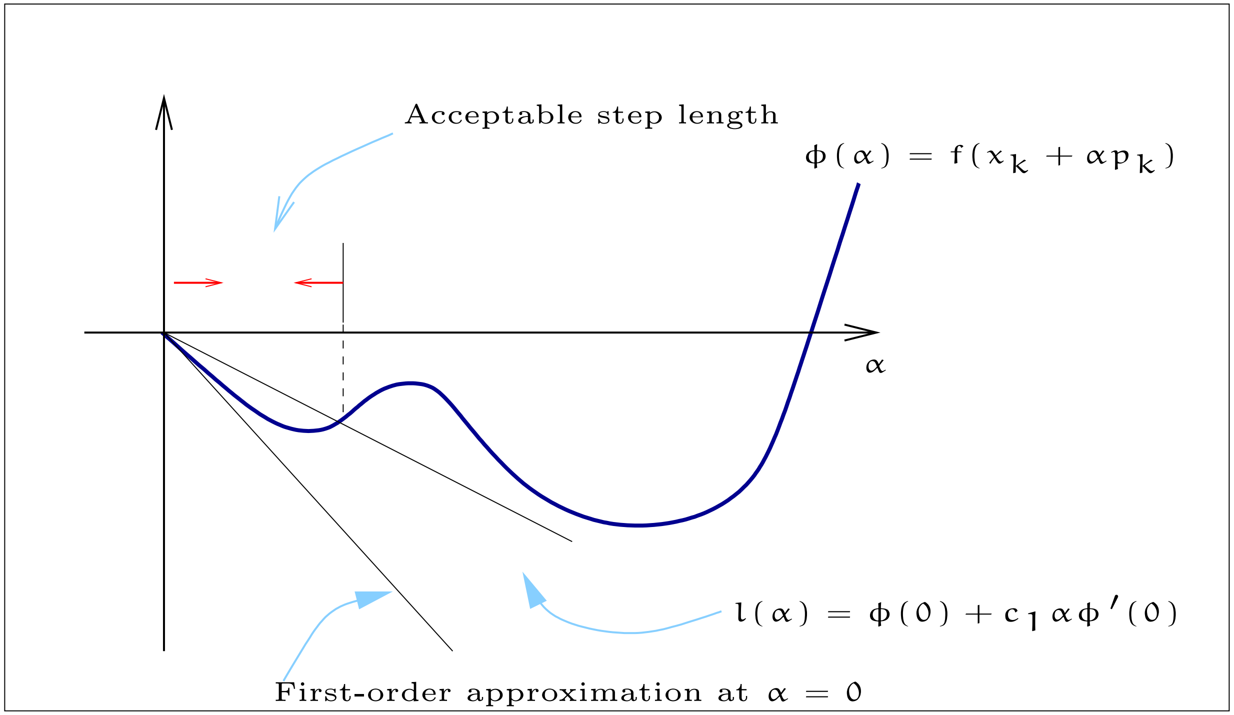

Acceptable step length
$\phi(\alpha) = f(x_k + \alpha p_k)$
$\alpha$
$l(\alpha) = \phi(0) + c_1 \alpha \phi'(0)$
First-order approximation at $\alpha = 0$

Figure 5: The sufficient decrease condition

- The right-hand-side of Eq. 4.6 which is linear in $\alpha$, is denoted $l(\alpha)$.
- The function $l$ has negative slope as $p_k$ is a descent direction , but because $c_1 \in (0, 1)$, it lies above the graph of $\phi$ for small positive $\alpha$.
- The condition simply requires that $\phi(\alpha) \leq l(\alpha)$.
- In practice, $c_1$ is often chosen quite small, say $c_1 = 10^{-4}$, so even a small reduction in $\phi$ will do.

- To rule out unacceptably short steps, I introduce a second requirement — the **curvature condition** — which requires that $\alpha_k$ satisfies:

$$\phi'(\alpha) \geq c_2\phi'(0) \tag{4.7}$$

  for some constant $c_2 \in (c_1, 1)$

  or equivalently

$$p_k{}^T\nabla f(x_k + \alpha p_k) \geq c_2 p_k{}^T g_k\,. \tag{4.8}$$

- The curvature condition simply ensures that the slope of $\phi$ at $\alpha_k$ is greater than $c_2$ times the slope of $\phi$ at $\alpha = 0$.
- As the slope of $\phi$ is initially negative, this simply means that the graph has “flattened out sufficiently” (or has started to increase).
- This condition is illustrated in Fig. 6.

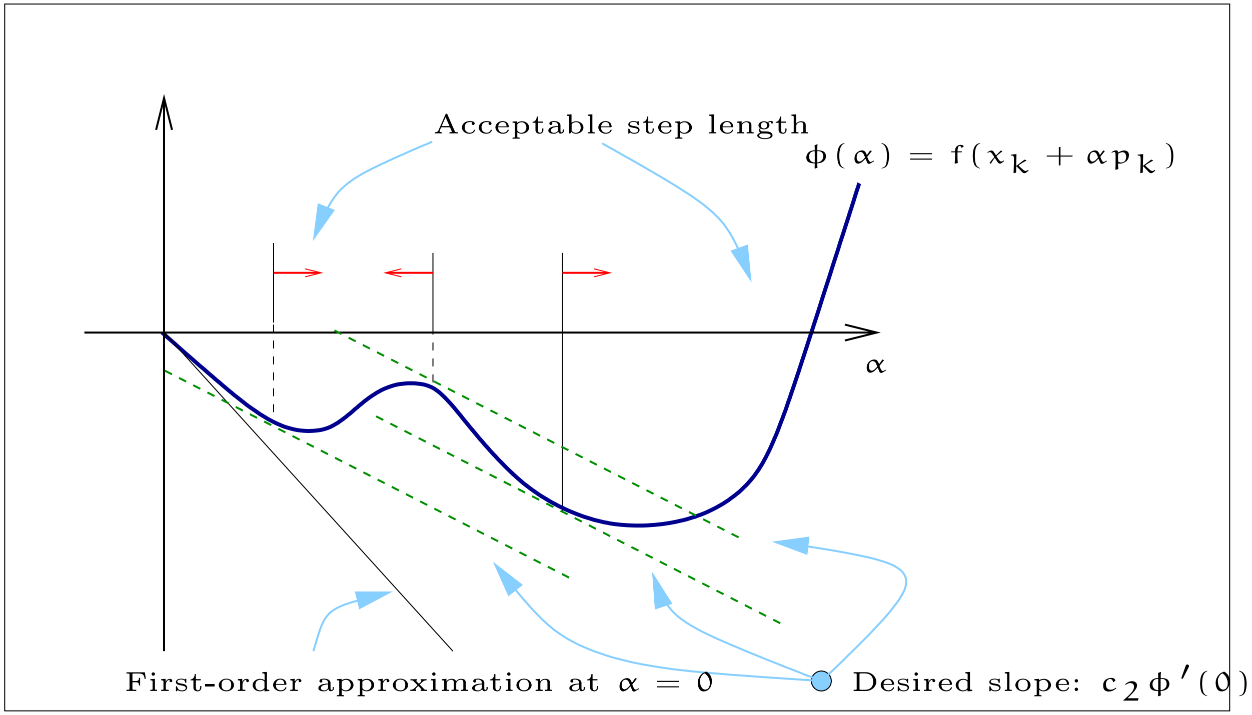

Acceptable step length
$\phi(\alpha) = f(x_k + \alpha p_k)$
$\alpha$
First-order approximation at $\alpha = 0$
Desired slope: $c_2\phi'(0)$

Figure 6: The curvature condition

- Typical values of $c_2$ are $0.9$ when a Newton-type method is used and $0.1$ when a non-linear conjugate gradient method is used to calculate $p_k$ .

  **Exercise 4.1** *Which choice of* $c_2$ *is more likely to produce an* $\alpha$ *value close to a minimum?*

- These two conditions are known collectively as the **Wolfe conditions**.

  For reference:

$$\phi(\alpha) \leq \phi(0) + c_1\alpha\phi'(0), \tag{4.9a}$$

$$\phi'(\alpha) \geq c_2\phi'(0). \tag{4.9b}$$

  with $0 < c_1 < c_2 < 1$.

- They are illustrated in combination in Fig. 7.

- Note that the combined conditions (for this example at least) produce an interval which brackets a minimum point of $\phi(\alpha)$ — i.e. a minimum of $f(x)$ along the search direction $p_k$ .

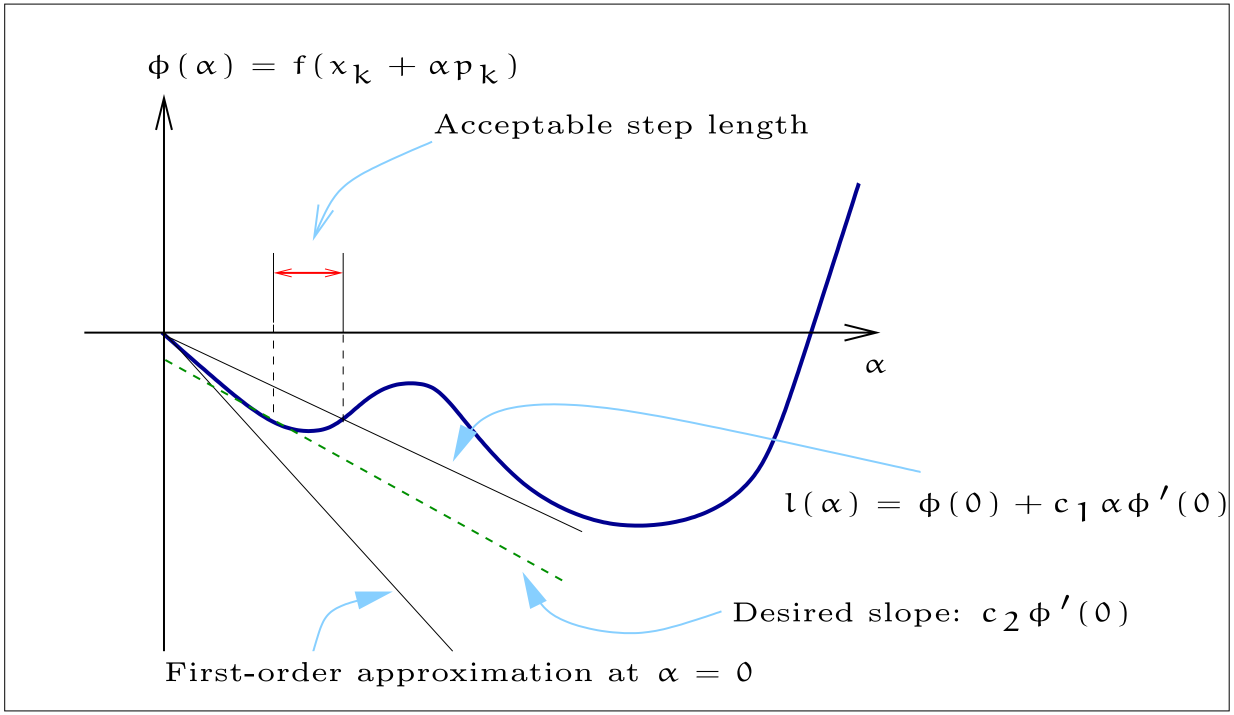

$\phi(\alpha) = f(x_k + \alpha p_k)$
Acceptable step length
$\alpha$
$l(\alpha) = \phi(0) + c_1 \alpha \phi'(0)$
Desired slope: $c_2 \phi'(0)$
First-order approximation at $\alpha = 0$

Figure 7: The **combined** Wolfe conditions

- A more stringent version of the Wolfe conditions, the **strong Wolfe conditions** require $\alpha_k$ to satisfy:

$$\begin{aligned} \phi(\alpha) &\le \phi(0) + c_1 \alpha \phi'(0), && (4.10a) \\ |\phi'(\alpha)| &\le c_2 |\phi'(0)| \equiv -c_2 \phi'(0) && (4.10b) \end{aligned}$$

  with $0 < c_1 < c_2 < 1$.

- The second condition (4.10b) can be written

$$c_2\phi'(0) \le \phi'(\alpha) \le -c_2\phi'(0) \tag{4.11}$$

  — the original (4.9b) with the addition of

$$\phi'(\alpha) \le -c_2\phi'(0) \tag{4.12}$$

  which ensures that $\phi'(\alpha)$ cannot become too positive, so we exclude points that are far from stationary points of $f$.

- I will prove in Lemma 4.1 that there always exist step lengths satisfying the (strong) Wolfe conditions provided f is smooth & bounded below.

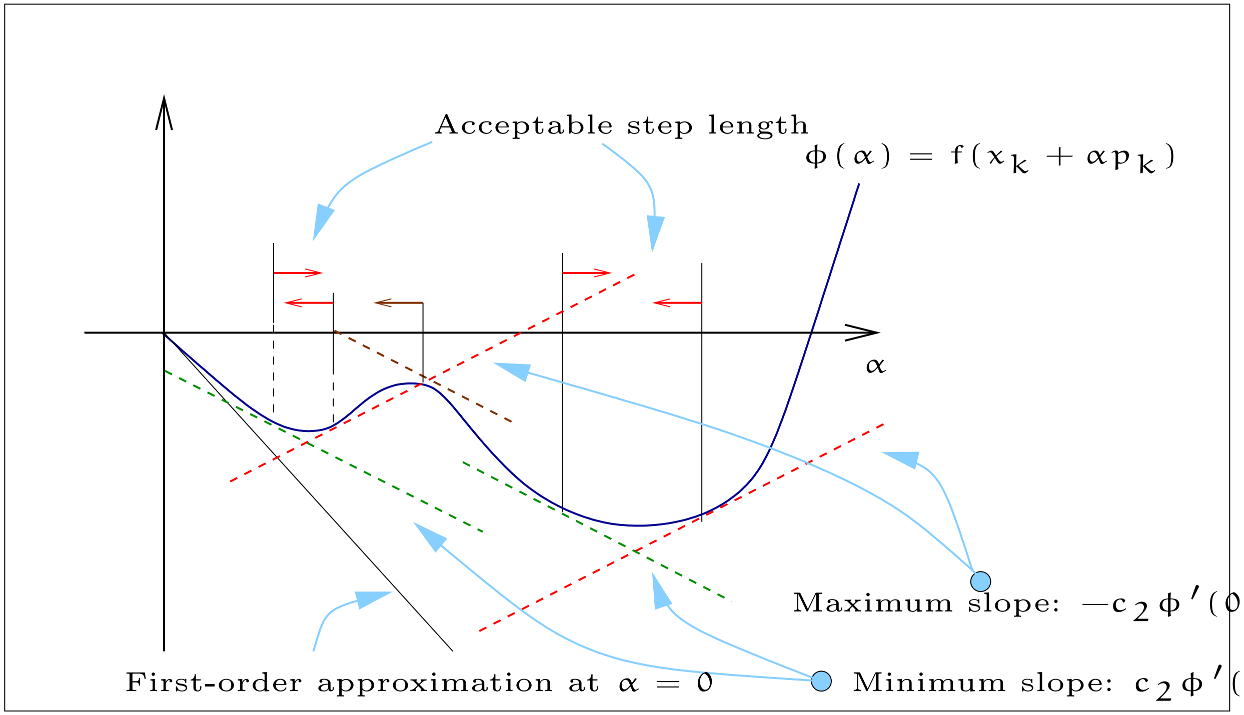

Acceptable step length
$\phi(\alpha) = f(x_k + \alpha p_k)$
$\alpha$
Maximum slope: $-c_2\phi'(0)$
First-order approximation at $\alpha = 0$
Minimum slope: $c_2\phi'(0)$

Figure 8: The **strong** Second Wolfe condition

**Example 4.1** *Suppose that $\phi(\alpha) = e^{-\alpha}$. It is easy to calculate the range of $\alpha$–values that satisfy the various Wolfe conditions.*

- *First Wolfe Condition:*

$$\phi(\alpha) \leq \phi(0) + c_1 \alpha \phi'(0)$$

$$e^{-\alpha} \leq 1 - c_1 \alpha \quad \textit{holds for } 0 < \alpha \leq \bar{\alpha} \approx \frac{1}{c_1} \textit{ provided } c_1 \ll 1.$$

*( Can you justify this result? What if $c_1$ is close to $1$?)*

- *Second Wolfe Condition (weak):*

$$\begin{aligned} \phi'(\alpha) &\geq c_2 \phi'(0) \\ -e^{-\alpha} &\geq -c_2 \\ e^{-\alpha} &\leq c_2 \\ \alpha &\geq \ln \frac{1}{c_2} > 0. \end{aligned}$$

– *Second Wolfe Condition (strong):*

$$|\phi'(\alpha)| \le -c_2\phi'(0)$$

$$c_2\phi'(0) \le \phi'(\alpha) \le -c_2\phi'(0)$$

$$\mathbf{-c_2 \le e^{-\alpha}} \le c_2$$ *The* **blue** *inequality holds as* $c_2 > 0$.

$$\alpha \ge \ln\frac{1}{c_2}$$ *The same as the "weak" WC2.*

– *So Weak/Strong Wolfe Conditions satisfied if* $\ln\frac{1}{c_2} \le \alpha \le \bar{\alpha} \approx \frac{1}{c_1}$ *when* $c_1 \ll 1$. *Check that the result is (approximately)* $\ln\frac{1}{c_2} \le \alpha \le \bar{\alpha} \approx 2(1-c_1)$ *when* $c_1 \approx 1$. *(You can get a good approximation for intermediate values of* $c_1$ *by interpolating:* $\bar{\alpha} \approx (1-c_1)/c_1 + 2c_1(1-c_1)$.*)*

***Exercise*** *4.2 Check this all makes sense by plotting for (say)* $c_1 = 0.1$, $c_1 = 0.5$ *and* $c_1 = 0.9$.

- That was a pretty simple example — $e^{-\alpha}$ is monotone decreasing.
- The next example is still easy to sketch but it is not as easy to calculate the range of $\alpha$–values.
- Of course in practice I don't try to calculate the range of $\alpha$–values that satisfy the Wolfe conditions — I use a search method to try to find **some** $\alpha$–value that satisfies them.

**Example 4.2** *Let $\phi(\alpha) = 1 - \sin\alpha$. Then (just looking for the* **first** *interval where the conditions hold)*

- *WC1:* $1 - \sin\alpha \le 1 - c_1\alpha$ *or* $\sin\alpha \ge c_1\alpha$. *There is no "formula" for the* $\alpha$*–range but when* $c_1 < 2/\pi$, *this holds at least in the interval* $[0, \pi/2]$. *As* $c_1$ *decreases, the interval widens approaching* $[0, \pi]$ *as* $c_1 \to 0$ *— draw a sketch.*
- *WC2 (weak):* $-\cos\alpha \ge -c_2$ *or* $\cos\alpha \le c_2$ *which holds when* $\alpha \ge \arccos c_2$.
- *WC2 (strong):* $-c_2 \le -\cos\alpha \le c_2$ *or* $-c_2 \le \cos\alpha \le c_2$ *which holds when* $\arccos c_2 \le \alpha \le \pi - \arccos c_2$.

*So if* $\alpha \in [\arccos c_2, \min(\pi/2, \pi - \arccos c_2)]$ *then the strong Wolfe conditions hold.*

***Exercise** 4.3* *Download* *http://jkcray.maths.ul.ie/ms4327/m-files/wolfe.m* *and experiment with the effects of changing* $c_1$ *and* $c_2$.

***Exercise** 4.4* *Download* *http://jkcray.maths.ul.ie/ms4327/m-files/WolfeGen.m* *and* *http://jkcray.maths.ul.ie/ms4327/m-files/TestFun.m*. *Now experiment with the effects of changing* $c_1$ *and* $c_2$. *Edit* `TestFun` *to change* $\phi(\alpha)$.

*See Exercise 10 for some trickier examples.*

- A nice result which I will now prove is that, provided that $p$ is a descent direction, there are always $\alpha$–values that satisfy the Wolfe conditions.
- (Of course, I still have to find them.)

**Lemma 4.1** *Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is $C^1$. Let $p_k$ be a descent direction at $x_k$ and assume that $f$ is bounded below along the ray $\{x_k + \alpha p_k | \alpha > 0\}$. Then if $0 < c_1 < c_2 < 1$ there exists at least one interval of step lengths satisfying the Wolfe conditions Eqs. 4.9a and 4.9b.*

**Proof:** (You will find that a sketch graph makes the proof easy to follow.)

Since $\phi(\alpha) = f(x_k + \alpha p_k)$ is bounded below for all $\alpha > 0$ and since $0 < c_1 < 1$, the line $l(\alpha) = \phi(0) + c_1 \alpha \phi'(0)$ must intersect the graph of $\phi$ at least once (as $l$ decreases without limit and $\phi$ does not).

Let $\alpha' > 0$ be the **first** intersecting value of $\alpha$, i.e.

$$\phi(\alpha') = \phi(0) + \alpha' c_1 \phi'(0). \tag{4.13}$$

The sufficient decrease condition Eq. 4.9a must hold for **all** smaller step lengths $\alpha < \alpha'$ — as, for small $\alpha$, $\phi(\alpha) < l(\alpha)$ as $\phi$ initially decreases more rapidly than $l$.

By the Mean Value Theorem, there exists $\alpha'' \in (0, \alpha')$ such that

$$\phi(\alpha') - \phi(0) = (\alpha' - 0)\phi'(\alpha'') \equiv \alpha'\phi'(\alpha'') \tag{4.14}$$

Combining Eqs. 4.13 and 4.14, I have

$$\phi'(\alpha'') = c_1\phi'(0) > c_2\phi'(0), \tag{4.15}$$

since $c_2 > c_1 > 0$ and $\phi'(0) < 0$.

Therefore $\alpha''$ satisfies the Wolfe conditions Eqs. 4.9a and 4.9b.

By the smoothness assumption on $f$, there is an interval around $\alpha''$ for which the Wolfe conditions hold.

Finally, as the term in the left hand side of Eq. 4.15 is negative, the strong Wolfe conditions Eqs. 4.10a and 4.10b hold in the same interval. ■

### 4.2.2 Sufficient Decrease & Backtracking

- I have shown that the sufficient decrease condition Eq. 4.9a alone is not sufficient to ensure that the algorithm makes reasonable progress along the given search direction.
- However, if the line search algorithm chooses its candidate step lengths appropriately, by using a **backtracking** approach, I can dispense with Eq. 4.9b and use just the sufficient decrease condition to terminate the line search procedure.
- In its simplest form, backtracking takes the form:

**Algorithm 4.1 (Backtracking Line Search)**

```
begin
    Choose $\bar{\alpha} > 0$ and $\rho, c \in (0, 1)$
    $\alpha := \bar{\alpha}$
    while $\phi(\alpha) \geq \phi(0) + c\alpha\phi'(0)$ do
        $\alpha := \rho\alpha$
    end
    $\alpha_k := \alpha$
end
```

- This is the simple line search algorithm implemented in http://jkcray.maths.ul.ie/ms4327/m-files/SimpleLineSearch/sls.m.

- In this algorithm, the initial step length $\bar{\alpha}$ is chosen to be $1$ in Newton-type methods but may be smaller in algorithms such as steepest descent or nonlinear conjugate gradient methods.
- An acceptable step length will be found after a finite number of trials as $\alpha_k$ will eventually become small enough that the sufficient decrease condition will hold.
- The backtracking approach ensures that either the selected step length $\alpha_k$ is some fixed value (the initial choice $\bar{\alpha}$) or else that it is short enough to satisfy the sufficient decrease condition — but not <u>too</u> short.
- The latter follows as the accepted value $\alpha_k$ is close to the previous value which was rejected for being too long!

- A nice feature of this simple algorithm is that it is guaranteed to produce an interval which satisfies the Wolfe conditions.

- To make it clear what is going on; a Theorem:

  **Theorem 4.2** *Provided at least one iteration takes place, then for* $\rho$ *sufficiently close to* $1$ *(i.e. provided the backtracking is sufficiently slow) the algorithm Alg 4.1 will produce an interval* $I = [\alpha_1, \alpha_2]$ *such that the Wolfe conditions Eqs. 4.9a and 4.9b are satisfied for all* $\alpha \in I$ *with* $c_1 = c$ *and for some* $c_2 > c_1$*.*

  **Proof:** See Appendix A.9. ■

- Obviously I don't know in advance how close to $1$ the constant $\rho$ needs to be — so this simple algorithm isn't as bulletproof as it appears.

- Some extra code is included in the m-file to ensure that I have a starting value of $\alpha$ that violates the First Wolfe condition.
- The m-file also includes an outer loop that makes $\rho$ closer to 1 and restarts the “stepping to the left” process if it fails to find an $\alpha$–value satisfying the two Wolfe conditions.

## 4.3 Are Search Direction Methods Guaranteed To Converge?

- To ensure that an algorithm converges to a point $x$ where $\nabla f(x) = 0$, I need not only well-chosen step lengths but also well-chosen search directions $p_k$ .

- To improve readability I will write $\nabla f_k$ as $g_k$ from now on.

- I focus in this section on a key parameter, the angle $\theta_k$ between $p_k$ and the steepest descent direction $-g_k \equiv -\nabla f(x_k)$ defined by
$$\cos\theta_k = \frac{-{p_k}^T g_k}{\|g_k\|\,\|p_k\|}. \tag{4.16}$$

- It turns out that when the Wolfe conditions hold I can prove convergence for almost any line search method — irrespective of the details of the algorithm.

- I need a Theorem that gives sufficient conditions for convergence.
- First a technical definition:

  **Definition 4.1 (Lipschitz Continuity)** *A function* $f : \mathbb{R}^n \to \mathbb{R}$ *is* **Lipschitz continuous** *on an open set* $\mathcal{N}$ *if there exists a constant* $L$ *such that*

  $$|f(x) - f(y)| \leq L\|x - y\|, \quad \textit{for all } x, y \in \mathcal{N}. \tag{4.17}$$

  *A Lipschitz continuous function is certainly continuous.*
- Loosely speaking, this means that the function $f$ cannot have a rate of change greater than the constant $L$ — the directional derivative (when defined) along any direction from a given point is bounded between $-L$ and $L$.
- On $\mathbb{R}$, the slope (at any point where it is defined) can never be greater than $L$ or less than $-L$.

- If the gradient $g(x) = \nabla f(x)$ is Lipschitz continuous on an open set $\mathcal{N}$ then the function $f$ is certainly $C^1$ on $\mathcal{N}$.
- I use the change in the norm of the gradient $\|g(x) - g(y)\|$ in the definition above in this case.
- In addition if the gradient vector $g(x)$ is Lipschitz continuous then $g(x)$ cannot change faster than the constant $L$.
- **Exercise 4.5** *Make a sketch to illustrate the above.*

- If the Hessian of $f$ has eigenvalues bounded in magnitude, in particular if

$$\|\nabla^2 f(x)\| \leq M, \forall x \in \mathcal{N}, \tag{4.18}$$

(the 2-norm of a symmetric matrix is the magnitude of its largest eigenvalue) then I can use Taylor's Thm. 3.7 to show that the gradient $\nabla f$ is Lipschitz continuous.

- So Lipschitz continuity of the gradient is a weaker condition than (4.18) — I am not asking so much of the function $f$.
- See App. A.8 for the details.

**Theorem 4.3 (Zoutendijk)** *Consider any iteration of the form 4.1, where*

- $p_k$ *is a descent direction*
- $\alpha_k$ *satisfies the Wolfe conditions Eqs. 4.9a and 4.9b*
- $f$ *is bounded below in* $\mathbb{R}^n$
- *that the gradient* $g \equiv \nabla f$ *is* **Lipschitz continuous** *in an open set* $\mathcal{N}$ *containing the level set* $\mathcal{L} \equiv \{x : f(x) \leq f(x_0)\}$, *where* $x_0$ *is the starting point.*

*Then*

$$\sum_{k \geq 0} \cos^2 \theta_k \, \|g_k\|^2 < \infty \quad \textit{(The sum converges.)} \tag{4.19}$$

- Before proving the Theorem — let's see why it is so important.
- The inequality Eq. 4.19, referred to as the **Zoutendijk condition**, implies that

$$\cos^2\theta_k\,\|g_k\|^2 \to 0. \tag{4.20}$$

- This limit implies global convergence for the various line search algorithms, **provided** that my method for choosing the search direction $p_k$ at each iteration ensures that $\cos^2\theta_k$ does **not** converge to zero — or equivalently that the angle $\theta_k$ defined in 4.16 is bounded away from $90°$.
- This follows because if there is a positive constant $\delta$ such that $\cos\theta_k \geq \delta > 0$ for all $k$, then Eq. 4.20 immediately implies that

$$\lim_{k\to\infty}\|g_k\| = 0. \tag{4.21}$$

- So I know that the gradient norms converge to zero, provided that the search directions are never too close to being perpendicular to the gradient.
- In particular, the steepest descent method is guaranteed to produce a sequence of gradients which converge to zero, **provided that the line search satisfies the Wolfe conditions**.
- This shows the importance of the Wolfe conditions.

- Finally; the assumptions in the Theorem are not overly restrictive.
  - If $f$ were not bounded below, the problem would not have a solution.
  - As discussed above, the Lipschitz condition 4.17 on the gradient (essentially a smoothness assumption about $f$) holds — for example — if $f$ is $C^2$ and the Hessian is uniformly bounded in norm (so that the maximum eigenvalue is less than some constant).

- I now present the proof of Theorem 4.3, ( Zoutendijk's).

**Proof:** From the Second Wolfe condition Eq. 4.9b and from the update rule 4.1,I have that

$$(g_{k+1} - g_k)^T p_k \geq (c_2 - 1) g_k{}^T p_k,$$

while the Lipschitz condition 4.17 implies that

$$(g_{k+1} - g_k)^T p_k \leq \alpha_k L \|p_k\|^2.$$

Combining these two inequalities, I find that

$$\alpha_k \geq \frac{c_2 - 1}{L} \frac{g_k{}^T p_k}{\|p_k\|^2}.$$

Substituting this inequality into the first Wolfe condition Eq. 4.9a, I have

$$f_{k+1} \leq f_k - c_1 \frac{1 - c_2}{L} \frac{(g_k{}^T p_k)^2}{\|p_k\|^2}.$$

From the definition 4.16 of $\theta$ I write this relation as

$$f_{k+1} \le f_k - c\cos^2\theta_k \|g_k\|^2,$$

where $c = c_1(1-c_2)/L$. By summing this expression over all indices less than or equal to $k$, I have

$$f_{k+1} \le f_0 - c\sum_{j=0}^{k} \cos^2\theta_j \|g_j\|^2. \tag{4.22}$$

As $f$ is bounded below, I know that $f_0 - f_{k+1}$ is less than some positive constant, for all $k$.

So the sum from $0$ to $k$ is bounded for all $k$.

But if a sequence is bounded above and is non-decreasing then it has a limit. So, taking the limit as $k \to \infty$ in 4.22, I have

$$\sum_{j=0}^{\infty} \cos^2\theta_j \|g_j\|^2 < \infty. \qquad \blacksquare$$

- I use the term **globally convergent** to refer to algorithms for which the property 4.21 holds.
- In other words, globally convergent algorithms are ones which converge to a stationary point.
- Unfortunately, instead of a minimum point I could converge to a **maximum** point which is also a stationary point!
- To ensure that this does not happen I need "second-order" restrictions on the search direction $p_k$ .
- Remember that a necessary condition for $x^*$ to be a minimum point is that the Hessian $\nabla^2 f(x^*)$ be positive semidefinite (Thm. 3.4).

### 4.3.1 A Class of Methods Which Satisfy the Requirements for Zoutendijk's Thm.

- Consider the Newton-like method (4.1, 4.2) . Assume that the matrices $B_k$ are positive definite with a uniformly bounded condition number, i.e. that

$$\|B_k\| \, \|B_k^{-1}\| \leq M, \quad \text{for all } k.$$

- This is a desirable property in any case as it ensures that the errors that result from solving the Newton equation $Bp = -g$ are also bounded.

- In Appendix A.19 I give the definition of the standard Euclidean norm of a square matrix B and show that it reduces to $\|B\| = \max_{i=1,\dots,n} \sqrt{\Lambda_i}$ where $\Lambda_i$ are the eigenvalues of the symmetric positive semidefinite matrix $B^T B$.

- In fact, when $B$ is symmetric (as the Hessian is), the formula simplifies to $\|B\| = \max_{i=1,\dots,n} |\lambda_i|$ — the largest of the eigenvalues of $B$ in absolute value.
- So
$$\|B_k\| \, \|B_k^{-1}\| \leq M, \quad \text{for all } k.$$
is equivalent to
$$\frac{|\lambda_{\max}|}{|\lambda_{\min}|} \leq M, \quad \text{for all } k.$$
- Which means that the largest eigenvalue never gets too big and the smallest eigenvalue never gets too small.

- Then it is easy to check that (see Exercise 4 on Slide 150 at the end of this Chapter):

$$\cos \theta_k \geq 1/M. \tag{4.23}$$

- Combining this bound with 4.20 I again find that $\lim_{k \to \infty} \|g_k\| = 0.$
- For conjugate gradient methods (see Ch. 5), obviously the result (4.23) above may not hold as information about the Hessian is not usually available.
- A weaker result can be proved; namely that a subsequence of the gradient norms converges to zero.
- In practice this is good enough — why?

## 4.4 How Quickly Do Search Direction Methods Converge?

- The conclusion from the previous Section is that to ensure convergence I need to choose my search directions so that the angles $\theta_k$ do not converge to $\pi/2$ as $k \to \infty$.

- However, this is no guarantee of the speed or **rate** of convergence — I want an algorithm that converges to a stationary point (where $g = 0$) as fast as possible.

### 4.4.1 Convergence Rate for Steepest Descent Method

- The special case of the steepest descent method for a quadratic problem with exact line searches turns out to be instructive.
- Let the quadratic objective $f(x)$ be given by:

$$f(x) = \frac{1}{2}x^T Q x - b^T x, \tag{4.24}$$

  where $Q$ is symmetric and positive definite.
- The gradient is just $g(x) = Qx - b$ and the minimiser $x^*$ is the unique solution of the linear system $Qx = b$.

- The step length that **exactly** minimises $f(x_k - \alpha g_k)$ must satisfy

$$\frac{d}{d\alpha} f(x - \alpha g_k) = 0.$$

- Using the Chain Rule it is easy to see that

$$g_k{}^T g(x_k - \alpha g_k) = 0 \tag{4.25}$$

and so, using $g(x) = Qx - b$, I have

$$0 = g_k{}^T \Big( Q(x_k - \alpha g_k) - b \Big)$$
$$0 = g_k{}^T \Big( g_k - \alpha Q g_k \Big).$$

- So I can solve for $\alpha$, giving

$$\alpha_k = \frac{g_k{}^T g_k}{g_k{}^T Q g_k}. \tag{4.26}$$

- If I use this exact minimiser $\alpha_k$ , the steepest descent iteration for 4.24 is

$$x_{k+1} = x_k - \left( \frac{{g_k}^T g_k}{{g_k}^T Q g_k} \right) g_k \,. \tag{4.27}$$

- Note that ${g_k}^T g_{k+1} = 0$ so successive search directions (gradients) are perpendicular.
- This causes the characteristic “zig-zag” behaviour of the steepest descent method — see Figure 9.
- The objective will usually be “approximately quadratic” at least locally and the step length may be close to the exact minimiser $\alpha_k$ — so often ${g_k}^T g_{k+1} \approx 0$ even for a general non-quadratic problem.

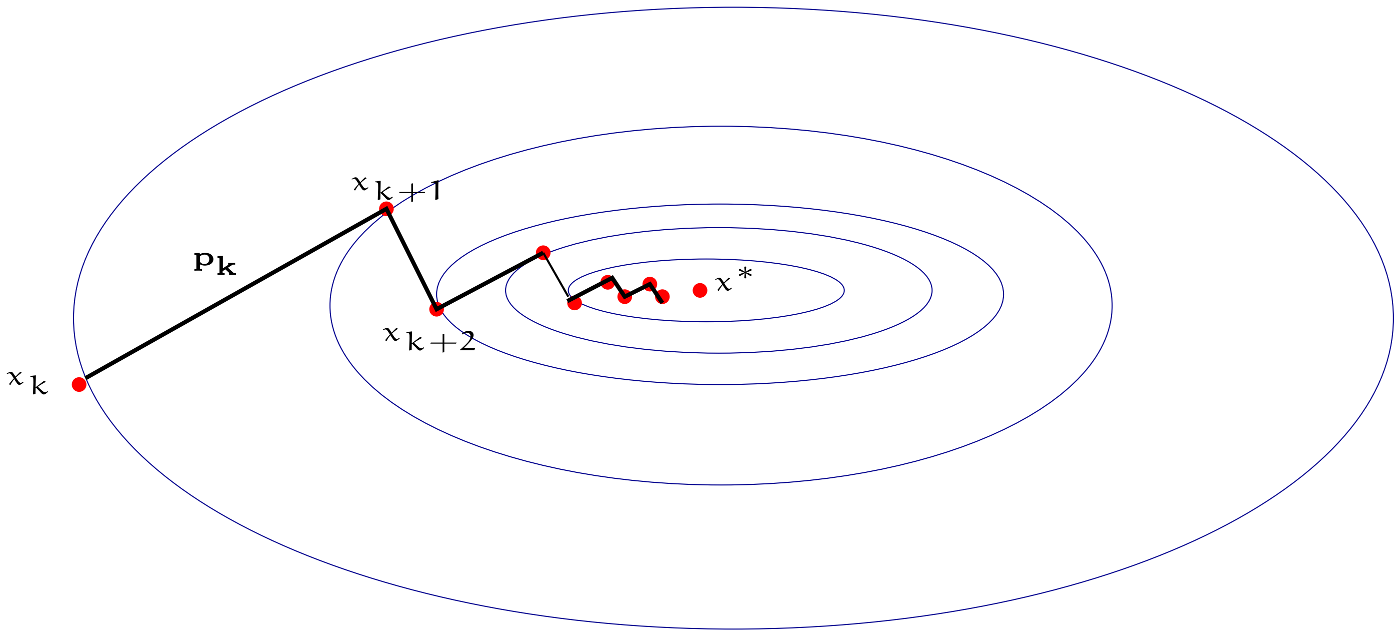


Figure 9: Typical Steepest Descent steps

- To quantify the rate of convergence I introduce the weighted norm $\|x\|_Q^2 \equiv x^T Q x$.
- Using the equation $Qx^* = b$, you should check (by expanding each side) that for any $x$ we have

$$f(x) - f(x^*) = \frac{1}{2}\|x - x^*\|_Q^2 \tag{4.28}$$

- Using 4.27 and noting that $g_k = Qx_k - b \equiv Q(x_k - x^*)$, I can show that (see Ex. 5 on Slide 150):

$$\|x_{k+1} - x^*\|_Q^2 = \left\{1 - \frac{({g_k}^T g_k)^2}{({g_k}^T Q g_k)({g_k}^T Q^{-1} g_k)}\right\} \|x_k - x^*\|_Q^2 \tag{4.29}$$

- It is easier to interpret this result if I bound it in terms of the condition number $\kappa = \dfrac{\lambda_{\text{max}}}{\lambda_{\text{min}}}$.

**Theorem 4.4** *When the steepest descent method with exact line searches is applied to the (convex) quadratic function 4.24, the error norm 4.28 satisfies*

$$\|x_{k+1} - x^*\|_Q^2 \leq \left(\frac{\kappa - 1}{\kappa + 1}\right)^2 \|x_k - x^*\|_Q^2. \qquad (4.30)$$

**Proof:** See Ex. 7 on Slide 151 or see Ex. 8 for a neater proof. ∎

- It follows that the function values $f_k$ converge to the minimum $f_*$ at a **linear** rate.
- If the condition number $\kappa(Q) = \lambda_{\max}/\lambda_{\min}$ is large, then the contours become stretched, the zig-zagging gets worse and the constant factor in Eq. 4.30 $\to 1$ so the convergence slows.

- It can be shown that essentially the same result holds for general objective functions:

  **Theorem 4.5** *Suppose that* $f : \mathbb{R}^n \to \mathbb{R}$ *is* $C^2$ *and that the iterates generated by the steepest descent method with exact line searches converge to a point* $x^*$ *where the Hessian matrix* $\nabla^2 f(x^*)$ *is positive definite. Then*

  $$f(x_{k+1}) - f(x^*) \leq \left( \frac{\kappa - 1}{\kappa + 1} \right) [f(x_k) - f(x^*)] . \tag{4.31}$$

  **Proof:** Not given.

- If the rate of convergence is poor with an exact line search then it isn't going to be any better when an inexact line search is used.

- So the Theorem shows that the steepest descent method can be unacceptably slow, even when the Hessian is reasonably well-conditioned.

  **Exercise 4.6** *If* $\kappa(Q) = 800$, $f(x_1) = 1$ *and* $f(x^*) = 0$, *check that the function value could be still as large as* $0.08$ *— a reduction by a factor of only about* $12$ *— after a thousand iterations of the steepest descent method !*

  *And a condition number* $\kappa = 800$ *is not particularly extreme.*

### 4.4.2 Convergence for Newton's Method

- Consider the Newton iteration where the search direction is given by
$$p_k^N = -{\nabla^2 f_k}^{-1} g_k \,. \tag{4.32}$$
- I restrict myselves to the case where the Hessian $\nabla^2 f_k$ is positive definite — this is a sufficient condition for $p_k$ to be a descent direction.
- In this case I can achieve **quadratic** convergence — as good as it gets.
- This result is stated and proved in the following Theorem.
- For readability we refer to the gradient $\nabla f(x)$ as $g(x)$ and the Hessian $\nabla^2 f(x)$ as $H(x)$ in the following.

**Theorem 4.6** *Let* $f : \mathbb{R}^n \to \mathbb{R}$ *be* $C^3$.

- *Let* $x^*$ *be such that* $g(x^*) = 0$ *(*$x^*$ *a stationary point) and* $H(x^*)$ *is positive definite (so* $x^*$ *is a local minimum point).*
- *Let* $R = \|\mathbf{x}_0 - x^*\|$ *be the distance from the start point* $x_0$ *to* $x^*$.
- *Assume that* $R$ *is small enough for the condition* $R < \frac{1}{k_1 k_2}$ *to hold for some* $k_1, k_2 > 0$ *defined by the two conditions:*
  1. *the inverse Hessian is bounded in norm:* $\|H(x)^{-1}\| \leq k_1$
  2. *the remainder term in a second-order Taylor expansion of* $g(x^*)$ *centred at* $x$ *is uniformly bounded:* $\|g(x^*) - g(x) - H(x)(x^* - x)\| \leq k_2 \|x^* - x\|^2$

  *for each* $x$ *inside the ball* $B = \{\|x - x^*\| \leq R\}$.

- *Condition 1 is equivalent to* $|\lambda_{\min}| \geq 1/k_1$, *so* $|\lambda_{\min}|$ *may be small when* $k_1$ *large.*
  - *If* $k_1 \gg 1$, $H(x)$ *may be close to singular for some* $x$ *values.*
- *If* $k_2 \gg 1$ *then a quadratic approximation to* $g(x^*)$ *is poor.*
- *Note that if* $k_1$ *and* $k_2$ *are large then the start point* $x_0$ *has to be very close to* $x^*$ *(*$R = \|\mathbf{x}_0 - x^*\| < 1/k_1 k_2$*) for the Theorem to apply.*

*Then the algorithm converges to* $x^*$ *with* **quadratic** *convergence.*

**Proof:**

- I have $B = \{x : \|x - x^*\| \leq R = \|x_0 - x^*\|\}$ — the ball centred at $x^*$ with $x_0$ on the boundary.
- I need to show that if $x \in B$ then the new point $x' = x + p$ where $p$ satisfies $H(x)p = -g(x)$ is closer to the centre of the ball than $x$ is.
- Now (assuming $x \neq x^*$) the path from $x^*$ to the new point $x'$ is

$$\begin{aligned} x' - x^* &= (x - x^*) - H^{-1}(x)\left[g(x) - g(x^*)\right] \\ &= H^{-1}(x)\left[g(x^*) - g(x) - H(x)(x^* - x)\right] \end{aligned}$$

since $g(x^*) = 0$.

- So, using the assumptions stated in the Theorem, I have

$$\begin{aligned}
\|x' - x^*\| &= \|H(x)^{-1}\left[g(x^*) - g(x) - H(x)(x^* - x)\right]\| \\
&\leq \|H(x)^{-1}\|\|g(x^*) - g(x) - H(x)(x^* - x)\| \\
&\leq k_1 k_2 \|x - x^*\|^2 \leq k_1 k_2 \|x_0 - x^*\|\|x - x^*\| \\
&< \|x - x^*\|.
\end{aligned}$$

This shows that the new point is always “better” than the old — closer to $x^*$ .

Finally, I have that for any $\mathbf{x}_k \in B$, the new point $\mathbf{x}_{k+1}$ produced by Newton's method satisfies $\|\mathbf{x}_{k+1} - x^*\| \le k_1 k_2 \|\mathbf{x}_k - x^*\|^2$ from the above discussion. Since $\mathbf{x}_k \to x^*$ I have at least quadratic convergence. ■

For Newton's method , the limit 4.33 is automatically satisfied (the ratio is zero for all $k$) and Theorem 4.7 shows that the Wolfe conditions will accept the step length $\alpha_k$ for all large $k$. So when Newton's method is implemented using these conditions and when the line search always tries the unit step length first; the algorithm will set $\alpha_k = 1$ for all large $k$ and attain a quadratic rate of convergence.

### 4.4.3 Convergence for Quasi-Newton Methods

Finally suppose that the search direction $p_k$ is defined by 4.2 where the symmetric and positive definite matrix $B_k$ is updated at each iteration by a quasi-Newton update formula (such as BFGS). I will assume that the step length $\alpha_k$ will be computed by an inexact line search that satisfies the Wolfe or strong Wolfe conditions with the addition that the step length $\alpha = 1$ will always be tried first and accepted if it satisfies the Wolfe conditions.

It can be shown that if the search direction of a quasi-Newton method approximates the Newton direction well enough, then the unit step length will satisfy the Wolfe condition as the iterates converge to the solution. I will also state a result which gives a condition that the search direction must satisfy in order to give rise to **superlinear** convergence.

**Theorem 4.7** *Suppose that* $f : \mathbb{R}^n \to \mathbb{R}$ *is* $C^3$. *Consider the iteration* $x_{k+1} = x_k + \alpha_k p_k$, *where* $p_k$ *is a descent direction and* $\alpha_k$ *satisfies the Wolfe conditions with* $c_1 < \frac{1}{2}$. *If the sequence* $\{x_k\}$ *converge to a point* $x^*$ *such that* $g(x^*) = 0$ *and* $\nabla^2 f(x^*)$ *is positive definite and if the search direction satisfies*

$$\lim_{k \to \infty} \frac{\|g_k + \nabla^2 f_k p_k\|}{\|p_k\|} = 0, \tag{4.33}$$

*then*

- *(i) the step length* $\alpha_k = 1$ *is admissable for all* $k$ *greater than some* $k_0$ *and*
- *(ii) if* $\alpha_k = 1$ *for all* $k > k_0$ *then* $\{x_k\}$ *converges to* $x^*$ **superlinearly**.

If $p_k$ is a quasi-Newton method search direction then 4.33 is equivalent to

$$\lim_{k\to\infty} \frac{\|(B_k - \nabla^2 f_k)p_k\|}{\|p_k\|} = 0, \tag{4.34}$$

so superlinear convergence can be achieved provided that the sequence of $B_k$ become increasingly accurate approximations to $\nabla^2 f_k$ **along the direction $p_k$** .

**Proof:** Not given.

## 4.5 When To Stop

- The question of when to accept an approximate solution to a minimisation problem at first seems easy to answer: I stop when the gradient is zero — or in practice when $\|g(x)\|$ is sufficiently small.
- This guarantees that I am at or near a stationary point.
- If the Hessian is positive definite  I know that I am at or near a local minimum.
- But: should I monitor the change in $x$ from iteration to iteration too? (After all if $x$ is not changing appreciably then no progress is being made.)
- And what about the change in the objective function? (Again, if $f(x)$ is "stuck" then I might as well stop?)

- I need to sort out the relationship between $|\Delta f|$, $\|\Delta x\|$ and $\|g(x)\|$.
- The following Theorem clarifies this relationship.
- For readability I again refer to the gradient $\nabla f(x)$ as $g(x)$ and the Hessian $\nabla^2 f(x)$ as $H(x)$ in the following.

**Theorem 4.8** *Let* $f \in C^2(\mathbb{R})$*. Let* $x^*$ *be a local minimum for* $f$ *on for an open ball* $\mathcal{B}(x^*)$ *centred at* $x^*$ *(of radius* $R$*, say).*

*Suppose that there is a number* $m > 0$ *such that for all* $x$ *in the ball* $\mathcal{B}(x^*)$

$$m\|d\|^2 \leq d^\mathsf{T} H(x) d, \quad \forall d \in \mathbb{R}^n.$$

*(In words, the inverse of the Hessian* $f$ *has uniformly bounded 2-norm on* $\mathcal{B}(x^*)$*.)*

*Then,* $\forall x \in \mathcal{B}$*,*

$$\|x - x^*\| \leq \frac{\|g(x)\|}{m} \tag{4.35}$$

$$f(x) - f(x^*) \leq \frac{\|g(x)\|^2}{m}. \tag{4.36}$$

**Proof:**

- Taylor's Thm. 3.7 gives:

$$g(x) - g(x^*) = \int_0^1 H\left(x^* + t(x - x^*)\right) \quad (x - x^*)dt.$$

- As $g(x^*) \equiv 0$, I have that for any $x \in \mathcal{B}$,

$$\begin{aligned} (x - x^*)^T g(x) &= \int_0^1 (x - x^*)^T H\left(x^* + t(x - x^*)\right) \quad (x - x^*)dt \\ &\geq m \int_0^1 \|x - x^*\|^2 dt \\ &= m\|x - x^*\|^2. \end{aligned}$$

- Using the Cauchy-Schwartz Inequality;

$$(x - x^*)^T g(x) \leq \|x - x^*\| \|g(x)\|,$$

and so $m\|(x - x^*)\|^2 \leq \|(x - x^*)\| \|g(x)\|$ which proves (4.35).

- Now define $F(t) = f(x^* + t(x - x^*))$ for $t \in [0, 1]$.
- Then $F'(t) = (x - x^*)^T g(x^* + t(x - x^*))$ and $F''(t) = (x - x^*)^T H(x^* + t(x - x^*))(x - x^*) \geq m\|(x - x^*)\|^2 \geq 0$.
- So $F'(t)$ is a non-decreasing function of $t$ and so $F'(1) \geq F'(t)$ for all $t \in [0, 1]$.
- Obviously $\int_0^1 F'(t)dt \equiv F(1) - F(0) \equiv f(x) - f(x^*)$ for $t \in [0, 1]$.
- But the integral is bounded above by $F'(1) \equiv (x - x^*)^T g(x) \ \leq \|x - x^*\| \|g(x)\|$ (again by the C-S inequality).
- The result (4.36) follows using (4.35) to eliminate $\|x - x^*\|$ .

■

The significance of the Theorem is easy to extract.

- If I place a bound on $\|g(x)\|$ as a stopping criterion, then I have implicitly stated (at least approximately) stopping criteria for $\|\Delta x\|$ and $|\Delta f(x)|$.
- In detail, if my stopping criterion is $\|g(x)\| \leq \varepsilon$ for some small quantity $\varepsilon$, then (if $x'$ is the current estimate and $x$ the previous one) by the triangle inequality $\|x - x'\| \leq \|x - x^*\| + \|x' - x^*\|$.
- The Theorem tells us that if the stopping criterion on the gradient held at $x$ and $x'$ then
$$\|\Delta x\| \equiv \|x - x'\| \leq \frac{2\varepsilon}{m} \tag{4.37}$$
- Similarly,
$$|\Delta f| \equiv |f(x) - f(x')| \leq \frac{2\varepsilon^2}{m}. \tag{4.38}$$

- Or, to turn things around, the most demanding stopping criterion that I can reasonably impose for $|\Delta f|$ is $|\Delta f| \leq K\varepsilon$ (where $K$ is some constant such as $10$ or $|f(x_0)|$ and $\varepsilon = \epsilon_M$ — machine epsilon, about $10^{-16}$).
- It follows that it only makes sense to bound $\|\nabla f\|$ and $\|\Delta x\|$ by multiples of $\sqrt{\varepsilon}$, about $10^{-8}$.
- Finally, the above can be easily extended to apply to $\Delta\phi$, $\Delta\alpha$ and $|\phi'|$

## 4.6 Exercises

1. Use Matlab (or Maple) to program the steepest descent method and Newton's method algorithms using the backtracking line search , Algorithm 4.1. Use them to minimise the Rosenbrock function 3.25 on Slide 77. Set the initial step length $\alpha_0 = 1$ and print the step length used by each method at each iteration. First try the initial point $x_0 = (1.2, 1.2)$ and then the more difficult point $x_0 = (-1.2, 1)$.

2. Show that if $c_1 > \frac{1}{2}$, a line search would exclude the minimiser of a quadratic so the restriction $c_1 \leq \frac{1}{2}$ is reasonable.

3. Show that the one-dimensional minimiser of a strongly convex quadratic function is given by $\alpha_k = -\dfrac{{g_k}^T p_k}{{p_k}^T Q p_k}$.

4. Derive Eq. 4.23 on Slide 123. Hint: use the fact that as $\|B\| = |\lambda_{\max}|$ I can write $\|B^{-1}\| = \frac{1}{|\lambda_{\min}|}$. (See Appendix A.10 for a solution.)

5. Prove the result Eq. 4.29 on Slide 129 using the following steps:

   (A) Use Eq. 4.27 to show that

   $$\begin{aligned}\|x_k - x^*\|_Q^2 - \|x_{k+1} - x^*\|_Q^2 = 2\alpha_k {g_k}^T Q(x_k - x^*) \\ -{\alpha_k}^2 {g_k}^T Q g_k\,.\end{aligned}$$

   where the Q-norm is defined as before.

(B) Using the formula (4.26) for $\alpha_k$ and the fact that $g_k = Q(x_k - x^*)$ to show that

$$\|x_k - x^*\|_Q^2 - \|x_{k+1} - x^*\|_Q^2 = \frac{(g_k^T g_k)^2}{(g_k^T Q g_k)}.$$

and

$$\|x_k - x^*\|_Q^2 = g_k{}^T Q^{-1} g_k$$

6. Let Q be a positive definite symmetric matrix. Prove that for any vector $x$,

$$\frac{(x^T x)^2}{(x^T Q x)(x^T Q^{-1} x)} \geq \frac{4\lambda_{\max}\lambda_{\min}}{(\lambda_{\max} + \lambda_{\min})^2}.$$

(This relation is called the Kantorovitch inequality and a proof is given in Appendix A.11.)

7. Use the Kantorovitch inequality to deduce Eq. 4.30 from Eq. 4.29.

8. By looking at the special case where $f(x) = \frac{1}{2}x^TQx$ (no linear term), prove Eq. 4.30 without using the Kantorovitch inequality. See Appendix A.12 for details.

9. Program the BFGS algorithm using the line search algorithm described in this Chapter which implements the strong Wolfe conditions . Use your program to re-solve the problem at the end of Exercise 1.

10. Find the range of $\alpha$–values that satisfy the strong Wolfe conditions for the following functions:

$$\text{(i)} \cos\alpha \quad \text{(ii)} 1 - \sin\alpha/10$$
$$\text{(iii)} (\alpha - 1)^2 \quad \text{(iv)} \alpha^3 + \alpha^2 - \alpha + 1$$

(Use the Matlab function m-file `http://jkcray.maths.ul.ie/ms4327/m-files/WolfeGen.m` to plot and solve the problem before trying an algebraic solution.)

# 5 Conjugate Gradient Methods

The conjugate gradient method can be viewed in two ways; it is a very effective method for solving large linear systems of equations and it can (as I will show) be adapted to solve (in particular very large) nonlinear optimisation problems.

In this course I have little or no interest in solving linear systems (that is what Linear Algebra 2 was mainly about).

Despite this, because of:

1. the historical development of conjugate gradient methods
2. and the fact that the nonlinear conjugate gradient method follows naturally from the older "linear" version designed for the solution of linear systems,

the Chapter is divided into two Sections, the **linear** conjugate gradient method and the **nonlinear** conjugate gradient method. In the first Section the linear conjugate gradient method & its properties are examined. I will move quickly though the Section noting the results without examining the proofs. **This material is for reference only.**

The second Section on the nonlinear conjugate gradient method will examine the extensions of the linear conjugate gradient method to non-linear problems, our real objective.

## 5.1 Linear Conjugate Gradient Method

In the following I will often drop the word “linear” for brevity. The conjugate gradient method is an iterative method for solving a system of linear equations

$$Ax = b, \tag{5.1}$$

where $A$ is $n \times n$, symmetric & positive definite. The problem can be restated as the following minimisation problem:

$$\min \phi(x) = \frac{1}{2} x^T A x - b^T x \tag{5.2}$$

Note that the gradient of $\phi$ is the residual or error $r(x)$ of the linear system:

$$\nabla \phi(x) = Ax - b \equiv r(x). \tag{5.3}$$

and that the Hessian of $\phi$ is just the constant matrix $A$.

### 5.1.1 The Conjugate Gradient Method

I take the starting search direction to be $p_0 = -\nabla f(x_0)$ — the obvious choice. I can summarise the algorithm as follows:

## Algorithm 5.1 (linear CGM)

```
begin
    Given $x_0$.
    set    $r_0 \leftarrow Ax_0 - b, p_0 \leftarrow -r_0, k \leftarrow 0;$
    while $r_k \neq 0$ do
            $\alpha_k \leftarrow \frac{{r_k}^T r_k}{p_k^T A p_k} \equiv \frac{\|r_k\|^2}{p_k^T A p_k};$
            $x_{k+1} \leftarrow x_k + \alpha_k p_k;$
            $r_{k+1} \leftarrow r_k + \alpha_k A p_k;$
            $\beta_{k+1} \leftarrow \frac{r_{k+1}^T r_{k+1}}{{r_k}^T r_k} \equiv \frac{\|r_{k+1}\|^2}{\|r_k\|^2};$
            $p_{k+1} \leftarrow -r_{k+1} + \beta_{k+1} p_k;$
            $k \leftarrow k + 1;$
    end    (while)
end
```

At any given point, I never need to know the vectors $x$, $r$ and $p$ for more than two iterations. The other advantage of the linear conjugate gradient method is that the matrix $A$ is not changed during execution so if the matrix has desireable properties such as sparsity, they are preserved.

Before moving on to the major topic of this Chapter, the nonlinear conjugate gradient method , I'll examine the convergence properties of the linear conjugate gradient method – the reasoning being: I cannot hope that the conjugate gradient method will perform better on general non-linear problems than on the simple quadratic problem so the results below provide us with an "upper bound" on the convergence for general nonlinear problems.

The crucial result is Thm 5.1 — essentially nonlinear conjugate gradient method converges linearly like the steepest descent method – but with a smaller "constant factor".

A final convergence result — useful if only an estimate of the condition number $\kappa = \dfrac{\lambda_n}{\lambda_1}$ is available. Notice that while it is similar to (4.30) on Slide 130 for the Steepest Descent Method, the presence of the square roots means the constant factor is smaller (check) so that the conjugate gradient method will converge faster.

**Theorem 5.1** *If* $A$ *has non-negative eigenvalues* $0 < \lambda_1 \le \lambda_2 \le \cdots \le \lambda_n$, *I have for any* $k < n$ *that*

$$\|x_{k+1} - x^*\| \le \left( \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right) \|x_k - x^*\|. \tag{5.4}$$

## 5.2 Nonlinear Conjugate Gradient Methods

I now extend the linear conjugate gradient method to nonlinear optimisation problems.

### 5.2.1 Fletcher-Reeves Method

Fletcher & Reeves showed that an extension of this kind is possible if I make two simple changes in Alg. (5.1). First, I need to replace the expression for $\alpha_k$ by a line search along $p_k$ . Second, the residual $r_k$ (which is just the gradient of $\phi$ for linear problems) must be replaced by the gradient of the nonlinear objective function $f$.

### Algorithm 5.2 (FR–CGM)

```
begin
    Given $x_0$.
    set  $r_0 \leftarrow \nabla f_0, p_0 \leftarrow -r_0, k \leftarrow 0;$
    while $r_k \neq 0$ do
        $\alpha_k \leftarrow$ Result of line search along $p_k$ ;
        $x_{k+1} \leftarrow x_k + \alpha_k p_k;$
        $r_{k+1} \leftarrow \nabla f_{k+1}$
        $\beta^{FR}_{k+1} \leftarrow \frac{r^T_{k+1} r_{k+1}}{r_k{}^T r_k} \equiv \frac{\|r_{k+1}\|^2}{\|r_k\|^2};$
        $p_{k+1} \leftarrow -r_{k+1} + \beta^{FR}_{k+1} p_k;$
        $k \leftarrow k + 1;$
    end  (while)
end
```

**Fletcher-Reeves Method – Descent Directions** Of course if $f$ is a quadratic function with positive definite Hessian, then (if I take $\alpha_k$ to be the exact line minimiser) Alg. (5.2) reduces to Alg. (5.1).

I need to be more specific about the choice of $\alpha_k$. Because of the second term in Line (9), the search direction $p_k$ could fail to be a descent direction. Taking the inner product (with $k$ replacing $k+1$) with the gradient $g_k$, I get

$$g_k^T p_k = -\|g_k\|^2 + \beta_k^{FR} g_k^T p_{k-1}. \tag{5.5}$$

If the line search is exact then the second term vanishes and $p_k$ is a descent direction.

Otherwise I need to appeal to the **strong** Wolfe conditions Eqs. (4.10a) and (4.10b) studied in Ch. 4.

I will show in Theorem 5.2 below that the strong second Wolfe condition (4.10b) implies that (5.5) is negative and so $p_k$ is a descent direction.

The following result gives conditions on the line search that guarantee that all search directions generated by the conjugate gradient method are descent directions . It assumes that the set $\mathcal{L} = \{x : f(x) \leq f(x_0)\}$ is bounded and that $f$ is $C^2$.

**Theorem 5.2** *Suppose that Alg. 5.2 is implemented with a step length* $\alpha_k$ *that satisfies the strong Wolfe conditions with* $0 < c_2 < \frac{1}{2}$. *Then the method generates descent directions* $p_k$ *that satisfy the following inequalities:*

$$-\frac{1}{1-c_2} \leq \frac{p_k^T g_k}{\|g_k\|^2} \leq \frac{2c_2 - 1}{1-c_2}, \textit{for all } k = 0, 1, \ldots . \tag{5.6}$$

**Proof:**

It is easy to check that if $0 < c_2 < \frac{1}{2}$, I have

$$-1 \leq \frac{2c_2 - 1}{1 - c_2} < 0. \tag{5.7}$$

(The descent direction condition will follow immediately once I prove (5.6)).

I now prove (5.6)) by induction.

**[Base Step]** For $k = 0$, the middle term in (5.6) is $-1$ as $p_0 \equiv -\nabla f_0 \equiv -g_0$ so by using (5.7) both inequalities in (5.6) are satisfied.

**[Inductive Step]** Assume that (5.6) holds for some $k \geq 1$. First I note that **if** a vector $p_k$ is a descent direction then by Lemma (4.1) there exists a step length $\alpha_k$ that satisfies the strong Wolfe conditions . The direction $p_k$ is a descent direction by the inductive hypothesis , so I conclude that a step length $\alpha_k$ can be found **which satisfies the strong Wolfe conditions** . Then use this $\alpha_k$ to compute $x_{k+1} \equiv x_k + \alpha_k p_k$ & so calculate $p_{k+1}$ and $g_{k+1}$.

From lines (9) and (8) in the FR-CGM algorithm, we have

$$\frac{g_{k+1}^T p_{k+1}}{\|g_{k+1}\|^2} = -1 + \beta_{k+1} \frac{g_{k+1}^T p_k}{\|g_{k+1}\|^2} = -1 + \frac{g_{k+1}^T p_k}{\|g_k\|^2}. \tag{5.8}$$

The strong Wolfe conditions give

$$|g_{k+1}^T p_k| \leq -c_2 g_k^T p_k$$

so, combining with (5.8), I have

$$-1 + c_2 \frac{g_k^T p_k}{\|g_k\|^2} \le \frac{g_{k+1}^T p_{k+1}}{\|g_{k+1}\|^2} \le -1 - c_2 \frac{g_k^T p_k}{\|g_k\|^2}.$$

Finally, substituting for the term $\frac{g_k^T p_k}{\|g_k\|^2}$ from the LHS of the inductive hypothesis (5.6), I get

$$-1 - \frac{c_2}{1 - c_2} \le \frac{g_{k+1}^T p_{k+1}}{\|g_{k+1}\|^2} \le -1 + \frac{c_2}{1 - c_2}.$$

So (5.6) holds for $k + 1$.

By the Principle of Induction, (5.6) holds for all $k$. ■

**Fletcher-Reeves Method — Global Convergence** Global convergence can be proved for the FR method — i.e. "the gradient goes to zero". The proof is a little tricky but is worth the effort as it is typical of the proofs that I will show later for other verions of the nonlinear conjugate gradient method.

(I write $\nabla f(x_k) = g_k$ for readibility.)

**Theorem 5.3** *If Alg. (5.2) is implemented with a line search which satisfies the strong Wolfe conditions with $0 < c_1 < c_2 < \frac{1}{2}$) then under reasonable assumptions on $f$ (the same as in Zoutendijk's Thm 4.3)*

$$\liminf_{k\to\infty} \|g_k\| = 0.$$

This means that while the gradient may not have a limit of zero, it is not bounded away from zero, i.e., picking any $\varepsilon > 0$, for $K$ sufficiently large, there is some $k > K$ for which $\|g_k\| < \varepsilon$.

**Proof:** Assume that the conclusion is false, so that $\|g_k\|$ is bounded away from zero — so that I must have $\|g_k\| \geq \gamma > 0$ for all $k$. I have

$$\cos(\theta_k) = \frac{-g_k^T p_k}{\|g_k\| \|p_k\|}$$

so (5.6) from Thm 5.2 can be written:

$$\frac{(1-2c_2)}{(1-c_2)} \frac{\|g_k\|}{\|p_k\|} \leq \cos(\theta_k) \leq \frac{1}{(1-c_2)} \frac{\|g_k\|}{\|p_k\|}. \tag{5.9}$$

I also have the Zoutendijk condition

$$\sum_{k=0}^{\infty} \cos(\theta_k)^2 \|g_k\|^2 < \infty.$$

Using the lefthand inequality in (5.9) I can write $\cos(\theta_k)^2 \geq \left(\frac{1-2c_2}{1-c_2}\right)^2 \frac{\|g_k\|^2}{\|p_k\|^2}$ and so I can restate the Zoutendijk condition as:

$$\sum \frac{\|g_k\|^4}{\|p_k\|^2} < \infty. \tag{5.10}$$

(We'll use this to get a contradiction at the end of the proof.)

The Strong version of the Second Wolfe condition can be written:

$$|g_{k+1}^T p_k| \leq -c_2 g_k^T p_k$$

so again referring to (5.6) I have (note change from $k+1$ to $k$)

$$|g_k^T p_{k-1}| \leq -c_2 g_{k-1}^T p_{k-1} \leq \frac{c_2}{1-c_2} \|g_{k-1}\|^2. \tag{5.11}$$

Now referring to the nonlinear conjugate gradient method update rule $p_k = -g_k + \beta_k^{FR} p_{k-1}$ and taking the squared norm of each side, I have

$$\begin{aligned}
\|p_k\|^2 &= \|g_k\|^2 + \beta_k^2 \|p_{k-1}\|^2 - 2\beta_k g_k^T p_{k-1} \\
&\le \|g_k\|^2 + \beta_k^2 \|p_{k-1}\|^2 + 2\beta_k |g_k^T p_{k-1}| \\
&\le \|g_k\|^2 + \beta_k^2 \|p_{k-1}\|^2 + 2\frac{c_2}{1-c_2}\beta_k \|g_{k-1}\|^2 \quad \text{by (5.11)} \\
&= \frac{1+c_2}{1-c_2}\|g_k\|^2 + \beta_k^2 \|p_{k-1}\|^2 \quad \text{using def. of } \beta_k.
\end{aligned}$$

So I have (with $c_3 = \frac{1+c_2}{1-c_2}$)

$$\|p_k\|^2 \le c_3\|g_k\|^2 + \beta_k^2\|p_{k-1}\|^2.$$

Iterating this inequality twice (again using the def. of $\beta_k$);

$$\|p_k\|^2 \le c_3\left\{\|g_k\|^2 + \frac{\|g_k\|^4}{\|g_{k-1}\|^2} + \frac{\|g_k\|^4}{\|g_{k-2}\|^2}\right\} + \frac{\|g_k\|^4}{\|g_{k-3}\|^4}\|p_{k-3}\|^2.$$

Now iterating $k$ times I get

$$\|p_k\|^2 \le c_3\|g_k\|^4\left\{\frac{1}{\|g_k\|^2} + \frac{1}{\|g_{k-1}\|^2} + \frac{1}{\|g_{k-2}\|^2} + \cdots + \frac{1}{\|g_1\|^2}\right\} + \frac{\|g_k\|^4}{\|g_0\|^4}\|p_0\|^2.$$

Using the fact that $p_0 = -g_0$ and that $c_3 \geq 1$ I can finally write

$$\|p_k\|^2 \leq c_3 \|g_k\|^4 \sum_{j=0}^{k} \frac{1}{\|g_j\|^2}. \tag{5.12}$$

By the conditions assumed for $f$ ($\mathcal{L}$ bounded and $f$ Lipschitz–$C^1$ on an open set $\mathcal{N}$ containing $\mathcal{L}$) I have that $g_k$ is bounded above $\|g_j\| \leq \bar{\gamma}$.

I also assumed that $\|g_j\| \geq \gamma > 0$.

Now let's use these bounds on $\|g_j\|$ and $\|g_k\|$ in (5.12).

I find that $\|p_k\|^2 \leq c_3\bar{\gamma}^4(k+1)/\gamma^2$.

So

$$\sum_{k=0}^{\infty} \frac{1}{\|p_k\|^2} \geq \frac{\gamma^2}{c_3\bar{\gamma}^4} \sum_{k=0}^{\infty} \frac{1}{k+1}$$

which diverges.

But with $\|g_k\| \geq \gamma$ I can simplify our modified Zoutendijk condition (5.10) to:

$$\sum_{k=0}^{\infty} \frac{1}{\|p_k\|^2} < \infty$$

which gives a contradiction. ■

So $\liminf_{k\to\infty} \|g_k\| = 0$, i.e. the Fletcher-Reeves Method has (more or less) the global convergence property.

## Fletcher-Reeves Method — Numerical Behaviour

Simplifying (5.9) above, I can write

$$\chi_1 \frac{\|g_k\|}{\|p_k\|} \leq \cos(\theta_k) \leq \chi_2 \frac{\|g_k\|}{\|p_k\|}, \tag{5.13}$$

where

$$\cos(\theta_k) = \frac{-g_k^T p_k}{\|g_k\| \|p_k\|}. \tag{5.14}$$

- A "bad" choice of $p_k$, (large in norm compared to $g_k$)

  $$\frac{\|g_k\|}{\|p_k\|} << 1,$$

  leads to $\cos(\theta_k) \approx 0$— as $\chi_1, \chi_2 \approx 1$.

- So I only get a very small change in $x_k$ as reductions in $f$ will be negligible along a direction nearly tangent to a surface $f(x) =$ constant.

- It follows that $\|g_k\| \approx \|g_{k+1}\|$ and so $\beta^{FR}_{k+1} \approx 1$.
- I am assuming that $\frac{\|g_k\|}{\|p_k\|} << 1$ and so both $g_{k+1}$ and $g_k$ are much smaller than $p_k$.
- Using the update rule (9) I have

$$\|p_{k+1} - p_k\| \approx \|p_{k+1} - \beta^{FR}_{k+1} p_k\| \approx \|g_{k+1}\| << \|p_k\| \quad (5.15)$$

  and so $p_{k+1} \approx p_k$.
- So the FR method "sticks" if ever it generates a bad search direction.
- This behaviour means that, despite its good descent direction and convergence properties, the Fletcher-Reeves method is rarely used in practice.

### 5.2.2 Polak-Ribière Method

There are many variants on the older Fletcher-Reeves Method that differ in the choice of the parameter $\beta_k$ . They all reduce to the FR method in the case of quadratic objective functions with an exact line search. The most often used of these — due to Polak and Ribière — defines $\beta_k$ by

$$\beta^{PR}_{k+1} = \frac{g^T_{k+1}(g_{k+1} - g_k)}{\|g_k\|^2} \tag{5.16}$$

I refer to the algorithm where (5.16) replaces line (8) of **FR–CGM** as **PR–CGM**.

**Polak-Ribière Method — Numerical Behaviour** In practice, the Polak-Ribière method is found to be more robust than the Fletcher-Reeves method. It is easy to see why. Following the argument in Section 5.2 above, $\cos(\theta_k) \approx 0$ leads to $g_k \approx g_{k+1}$ but now it follows that $\beta^{PR}_{k+1} \approx 0$ so that the new search direction $p_{k+1} \approx -g_{k+1}$. This "automatic restart" property means that them method will not "stick" if a poor choice of $p_k$ is made. This good behaviour has made **PR–CGM** the most widely used nonlinear conjugate gradient method for several decades.

## Polak-Ribière Method – Global Convergence

Unfortunately the strong Wolfe conditions do **not** guarantee that $p_k$ is a descent direction and so convergence is not guaranteed. However the simple change

$$\beta^+_{k+1} = \max\{\beta^{PR}_{k+1}, 0\} \tag{5.17}$$

is all that is necessary to ensure that $p_{k+1}$ is globally convergent — provided that I ensure that each $p_k$ satisfy a "sufficient descent condition" $g_k^T p_k \le -c_3 \|g_k\|^2$, in addition to the strong Wolfe conditions. Note that this ensures that each $p_k$ is a descent direction. In practice this has not been a difficulty but has required extra checking and (where necessary) extra iterations in the line search code.

### 5.2.3 Advanced Methods

In Appendix A.18 I discuss some newer variations on nonlinear conjugate gradient method — based on alternative formulas for $\beta$.

The material is technical — overall conclusions:

- FR cgm generates descent directions & is globally convergent.
- FR cgm behaves badly in that it tends to fail if ever $\cos\theta_k \approx 0$ and $g_k \approx g_{k-1}$.
- PR cgm only generates descent directions & is globally convergent when a two-stage version of the Strong Wolfe conditions is applied.
- PR cgm does not suffer from the "sticking" problem — informally; the top line in $\beta_k^{PR} \to 0$ if $g_k \approx g_{k-1}$, this causes the algorithm to effectively "restart" by setting $p_{k+1} \approx -g_{k+1}$.

- A new method (DY-cgm) where

$$\beta_k^{DY} = \frac{\|g_k\|^2}{p_{k-1}^T y_{k-1}} \quad (y_{k-1} \equiv g_k - g_{k-1}) \tag{5.18}$$

  both generates descent directions & is globally convergent requiring only weak Wolfe conditions.

- But "sticks" like FR-cgm.
- An even newer "hybrid" method where

$$\beta_k^H = \max\{0, \min\{\beta_k^{HS}, \beta_k^{DY}\}\} \tag{5.19}$$

  generates descent directions & is globally convergent requiring only weak Wolfe conditions and does not have the "sticking" problem.

- Newer methods are still being developed.

## 5.3 Exercises

1. Show that, when applied to a quadratic function with exact line searches, the Polak-Ribière Method reduces to the Fletcher-Reeves formula.

# References

[1] Y. H. Dai, Y. Yuan, A Nonlinear Conjugate Gradient Method with a Strong Global Convergence Property, SIAM Journal on Optimization, Volume 10, Number 1 pp. 177-182, 1999.

[2] Y. H. Dai, Y. Yuan, An Efficient Hybrid Conjugate Gradient Method for Unconstrained Optimization, Annals of Operations Research, Vol 103 pp. 33–47, 2001.

# 6 Quasi-Newton Methods

- My final class of line search methods for unconstrained problems are referred to as quasi-Newton methods.
- Like the steepest descent method, they only require the gradient of the objective function $f$ to be supplied at each iteration.
- By measuring the changes in the gradient, they construct an approximation to $f$ good enough to produce super-linear convergence (remember that Newton's method converges quadratically while the steepest descent method and nonlinear conjugate gradient methods only converge linearly).

- As second derivatives are not required, quasi-Newton methods are often more efficient than Newton's method.
- (Conjugate gradient methods have the further advantage that — unlike quasi-Newton methods — they do not require even an estimate of the Hessian to be stored, at the price of slower convergence).

## 6.1 The BFGS Method

In this Section, I will discuss the most popular quasi-Newton method, the BFGS method, together with its precursor & close relative, the DFP algorithm.

Start by forming the familiar quadratic model/approximation:

$$m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T H_k p \tag{6.1}$$

- Here $H_k$ is an $n \times n$ positive definite symmetric matrix that will be updated at each iteration.
- In most books and journals $B$ is used for the current estimate of the Hessian.
- In this Chapter for clarity I will use $H$ for approximations to the **Hessian** and $J$ for approximations to the **Inverse Hessian**.

- The function and gradient values of the model at $p = 0$ match $f_k$ and $g_k$ .
- In other words $m_k(0) = f_k$ and $\nabla_p m_k(p)_{|p=0} = g_k$.
- The minimiser of this model is as usual:

$$p_k = -H_k^{-1} g_k \tag{6.2}$$

  and is used as the search direction.
- The new iterate is

$$x_{k+1} = x_k + \alpha_k p_k \tag{6.3}$$

  again as usual, where the step length $\alpha_k$ is chosen to satisfy the Wolfe conditions.
- Clearly if $H_k$ is the exact Hessian, I have Newton's method — in this Chapter, $H_k$ will be an approximation to the Hessian based on gradient values.

- Instead of computing $H_k$ afresh at each iteration, Davidon used the following clever argument:
- Suppose that I have generated a new iterate $x_{k+1}$ and wish to construct a new quadratic model of the form

$$m_{k+1}(p) = f_{k+1} + g_{k+1}^T p + \frac{1}{2} p^T H_{k+1} p.$$

- How should I keep $H_{k+1}$ consistent with $H_k$?
- It is reasonable to ask that the gradient of $m_{k+1}$ should match the gradient of $f$ at $x_k$ & $x_{k+1}$ .
- Since $\nabla m_{k+1}(0) \equiv g_{k+1}$ , (they match at $x_{k+1}$ ) I need only check that they match at $x_k$ — which means I require that:

$$\nabla m_{k+1}(-\alpha_k p_k) \equiv g_{k+1} - \alpha_k H_{k+1} p_k = \nabla m_k(0) \equiv g_k .$$

- Rearranging, I have

$$H_{k+1} \alpha_k p_k = g_{k+1} - g_k . \tag{6.4}$$

- First define:

$$"\Delta x" \quad s_k = x_{k+1} - x_k \equiv \alpha_k p_k \tag{6.5a}$$

$$"\Delta g" \quad y_k = g_{k+1} - g_k \tag{6.5b}$$

  Then (6.4) gives us the **secant equation**

$$H_{k+1} s_k = y_k. \tag{6.6}$$

- I am taking $H_{k+1}$ to be positive definite so $s_k{}^T H_{k+1} s_k > 0$ and so this equation is possible only if the step $s_k$ and change in gradients $y_k$ satisfy the **curvature condition**

$$s_k^T y_k > 0. \tag{6.7}$$

- When $f$ is strongly convex, this condition is always satisfied (see exercises).
- In general, though, I need to enforce 6.7 by imposing restrictions on the line search procedure for choosing $\alpha_k$.

- In fact, 6.7 is guaranteed to hold if I impose the Wolfe conditions (strong or otherwise) on the line search.
- I check this: using $SW_2$ (4.9b) and remembering (6.5a) that $s_k \equiv \alpha_p p_k$, $SW_2$ reads $g_{k+1}^T s_k \ge c_2 g_k^T s_k$ and so

$$y_k^T s_k \ge (c_2 - 1)\alpha_k g_k^T p_k. \tag{6.8}$$

- Since $c_2 < 1$, $\alpha_k > 0$ and $p_k$ is a descent direction, the term on the right will be positive and the curvature condition holds.
- When the curvature condition is satisfied, the secant equation 6.6 always has a solution $H_{k+1}$.
- The problem is that there are infinitely many solutions for $H_k$ as there are $n(n+1)/2$ degrees of freedom in a symmetric matrix and the secant equation represents only $n$ conditions.
- Requiring that $H_{k+1}$ be positive definite represents $n$ additional conditions but there are still degrees of freedom left.

- To determine $H_{k+1}$ uniquely, I impose the additional condition that; **among all symmetric matrices satisfying the secant equation, $H_{k+1}$ is "closest to" the current matrix $H_k$ .**

- So I need to solve the problem:

$$\min_{H} \|H - H_k\| \tag{6.9a}$$

$$\text{subject to} \quad H = H^T, Hs_k = y_k \tag{6.9b}$$

- I can use any convenient matrix norm — a choice that simplifies the algebra (reduces the pain) is the "weighted Frobenius norm":

$$\|A\|_W \equiv \|W^{\frac{1}{2}} A W^{\frac{1}{2}}\|_F, \tag{6.10}$$

where $\|C\|_F^2 \equiv \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij}^2$ for any square matrix $C$.

- **Any** choice of the weight matrix $W$ will do provided it is positive definite, symmetric and satisfies $Wy_k = s_k$.
- The weight matrix $W$ doesn't play any role in the algorithm to be discussed below — but I need $W$ to **derive** the algorithm.
- So I just need to know that a matrix $W$ can be found s.t. $Wy_k = s_k$ — as if not, the derivation below is built on sand.
- For example, I could take $W = \overline{H_k}^{-1}$, where $\overline{H}_k$is the **average Hessian** defined by

$$\overline{H}_k = \int_0^1 \nabla^2 f(x_k + \tau\alpha_k p_k) d\tau. \tag{6.11}$$

- It follows that

$$y_k = \overline{H}_k \alpha_k p_k = \overline{H}_k s_k \tag{6.12}$$

  using the definitions of $s_k$ & $y_k$ and the application of the Chain Rule.

- **Check** that this choice of $W$ is positive definite and symmetric.

I can now state my update formula for the Hessian estimate $H_k$ as a Theorem:

**Theorem 6.1** **a** *solution of (6.9a, 6.9b) is*

$$\textbf{DFP} \quad H_{k+1} = (I - \gamma_k y_k s_k^T) H_k (I - \gamma_k s_k y_k^T) + \gamma_k y_k y_k^T, \qquad (6.13)$$

*where*

$$\gamma_k = \frac{1}{y_k^T s_k}$$

Before I prove the Theorem a couple of points:

- $H_k$ is my current estimate of the Hessian, usually initially the Identity matrix.
- $H_{k+1}$ is my (I hope) improved estimate of the Hessian, using newly available information, namely the two vectors $s_k$ & $y_k$ .

**Proof:** It is convenient to write

$$K = I - \gamma_k s_k y_k^T \quad \text{so that } K^T = I - \gamma_k y_k s_k^T. \tag{6.14}$$

$$P = \gamma_k y_k y_k^T \quad \text{so that } P^T = P. \tag{6.15}$$

It is easy to check that $Ks_k = 0$, $K^T y_k = 0$ and $Ps_k = y_k$.

- So **any** matrix of the form

$$H = AK + P \tag{6.16}$$

  satisfies the secant equation — where $A$ is an arbitrary $n \times n$ matrix.

- Requiring symmetry tells us that $H$ must take the form

$$H = K^T AK + P \tag{6.17}$$

  — where now $A$ is is an arbitrary $n \times n$ **symmetric** matrix.

- Now comes the hard part: I want to show that $A = H_k$.

- With $\mathsf{H}$ defined as above, define $\mathsf{F}(A) = \|\mathsf{H} - \mathsf{H}_k\|_W^2$.
- I need to minimise $\mathsf{F}(A)$ wrt $A$.
- It is easy to check that $\mathsf{F}(A) = \text{Trace}(W\mathsf{C}W\mathsf{C})$ defining $\mathsf{C}$ by

$$\mathsf{C} \equiv \mathsf{H} - \mathsf{H}_k = \mathsf{K}^T A \mathsf{K} + \mathsf{P} - \mathsf{H}_k. \tag{6.18}$$

- I want to find the matrix $A$ which minimises $\mathsf{F}(A)$.
- I will use the “Einstein summation convention” — repeated indices are implicitly summed over.
- So I write:

$$\mathsf{F}(A) = W_{ij}\mathsf{C}_{jk}W_{kl}\mathsf{C}_{li},$$

and can calculate the partial derivative of $\mathsf{F}$ wrt $A_{\alpha\beta}$ as:

$$\frac{\partial \mathsf{F}(A)}{\partial A_{\alpha\beta}} = W_{ij}W_{kl}\mathsf{C}_{li}\frac{\partial \mathsf{C}_{jk}}{\partial A_{\alpha\beta}} + W_{ij}\mathsf{C}_{jk}W_{kl}\frac{\partial \mathsf{C}_{li}}{\partial A_{\alpha\beta}}.$$

- You should be able to see that

$$\frac{\partial C_{jk}}{\partial A_{\alpha\beta}} = K^T_{j\alpha} K_{\beta k} \text{ and } \frac{\partial C_{li}}{\partial A_{\alpha\beta}} = K^T_{l\alpha} K_{\beta i}.$$

- Putting it all together, the choice of matrix $A$ which minimises $F(A)$ must satisfy $2\left(KWCWK^T\right)_{\beta\alpha} = 0$ for all coices of the subscripts $\alpha$ and $\beta$ so I must have

$$KWCWK^T = 0.$$

- Substituting for $C$, $K$, $K^T$ and $P$ from 6.18, 6.14, 6.15, I have

$$\begin{aligned} 0 &= KW\left(K^TAK + P - H_k\right)WK^T \\ &= KWK^TAKWK^T + KWPWK^T - KWH_kWK^T. \end{aligned}$$

- But (as $Wy_k = s_k$), $KWP = KW\gamma_k y_k y_k^T = \gamma_k K s_k y_k^T = 0$, so we finally have (noting that $KW = KWK^T$)

$$(KW)(A - H_k)(KW)^T = 0 \tag{6.19}$$

which is satisfied by $A = H_k$ as claimed. ■

- The formula Eq. 6.13 is called the **DFP** formula due to Davidon, Fletcher & Powell.
- Other choices of $A$ also satisfy (6.19).
- Check that for any choice of the real parameters $\alpha$ & $\beta$, a matrix of the form $A = H_k + \alpha y s^T + \beta y y^T$ also satisfies the equation.
- I say that there is a "family" of formulas corresponding to different choices of $\alpha$ and $\beta$. See Ex. 8.
- I will not discuss this approach further here.

## 6.2 Inverting the Hessian approximation

It would be very useful if I could calculate an estimate of the the **inverse** Hessian $\nabla^2 f$ — say $J_k \equiv H_k^{-1}$. This would allow us to calculate $p_k = -J_k g_k$ instead of solving $H_k p_k = -g_k$ for the search direction $p_k$ — giving a speedup in the algorithm.

But how to transform Eq. 6.13 into an update formula for $J_{k+1}$ in terms of $J_k$ ?

I need a formula that gives the inverse of $H_{k+1}$ in terms of the inverse of $H_{k+1}$.

The Sherman-Morrison-Woodbury formula is what I need.

It states that if a square non-singular matrix $A$ is updated by

$$\hat{A} = A + RST^T$$

where $R, T$ are $n \times p$ matrices for $1 \leq p < n$ and $S$ is $p \times p$ then

$$\hat{A}^{-1} = A^{-1} - A^{-1} R U^{-1} T^T A^{-1}, \tag{6.20}$$

where $U = S^{-1} + T^T A^{-1} R$.

See Ex. 3 for hints on proving this result.

Using the SMW formula, I can derive the following equation for the update of the inverse Hessian approximation, $H_k$ that corresponds to the DFP update of $B_k$ in Eq. 6.13;

$$\textbf{DFP} - \textbf{Inverse} \quad J_{k+1} = J_k - \frac{J_k y_k y_k^T J_k}{y_k^T J_k y_k} + \frac{s_k s_k^T}{y_k^T s_k}. \tag{6.21}$$

(See Exercise 4 for some of the details.)

This is a rank-2 update as the two terms added to $J_k$ are both rank-1.

The DFP method has been superseded by the BFGS (Broyden, Fletcher, Goldfarb & Shanno) method. It can be derived by making a small change in the derivation that led to Eq. 6.13. Instead of imposing conditions on the Hessian approximations $H_k$ , I impose corresponding conditions on their inverses $J_k$ . The updated approximation $J_{k+1}$must be symmetric and positive definite. It must satisfy the secant equation Eq. 6.6, now written as

$$J_{k+1} y_k = s_k. \tag{6.22}$$

and also the "closeness" condition

$$\min_J \|J - J_k\| \tag{6.23a}$$

$$\text{subject to} \quad J = J^T, J y_k = s_k. \tag{6.23b}$$

The matrix norm is again the weighted Frobenius norm, where the weight matrix is now any matrix satisfying $W s_k = y_k$.

(You can take $W$ to be the "average" Hessian $\overline{H}_k$ defined in Eq. 6.11 above — though any matrix satisfying $Ws_k = y_k$ will do.)

Using the same reasoning as above, a solution to 6.23a is given by

$$\textbf{BFGS} \quad J_{k+1} = (I - \gamma_k s_k y_k^T) J_k (I - \gamma_k y_k s_k^T) + \gamma_k s_k s_k^T. \qquad (6.24)$$

Note the symmetry between this equation and Eq. 6.13 — one transforms into the other by simply interchanging $s_k$ and $y_k$ — of course $\gamma_k = \frac{1}{{s_k}^T y_k}$ is invariant under this transformation.

I often take $J_0$ to be just the identity matrix — possibly scaled.

(Again, as the DFP update formula on Slide 199 above and in Ex. 8, there is a two-parameter family of formulas that solve this problem. See Ex. 9.)

**Algorithm 6.1 (BFGS)**

```
begin
    Given $x_0$, tolerance $\varepsilon > 0$ and starting $J_0$
    $k \leftarrow 0$;
    while $\|g_k\| > \varepsilon$ do
            $p_k \leftarrow -J_k g_k$
            $x_{k+1} \leftarrow x_k + \alpha_k p_k$(where $\alpha_k$ satisfies the Wolfe conditions )
            Define $s_k = x_{k+1} - x_k$ and $y_k = g_{k+1} - g_k$
            Compute $J_{k+1}$using Eq. 6.24
            $k \leftarrow k + 1$;
    end (while)
end
```

Each iteration takes $O(n^2)$ arithmetic operations (plus the cost of function & gradient operations). (Solving linear systems requires $O(n^3)$ arithmetic operations.) The rate of convergence is superlinear. Newton's method has quadratic convergence but needs solution of a linear system and of course calculation of the Hessian at each iteration.

One very nice property of the BFGS method is that if $J_k$ is positive definite then $J_{k+1}$is also, so a suitable choice for $J_0$ ensures that all subsequent $p_k$ 's will be descent directions. (See Ex. 6.)

### 6.2.1 Robustness of BFGS

Finally, the superiority of the BFGS algorithm over the DFP method is due to the its robustness wrt the scalar $\gamma_k$ becoming large. Suppose that at some iteration $k$, the inner product $s_k^T y_k$ is very small, then $\gamma_k = 1/s_k^T y_k$ becomes very large and so $J_{k+1}$will be also. Can the algorithm recover from this? It can be shown that BFGS will tend to correct itself after a few iterations while DFP will be slow to recover. See App. A.22 for details.

In fact robustness (in the Appendix) and convergence (in the next Section) are proved for the confusingly named "inverse" BFGS method **BFGS-Inverse** (6.25) that updates $H_k \approx \nabla^2 f$ — as the algebra is easier. Of course, the good properties of **BFGS-Inverse** hold for **BFGS** as well because $J_k \equiv H_k^{-1}$ at each iteration.

Similarly the bad behaviour of DFP (6.13) will also be seen in DFP-inverse (6.21) which updates $J_k \approx (\nabla^2 f)^{-1}$.

## 6.3 Convergence Analysis

The convergence properties of the BFGS method are satisfactory in practice. However, proofs of global convergence are only available under restrictive assumptions — in particular that $f$ is convex. I will just prove a more limited result.

Note that the condition (6.26) in the statement of the Theorem implies that the Hessian is positive definite — this is often an unrealistic assumption.

I will prove convergence for the "inverse" BFGS method that estimates the Hessian $H_k$ rather than the inverse Hessian $J_k$:

$$\textbf{BFGS} - \textbf{Inverse} \quad H_{k+1} = H_k - \frac{H_k s_k s_k^T H_k}{s_k^T H_k s_k} + \gamma_k y_k y_k^T \tag{6.25}$$

It can be derived from the inverse DFP formula (6.21) simply by swapping $s_k$ and $y_k$ — or by applying the SMW formula to the BFGS method.

(Rather confusingly, the "inverse" BFGS formula computes an estimate of the Hessian rather than the inverse Hessian. This potential confusion is the reason why $H$ and $J$ are used in these notes to refer to estimates of the Hessian and inverse Hessian respectively — irrespective of the method used.)

**Theorem 6.2** *For any starting point $x_0$; if the objective function $f$ is $C^2$, and there exist positive constants $m$ and $M$ such that*

$$m\|z\|^2 \leq z^T \nabla^2 f(x) z \leq M\|z\|^2 \tag{6.26}$$

*for all $z \in \mathbb{R}^n$ and all $x \in \Omega = \{x \in \mathbb{R}^n : f(x) \leq f(x_0)\}$, then the sequence $\{x_k\}$ generated by the* **inverse** *BFGS formula (6.25) satisfies*

$$\liminf \|\nabla f(x_k)\| = 0.$$

**Proof:**

First define

$$m_k = \frac{y_k^T s_k}{s_k^T s_k}, \quad M_k = \frac{y_k^T y_k}{y_k^T s_k} \tag{6.27}$$

I have already discussed the "average" Hessian $\overline{H}_k$ defined in Eq. 6.11 above. I saw that $\overline{H}_k s_k = y_k$. so

$$m_k \equiv \frac{y_k^T s_k}{s_k^T s_k} = \frac{s_k^T \overline{H}_k s_k}{s_k^T s_k} \geq m \tag{6.28}$$

by (6.26). Similarly (defining $z_k = \overline{H}_k^{1/2} s_k$),

$$M_k \equiv \frac{y_k^T y_k}{y_k^T s_k} = \frac{z_k^T \overline{H}_k z_k}{z_k^T z_k} \leq M \tag{6.29}$$

again using (6.26).

The following results are easily checked (see Exercises 7 and 12):

$$\operatorname{trace} H_{k+1} = \operatorname{trace} H_k - \frac{\|H_k s_k\|^2}{s_k^T H_k s_k} + \frac{\|y_k\|^2}{y_K^T s_k} \tag{6.30}$$

$$\det H_{k+1} = \det H_k \left( \frac{y_k^T s_k}{s_k^T H_k s_k} \right). \tag{6.31}$$

Now define

$$\cos\theta_k = \frac{s_k^T H_k s_k}{\|s_k\| \|H_k s_k\|},$$ equivalent to the standard definition (4.16)

and $$q_k = \frac{s_k^T H_k s_k}{\|s_k\|^2},$$ a useful intermediate quantity.

Using these definitions and those for $M_k$ and $m_k$ I have

$$\operatorname{trace} H_{k+1} = \operatorname{trace} H_k - \frac{q_k}{\cos^2\theta_k} + M_k$$

$$\det H_{k+1} = \det H_k \left( \frac{m_k}{q_k} \right).$$

Define $\psi(B) = \text{trace}(B) - \ln \det B$. For $B$ positive definite, it is easy to check that $\psi(B) > 0$. Now

$$\begin{aligned}\psi(H_{k+1}) &= \psi(H_k) - \frac{q_k}{\cos^2\theta_k} + M_k - \ln m_k + \ln q_k \\ &= \psi(H_k) + (M_k - \ln m_k) + \left[\ln \frac{q_k}{\cos^2\theta_k} - \frac{q_k}{\cos^2\theta_k}\right] \\ &+ \ln\cos^2\theta_k.\end{aligned}$$

As the function $h(t) = \ln t - t$ is negative for all positive $t$ (check) I have by iterating;

$$0 < \psi(H_{k+1}) \le \psi(H_1) + Ck + \sum_{j=1}^{k} \ln\cos^2\theta_j \qquad (6.32)$$

where $C \equiv M - \ln m$ can be taken to be positive without loss of generality.

Finally, by Zoutendijk's Theorem, 4.3, I know that if $\liminf \|g_k\| \not\to 0$ then $\cos\theta_k \to 0$. So (proving the Theorem by contradiction) I have that "$\ln\cos^2\theta_k \to -\infty$" or more usefully that there is some $N > 0$ that for all $j > N$, $\ln\cos^2\theta_j < -2C$, where $C$ is the positive constant defined above (or any positive constant for that matter).

Then

$$\begin{aligned} 0 &< \psi(H_{k+1}) \\ &\le \psi(H_1) + Ck + \sum_{j=1}^{N} \ln\cos^2\theta_j + \sum_{j=N+1}^{k} (-2C) \\ &= \psi(H_1) + \sum_{j=1}^{N} \ln\cos^2\theta_j + 2CN - Ck. \end{aligned}$$

But the RHS in the final equation is negative for large $k$, which contradicts $\psi(H_{k+1}) > 0$. ■

Note that if $f$ is convex, then the above Theorem ensures that $x$ converges to a minimum point $x^*$ of $f$. My final result is even more technical and so is not proved here. It (more or less) establishes that BFGS has performance midway between Steepest descent & Newton's Method.

**Theorem 6.3** *Suppose that $f$ is $C^2$ and that the sequence $\{x_k\}$ generated by the BFGS algorithm converges to the minimiser $x^*$ of $f$. Suppose also that the condition*

$$\|\nabla^2 f(x) - \nabla^2 f(x^*)\| \leq L\|x - x^*\|$$

*holds for all $x$ sufficiently close to $x^*$ for some positive constant $L$. Finally, suppose that $x_k$ converges to $x^*$ linearly. Then $x_k$ converges to $x^*$ superlinearly.*

## 6.4 L-BFGS Method

An important variant of the BFGS method is the L-BFGS method — based on the simple idea that the most recent $s$ and $y$–values are likely to be the most important in the sense that the current (inverse) Hessian approximation is more likely to be influenced by $s$ and $y$–values computed at recent iterations. As only the most recent $m$ values of $s$ and $y$ are stored I can also expect huge reductions in storage costs (and therefore increases in speed) compared with the standard BFGS method.

See App. A.26 for details.

## 6.5 Exercises

1. Show that if $f$ is strongly convex, then the curvature condition 6.7 holds for any vectors $x_k$ & $x_{k+1}$ .
2. Show that the second strong Wolfe condition 4.10b implies the curvature condition 6.7.
3. Prove the Sherman-Morrison-Woodbury formula 6.20. See Appendix A.20 for a proof.
4. Using the SMW formula derive (6.21) — the inverse DFP update formula. See Appendix A.21 for a proof.
5. Check that Eq. 6.13 and Eq. 6.21 are inverses of each other.
6. Show that the update rule 6.24 implies that if $H_k$ is positive definite, then so is $H_{k+1}$.
7. Prove (6.30).

8. Check the claim on Slide 199that any matrix $A = H_k + \alpha y s^T + \beta y y^T$ also satisfies (6.19).

9. Do the corresponding analysis for the BFGS update — find the two-parameter family of update formulas that solve the problem.

10. Prove that $\det(I + xy^T) = 1 + y^T x$ for any vectors $x$ and $y$. Hint: for any $x \neq 1$, I can find vectors $w_1, \dots, w_{n-1}$ such that the matrix $Q$ whose columns are $x$ and the vectors $w_1, \dots, w_{n-1}$ has an inverse and satisfies $x = Qe_1$. (The vector $e_1 = (1, 0, \dots, 0)^T$.) If I define $y^T Q = (z_1, \dots, z_n)$, then $z_1 = y^T Q e_1 = y^T Q(Q^{-1}x) = y^T x$ so

$$\det(I + xy^T) = \det\left(Q^{-1}(I + xy^T)Q\right) = \det(I + e_1 y^T Q).$$

11. Extend the above result to prove that

$$\det(I + xy^T + uv^T) = (1 + y^T x)(1 + v^T u) - (x^T v)(y^T u).$$

12. Prove (6.31) — a simple application of the previous Exercise.

# 7 Trust Region Methods

I showed previously that line search methods & trust region methods both generate steps based on a quadratic model of the objective function but that they use the model in different ways.

- In Ch. 4 I showed that line search methods use the quadratic model to generate a search direction & then focus on finding a suitable step length $\alpha$ along this direction.
- On the other hand, trust region methods define a region round the current point within which they trust the model to be an adequate representation of the objective function and then choose the step to be the (approximate) minimiser of the model in this trust region. (A more conservative approach.)

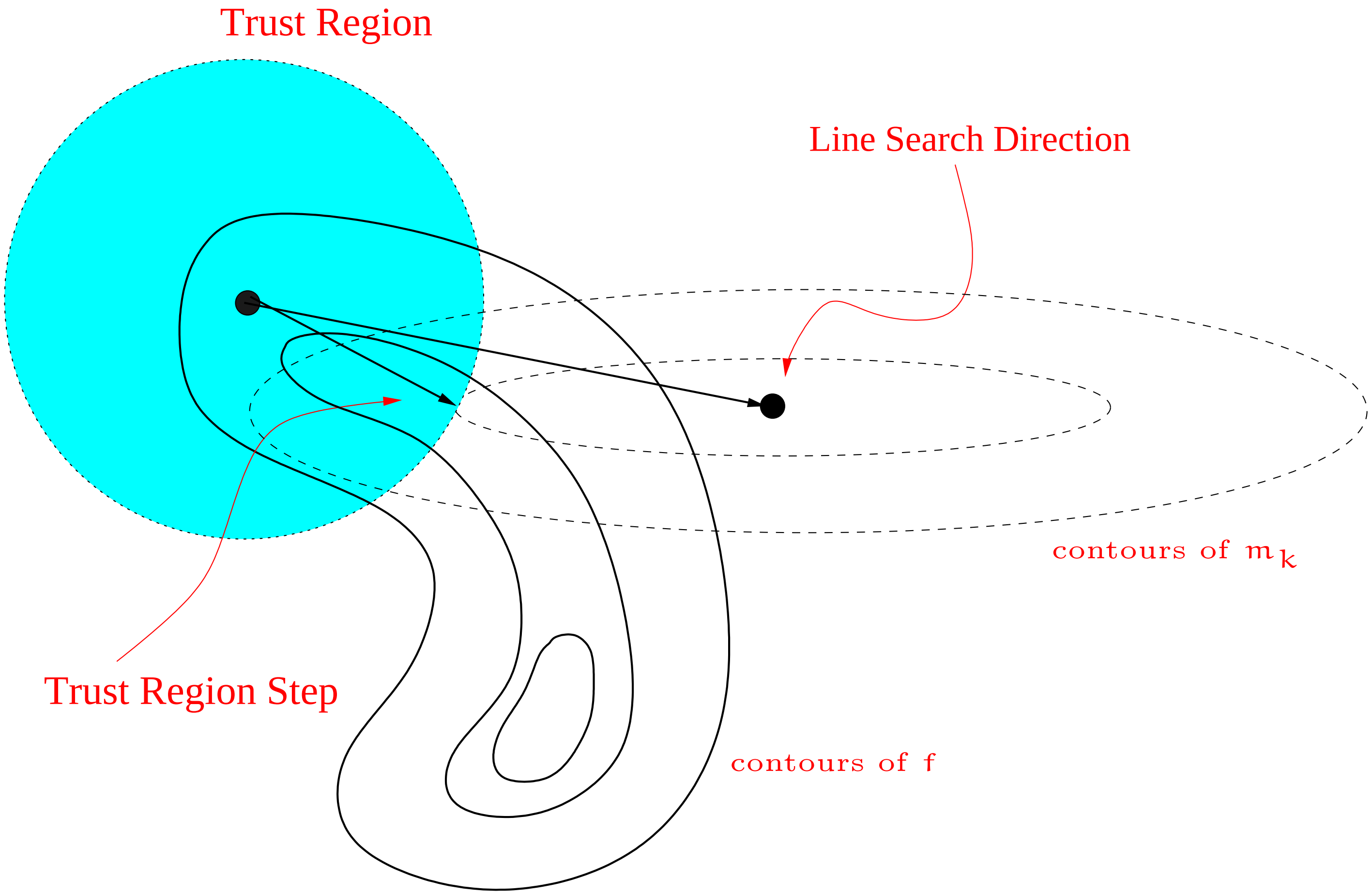


Figure 10: Trust-region vs. line-search methods

Obviously, the size of the trust region is crucial. Too small a region means a missed opportunity to take a large step while too large a region may mean a minimiser far from the minimiser of the objective function in the region.

Fig. 10 shows the trust region approach on a function $f$ (based on Rosenbrock's function — a "banana-shaped valley"). For a more concrete example see Example 3.5 and in particular Figure 4.

I will assume that the first two terms of the quadratic model function $m_k$ at each iterate $x_k$ are identical to the first two terms of the Taylor Series expansion of $f$ around $x_k$ . I have

$$m_k(p) = f_k + {g_k}^T p + \frac{1}{2} p^T B_k p, \tag{7.1}$$

where $B_k$ is some (as yet unspecified) symmetric matrix.

Comparing with the second-order Taylor Series expansion (3.8), the difference between $m_k(p)$ and $f(x_k + p)$ is $O(\|p\|^2)$.

The natural choice for $B_k$ is the exact Hessian $\nabla^2 f_k$ . For the moment I make no assumptions about $B_k$ except

- $B_k$ symmetric
- $\|B_k\| \leq M$ for all $k$ — $M$ some positive constant.
  - which means that the largest eigenvalue of $B_k$ is bounded by $M$ for all $k$ if I use the (Euclidean) 2-norm.

At each iteration, I seek a solution of the subproblem,

$$\min_{p \in \mathbb{R}^n} m_k(p) = f_k + {g_k}^T p + \frac{1}{2} p^T B_k p \quad \text{s.t.} \quad \|p\| \leq \Delta_k, \tag{7.2}$$

where $\Delta_k$ is the trust region radius.

- For the moment, I will use the 2-norm; so the trust region is a ball centred at $x_k$ , radius $\Delta_k$.
- If $B_k$ is positive definite and the Newton direction ${p_k}^B = -{B_k}^{-1} g_k$ has norm $\leq \Delta_k$, the solution of Eq. 7.2 is just the **unconstrained** minimum to the subproblem, (7.2).
- In this case I call ${p_k}^B$ the **full step**.
- In other cases it is not so easy to find an approximate solution to (7.2).

## 7.1 Outline of the Algorithm

- The first decision to make is the strategy for choosing the trust region radius $\Delta_k$ at each iteration.
- I base the strategy on the agreement between the model function $m_k$ and the objective function $f$ at previous iterations.
- Given a step $p_k$ , I define the ratio

$$r_k = \frac{f(x_k) - f(x_k + p_k)}{m_k(0) - m_k(p_k)} = \frac{\textbf{actual reduction}}{\textbf{predicted reduction}}. \tag{7.3}$$

- Note that since the step $p_k$ is found by minimising the model $m_k$ over a region that includes the step $p = 0$, the predicted reduction is always non-negative.

- So, if $r_k < 0$, the new objective value $f(x_k + p_k)$ is greater than the current value $f(x_k)$, so the step must be rejected.
- On the other hand, if $r_k \approx 1$ then the quadratic model is a good approximation to $f$ — so take the step to a new point and expand the trust region radius about the new point, confident that I can **trust** the quadratic model there too.
- If $r_k$ is positive but much less than 1, I take the step but leave the radius unchanged.
- Finally, if $r_k$ is close to zero or negative, I shrink the trust region radius, “stay where I am” and calculate a new step.

The following algorithm describes the process.

**Algorithm 7.1 (Trust Region)**

```
begin
  Choose $\bar{\Delta} > 0$, $\Delta_0 \in (0, \bar{\Delta})$, $\eta \in [0, \frac{1}{4})$ and $k_{max}$
  $k \leftarrow 0$
  while $k < k_{max}$ do
      Find $p_k$ by (approximately) solving 7.2.
      Evaluate $r_k$ from 7.3.
      if $r_k < \frac{1}{4}$
        then $\Delta_{k+1} \leftarrow \frac{1}{4}\|p_k\|$
        else
            if $r_k > \frac{3}{4}$ & $\|p_k\| = \Delta_k$
              then $\Delta_{k+1} \leftarrow \min(2\Delta_k, \bar{\Delta})$
              else
                  $\Delta_{k+1} \leftarrow \Delta_k$
            fi
      fi
      if $r_k > \eta$ then $x_{k+1} \leftarrow x_k + p_k$
                 else $x_{k+1} \leftarrow x_k$
      fi
      $k \leftarrow k + 1$
  end (while)
end
```

To turn this into a practical algorithm, I need to focus on solving the quadratic subproblem (7.2).

## 7.2 The Cauchy Point and Variants

In this Section I first describe how to find the Cauchy point — the most easily calculated approximate solution to the quadratic subproblem (7.2) — and then some algorithms that improve significantly on it.

### 7.2.1 Cauchy Point

I saw in the previous Chapter that a line search doesn't need to take the optimal step for the method to be globally convergent. In the same way, for a trust region method, it is enough for global convergence purposes to find an approximate solution $p_k$ that lies in the trust region and gives a **sufficient reduction** in the value of the model function.

This sufficient reduction can be expressed using as a benchmark the **Cauchy point** which I refer to as $p_k^c$ and define using the following simple steps:

1. Find the vector $p_k^s$ which minimises a **linear** version of $m_k$ , i.e.

$$p_k^s = \arg\min_{p \in \mathbb{R}^n} f_k + g_k{}^T p, \quad \text{such that } \|p\| \leq \Delta_k. \tag{7.4}$$

   Clearly $p_k^s$ is not the "right answer" but a poor approximation to it, so:

2. Calculate the scalar $\tau_k > 0$ that minimises the full quadratic $m_k(\tau p_k^s)$, subject to satisfying the trust region bound, i.e.

$$\tau_k = \arg\min_{\tau > 0} m_k(\tau p_k^s), \quad \text{such that } \|\tau p_k^s\| \leq \Delta_k. \tag{7.5}$$

3. Set $p_k^c = \tau_k p_k^s$.

I can write a closed-form definition of the Cauchy point. The solution to (7.4) is just

$$p_k^s = -\frac{\Delta_k}{\|g_k\|} g_k \,, \tag{7.6}$$

To calculate $\tau_k$ explicitly, consider the cases ${g_k}^T B_k g_k \le 0$ and ${g_k}^T B_k g_k > 0$ separately.

- In the first case, the function $m_k(\tau p_k^s)$ decreases monotonically with positive $\tau$ provided $g_k \neq 0$. (Can you explain why? See Ex. 4.) So $\tau_k$ is just the largest value that satisfies the trust region bound, namely $\tau_k = 1$.
- In the second case, $m_k(\tau p_k^s)$ is a convex quadratic in $\tau$, so $\tau_k$ is either the unconstrained minimiser of $m_k(\tau p_k^s)$, $\tau_k = \dfrac{1}{\Delta_k} \dfrac{\|g_k\|^3}{{g_k}^T B_k g_k}$ ; or the boundary value 1, whichever is smaller.

In summary,

$$p_k^c = -\tau_k \frac{\Delta_k}{\|g_k\|} g_k\,, \tag{7.7}$$

where

$$\tau_k = \begin{cases} 1 & \text{if } g_k{}^T B_k g_k \leq 0\,; \\ \min\left(1, \|g_k\|^3/(\Delta_k g_k{}^T B_k g_k)\right) & \text{otherwise.} \end{cases} \tag{7.8}$$

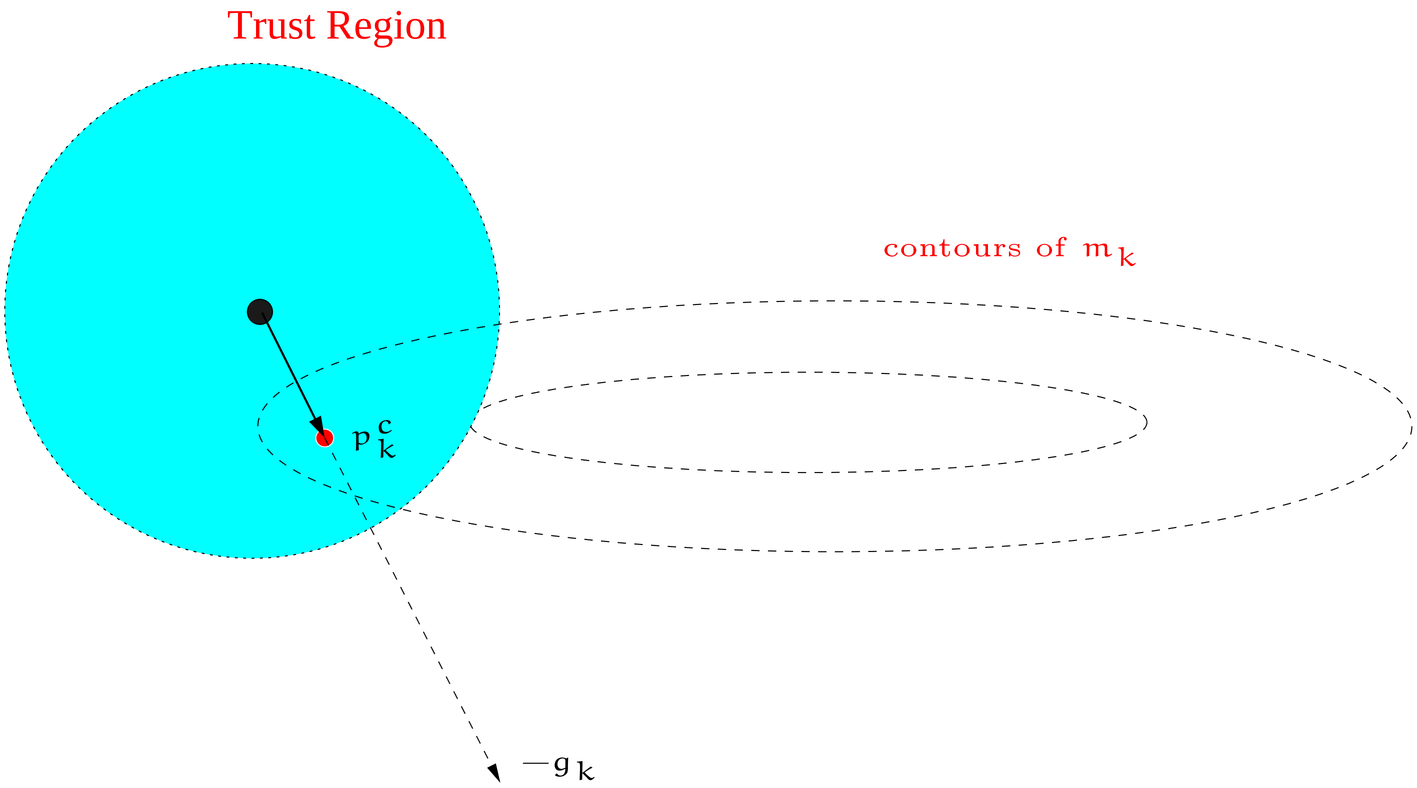


Figure 11: The Cauchy point

- The Cauchy point is quick to calculate — no linear systems have to be solved — and is crucial in deciding whether an approximate solution to the trust region subproblem is acceptable.
- I will show later (Section 7.3) that a trust region method is globally convergent if its steps $p_k$ attain a sufficient reduction in $m_k$ , i.e. they give a reduction in $m_k$ that is at least some fixed multiple of the decrease attained by the Cauchy step at each iteration.
- So the Cauchy point algorithm provides a benchmark against which other methods can be evaluated.

### 7.2.2 Improving on the Cauchy Point

Since the Cauchy point $p_k^c$ provides sufficient reduction in the model function $m_k$ to give global convergence (Section 7.3) and is so cheap to calculate, why bother looking for a better approximate solution to Eq. 7.2?

The answer: the Cauchy point is just the steepest descent method with a particular choice of step length. I saw in Ch. 4 that the steepest descent method converges slowly even for optimal step lengths.

A number of algorithms start by computing the Cauchy point and then improving on it. The improvement strategy is often designed so that the **full step** $p_k^B = -{B_k}^{-1} g_k$ is chosen whenever $B_k$ is positive definite and $\|p_k^B\| \le \Delta_k$. Whenever $B_k$ is the exact Hessian or a quasi-Newton approximation, this strategy will give superlinear performance.

I now consider two methods for approximating the solution to Eq. 7.2 that have these features. In what follows I will be examining the workings of a single iteration so I will drop the "k" subscript to improve readability. In this new notation, the trust region sub-problem is:

$$\min_{p \in \mathbb{R}^n} m(p) \equiv f + g^T p + \frac{1}{2} p^T B p, \quad \text{such that } \|p\| \leq \Delta, \tag{7.9}$$

where I write $g = \nabla f$ and both $f$ and $g$ are evaluated at the current point $x_k$.

I use $p^*(\Delta)$ for the approximate solution of 7.9, to emphasise the dependence on $\Delta$.

### 7.2.3 The Dogleg Method

I start by examining the effect of the trust region radius $\Delta$ on the solution $p^*(\Delta)$ . Look at the two extreme cases (indicated by the **red** and **blue** circles respectively in Fig. 12):

- **When $\Delta$ is very small**, the restriction $\|p\| \leq \Delta$ ensures that the quadratic term in $m$ has little effect on the solution to 7.9. The true solution in this case is approximately the same as that obtained by minimising the linear function $f + g^T p$ over $\|p\| \leq \Delta$, that is $p^*(\Delta) \approx -\Delta \dfrac{g}{\|g\|}, \quad$ when $\Delta$ is small.
- **If $\Delta$ is large** then, provided that $B$ is positive definite, the

unconstrained minimiser of $m$ is the full step $p^B = -B^{-1}g$. The point is guaranteed to be feasible as **$\Delta$ is large**, so I have $p^*(\Delta) = p^B$, when $\|p^B\| \leq \Delta$.

- - For intermediate values of $\Delta$ , (the circles drawn in black in the diagram), I would expect the **exact** solution $p^*(\Delta)$ to follow a curved trajectory (as a function of $\Delta$) — like in Fig 12.
  - For example, if the relevant trust region radius were $\Delta_2$ in the diagram then the **exact** solution would be $p^*(\Delta_2)$ on the boundary of the trust region somewhere closer to $-g$ than to $-B^{-1}g$.
  - The larger the value of $\Delta$, the closer $p^*(\Delta)$ is to $p^B$.
  - The dotted line in the diagram is the "trajectory" of values of $p^*(\Delta)$ as $\Delta$ increases.

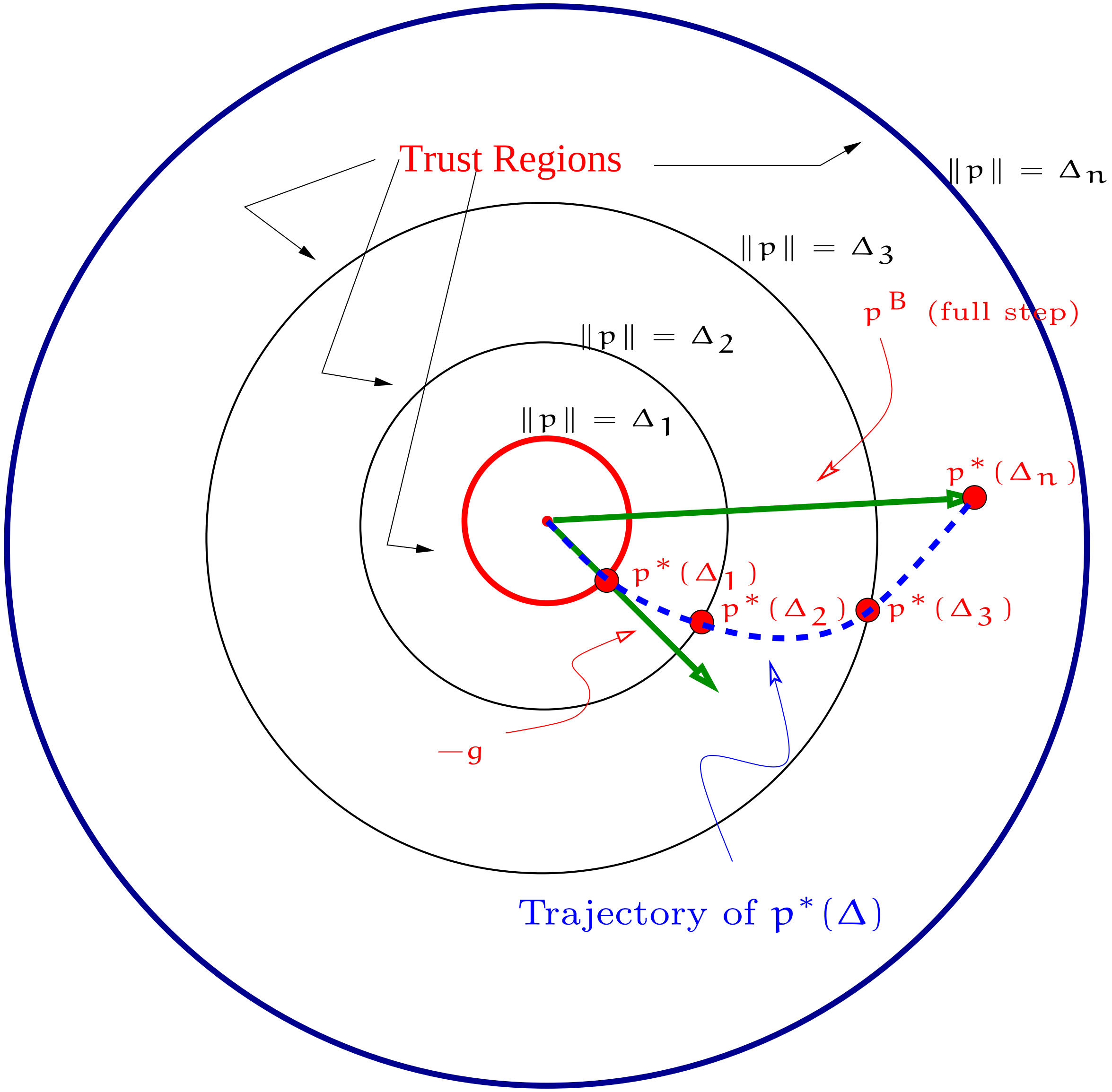


Figure 12: Trajectory of Exact Solutions As $\Delta$ Increases

Of course, I do not calculate the exact solution, but the diagram in Fig 12 motivates the **dogleg method**. This method finds an approximate solution by replacing the (unknown) exact point $p^*(\Delta)$ on the (unknown) curved trajectory with a path consisting of two line segments. The first line segment runs from the starting point to the unconstrained minimiser along the steepest descent direction defined by (substitute $\tau$ from the second expression in Eq. 7.8 into Eq. 7.7 for $p_k^c$)

$$p^U = -\frac{g^T g}{g^T B g} g \tag{7.10}$$

while the second line segment runs from $p^U$ to $p^B$ (see Fig. 13).

- Of course if $\|p^U\| > \Delta$ I need to "shorten" it, equivalent to choosing $\tau = 1$ in the second expression in Eq. 7.8.
- On the other hand if $\|p^U\| < \Delta$ it seems reasonable to continue along the vector $p^B - p^U$ until I reach the boundary $\|p\| = \Delta$.
- This strategy only makes sense if this "dogleg" path is guaranteed to intersect the boundary once only and if $m(p)$ decreases as I move along the path (I will confirm that these conditions hold in Lemma 7.1 below).
- I can define the trajectory as a path $\tilde{p}(\tau)$ parameterised by $\tau$ as follows:

$$\tilde{p}(\tau) = \begin{cases} \tau p^U, & 0 \le \tau \le 1, \\ p^U + (\tau - 1)(p^B - p^U), & 1 \le \tau \le 2. \end{cases} \tag{7.11}$$

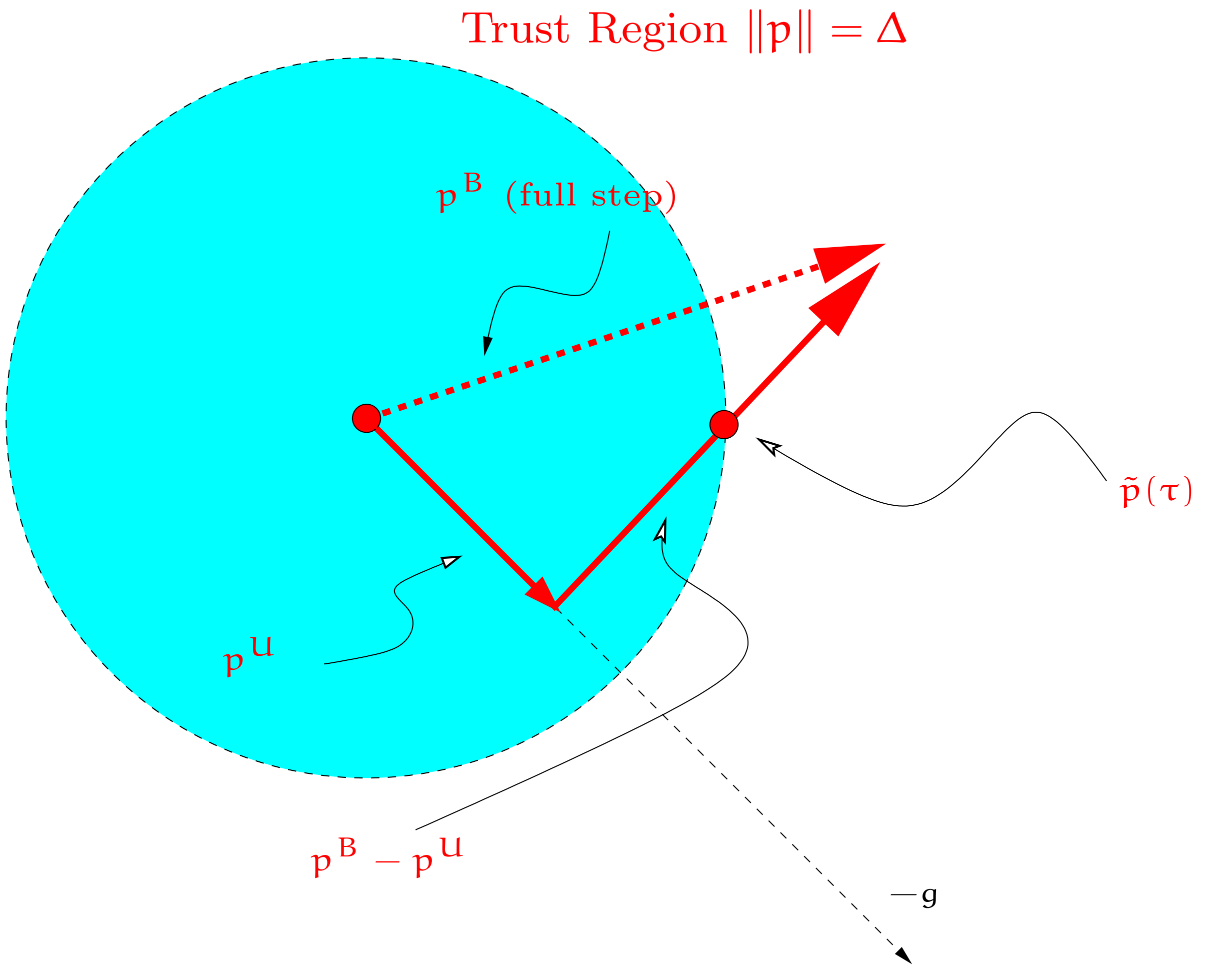


Figure 13: Dogleg approximation

The Dogleg Method chooses $\tau$ and therefore $\tilde{p}(\tau)$ to minimise the model function $m(p)$ along this path, subject to the trust region bound. In fact, no search is necessary as the dogleg path intersects the trust region boundary at most once and the intersection point can be found algebraically as the following Lemma explains:

**Lemma 7.1** *Let* $B$ *be positive definite. Then*

*(i)* $\|\tilde{p}(\tau)\|$ *is an increasing function of* $\tau$ *and*

*(ii)* $m(\tilde{p}(\tau))$ *is a decreasing function of* $\tau$*.*

**Proof:**

**[Case A]** $\tau \in [0, 1]$ Result (i) is obvious. In this case

$$m(\tilde{p}(\tau)) = m(\tau p^U) = f + \tau g^T p^U + \frac{1}{2}\tau^2 p^{UT} B p^U.$$

Substituting for $p^U$, the result (ii) follows.

**[Case B]** $\tau \in [1, 2]$ For result (i), define $h(\alpha), \quad 0 \leq \alpha \leq 1$ by

$$\begin{aligned} h(\alpha) &= \frac{1}{2}\|\tilde{p}(1+\alpha)\|^2 \\ &= \frac{1}{2}\| p^U + \alpha(p^B - p^U)\|^2 \\ &= \frac{1}{2}\| p^U\|^2 + \alpha p^{UT}(p^B - p^U) + \frac{1}{2}\alpha^2\|p^B - p^U\|^2. \end{aligned}$$

R.T.P. that $h'(\alpha) > 0$ for $\alpha \in (0, 1)$. Now,

$$\begin{aligned} h'(\alpha) &= -p^{U^T}(p^U - p^B) + \alpha\|p^B - p^U\|^2 \\ &\geq -p^{U^T}(p^U - p^B) \\ &= \frac{g^Tg}{g^TBg} g^T \left( -\frac{g^Tg}{g^TBg} g + B^{-1}g \right) \\ &= g^Tg \frac{g^TB^{-1}g}{g^TBg} \left[ 1 - \frac{(g^Tg)^2}{(g^TBg)(g^TB^{-1}g)} \right] \\ &\geq 0. \end{aligned}$$

where the final inequality follows from Ex. 5.

For result (ii), define $\hat{h}(\alpha) = m(\tilde{p}(1 + \alpha))$ and show that $\hat{h}'(\alpha) \leq 0$ for $\alpha \in (0, 1)$. Substituting, I find

$$\begin{aligned} \hat{h}'(\alpha) &= (p^B - p^U)^T(g + Bp^U) + \alpha(p^B - p^U)^T B(p^B - p^U) \\ &\leq (p^B - p^U)^T(g + Bp^U + B(p^B - p^U)) \\ &= (p^B - p^U)^T(g + Bp^B) = 0. \quad \blacksquare \end{aligned}$$

It follows from the lemma that the path $\tilde{p}(\tau)$ intersects the trust region boundary $\|p\| = \Delta$ at exactly one point if $\|\tilde{p}(2)\| \equiv \|p^B\| \geq \Delta$ and nowhere otherwise. Since $m$ is decreasing along the path, the chosen value of $p$ will be at $p^B$ if $\|p^B\| \leq \Delta$ , otherwise at the intersection of the dogleg and the trust region boundary. In this case, I find the appropriate value of $\tau$ by solving the quadratic

$$\|p^U + (\tau - 1)(p^B - p^U)\|^2 = \Delta^2$$

for $\tau$.

When the matrix B is not positive definite, I can use a simple variant on the Dogleg Method which I now describe.

### 7.2.4 Two-Dimensional Subspace Minimisation

Still assuming for the moment that $B$ is positive definite, the dogleg method can be improved by widening the search for $p$ to the entire two-dimensional subspace spanned by $p^U$ and $p^B$ ($g$ and $-B^{-1}g$). The subproblem 7.9 is replaced by

$$\min_{p \in \mathbb{R}^n} m(p) \equiv f + g^T p + \frac{1}{2} p^T B p, \text{ such that } \|p\| \le \Delta,\ p \in \text{span}(g, B^{-1}g) \tag{7.12}$$

This problem can be solved algebraically (with a final one-dimensional minimisation step) — see Exercise 6.

- The Cauchy point $p^c$ is feasible (satisfies the trust region constraint) wrt the two-dimensional subspace minimisation problem 7.12; so the optimal solution yields at least as much reduction in $m$ as the Cauchy point , resulting in global convergence of the algorithm.
- The two-dimensional subspace minimisation strategy is clearly an extension of the dogleg method as the entire dogleg path lies in $\text{span}(g, B^{-1}g)$.

A particular advantage of this method is that it can be easily modified to handle the case of indefinite $B$. When $B$ has negative eigenvalues, the two-dimensional subspace in 7.12 changes to

$$\text{span}\ \left(g, (B + \alpha I)^{-1} g\right), \text{ for some } \alpha \in (-\lambda_-, -2\lambda_-). \tag{7.13}$$

where $\lambda_-$ is the most negative eigenvalue of $B$. This choice of $\alpha$ ensures that $B + \alpha I$ is positive definite.

When $\|(B + \alpha I)^{-1} g\| \le \Delta$, I can use a different expression for $p$;

$$p = -(B + \alpha I)^{-1} g + \nu, \tag{7.14}$$

where $\nu$ is chosen to satisfy $\nu^T (B + \alpha I)^{-1} g \le 0$ and of course $\|p\|^2 = \Delta^2$. The first condition ensures that $\nu$ has a positive component along $-(B + \alpha I)^{-1} g$ and so continues to move roughly in the direction of $-(B + \alpha I)^{-1} g$.

When $B$ has zero eigenvalues but no negative eigenvalues, the Cauchy step $p = p^c$ is used as the approximate solution of the problem.

### 7.2.5 "Nearly Exact" Trust Region Methods

- The methods above do not attempt to find the exact solution of 7.9.
- When the dimension $n$ is small it may be worth the extra computational cost of finding a better approximation to the exact solution.
- The Theorem on the next Slide formally lists the properties that a solution $p^*$ will have — in particular that a solution $p^*$ to 7.9 satisfies
$$(B + \lambda I)p^* = -g,$$
for $\lambda \geq 0$ — I design an algorithm that selects an appropriate value for $\lambda$.
- The proof is technical and is left to an Appendix.
- But the result is important.

**Theorem 7.2** *The vector* $p^*$ *is a global solution of the trust region sub-problem 7.9 if and only if there is a scalar* $\lambda \geq 0$ *such that the following conditions are satisfied:*

$$(B + \lambda I)p^* = -g \tag{7.15a}$$

$$\lambda(\Delta - \|p^*\|) = 0 \tag{7.15b}$$

$$(B + \lambda I) \text{ is positive semi-definite.} \tag{7.15c}$$

See Appendix A.15 for a proof. ■

It follows that

- either $\lambda = 0$ satisfies (7.15a) and (7.15c) with $\|p\| \leq \Delta$
- or for $\lambda$ large enough for $(B + \lambda I)$ to be positive definite
  - define $p(\lambda) = -(B + \lambda I)^{-1} g$
  - search for a positive $\lambda$–value that satisfies $\|p(\lambda)\| = \Delta$.

The equation

$$\|p(\lambda)\| = \Delta \tag{7.16}$$

is a one-dimensional root-finding problem in $\lambda$.

To see that a $\lambda$–value with the required properties can be found, use the fact that a symmetric matrix $B$ can be written as

$$B = Q \Lambda Q^T \tag{7.17}$$

where $Q^T Q = Q Q^T = I$ ($Q$ is orthogonal) and $\Lambda$ is a diagonal matrix of the eigenvalues $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$.

The matrix $(B + \lambda I)$ can be written $Q(\Lambda + \lambda I)Q^T$ and so (for $\lambda \neq \lambda_j$)

$$p(\lambda) = -Q(\Lambda + \lambda I)^{-1} Q^T g = -\sum_{j=1}^{n} \frac{q_j^T g}{\lambda + \lambda_j} q_j, \tag{7.18}$$

where the vectors $q_j$ are the columns of $Q$ — the eigenvectors of B. As the eigenvectors of a symmetric matrix are orthonormal, I can write

$$\|p(\lambda)\|^2 = \sum_{j=1}^{n} \frac{(q_j^T g)^2}{(\lambda + \lambda_j)^2}. \tag{7.19}$$

Despite the general nature of the analysis, (7.19) tells us a lot about $\|p(\lambda)\|$.

- If $\lambda > -\lambda_1$ then $\lambda + \lambda_j > 0$ for $j = 1, \ldots, n$ so $\|p(\lambda)\|$ is a continuous non-increasing function of $\lambda$ on the interval $(-\lambda_1, \infty)$.
- $\|p(\lambda)\| \to 0$ as $\lambda \to \infty$
- Also, for each $j$, provided $q_j^T g \neq 0$, $\|p(\lambda)\|$ has a vertical asymptote at $\lambda = -\lambda_j$.

The sketch graph Fig. 14 makes it easy to see that (provided that $q_1^T g \neq 0$) there must be a single root $(-\lambda^*)$ in the interval $(-\lambda_1, \infty)$.

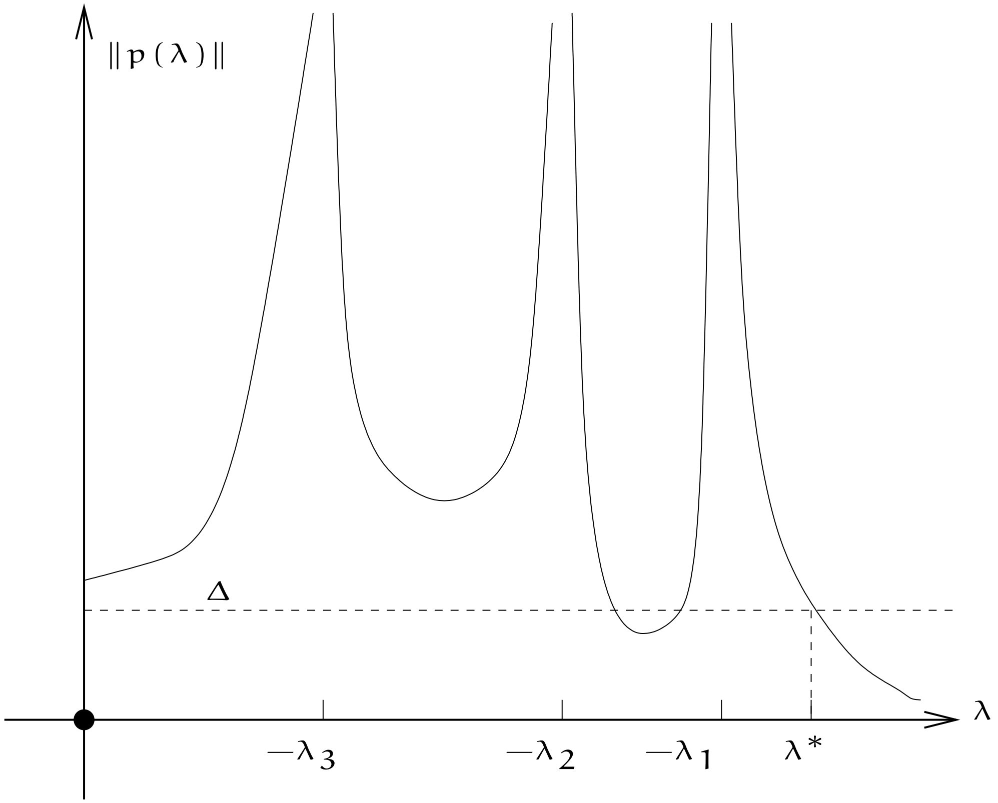


Figure 14: $\|p(\lambda)\|$ as a function of $\lambda$

- If B is positive definite and $\|B^{-1}g\| \leq \Delta$ then $\lambda = 0$ satisfies (7.15a)–(7.15c), so no root-finding is necessary.
- If B is positive definite and $\|B^{-1}g\| > \Delta$ then there is a strictly positive value of $\lambda$ for which $\|p(\lambda)\| = \Delta$ — as in this case $\|p(\lambda)\|$ is monotone decreasing and $\|p(0)\| > \Delta$ — see (7.19).
- If B is indefinite (zero or negative eigenvalues); then, (provided $q_1^T g \neq 0$), Thm. 7.2 guarantees a solution in $(-\lambda_1, \infty)$.

I could simply use Newton's method (first-year calculus) to solve the scalar equation:

$$\phi_1(\lambda) = \|p(\lambda)\| - \Delta = 0 \tag{7.20}$$

for $\lambda > \lambda_1$. However, for $\lambda$ slightly bigger than $-\lambda_1$, $\phi_1$ is changing rapidly. The dominant term (see (7.19)) is given by:

$$\phi_1(\lambda) \approx \frac{C_1}{\lambda + \lambda_1} + C_2, \quad C_1 > 0. \tag{7.21}$$

It is much better to solve:

$$\phi_2(\lambda) = \frac{1}{\Delta} - \frac{1}{\|p(\lambda)\|} = 0 \tag{7.22}$$

as for $\lambda$ slightly bigger than $-\lambda_1$,

$$\phi_2(\lambda) \approx \frac{1}{\Delta} - \frac{\lambda + \lambda_1}{C_3}, \quad C_3 > 0, \tag{7.23}$$

so $\phi_2$ is approximately linear in $\lambda$.

If I apply Newton's root-finding method to $\phi_2$, I get a succession of $\lambda$-values — $\lambda^{(l)}$, say — where

$$\lambda^{(l+1)} = \lambda^{(l)} - \frac{\phi_2(\lambda^{(l)})}{\phi_2'(\lambda^{(l)})}. \tag{7.24}$$

A natural objection is that there is no obvious way to calculate $\phi_2'$. In fact, using matrix factorisation it is possible to by-pass the problem to get the following algorithm (see Ex. 8 below).

**Algorithm 7.2 (Exact Trust Region)**

```
begin
    Given $\lambda_0 > 0$, $\Delta > 0$, $\varepsilon > 0$
    while $l < l_{max} \wedge \text{abs}(\|p_l(\lambda)\| - \Delta) > \varepsilon$ do
        Factor $B + \lambda^{(l)} I = R^T R$
        Solve $R^T R p_l = -g$, $R^T q_l = p_l$
        $\lambda^{(l+1)} := \lambda^{(l)} + \left(\frac{\|p_l\|}{\|q_l\|}\right)^2 \left(\frac{\|p_l(\lambda)\| - \Delta}{\Delta}\right)$
        $l := l + 1$
    end
end
```

Finally; what if $q_1^T g = 0$? In this case, $\|p(\lambda)\|$ does not have a vertical asymptote at $\lambda = -\lambda_1$. There are obviously three possibilities (draw a sketch):

1. $\|p(-\lambda_1)\| = \Delta$. The conditions of Thm. 7.2 are all satisfied by $\lambda^* = -\lambda_1$ and I am done.
2. $\|p(-\lambda_1)\| > \Delta$. In this case Thm. 7.2 guarantees a solution in $(-\lambda_1, \infty)$ as $\|p(\lambda)\|$ is monotone decreasing. Use the NR method as above to find the root.
3. $\|p(-\lambda_1)\| < \Delta$. This is the tricky case (called the Hard Case).

**The Hard Case:** $\|p(-\lambda_1)\| < \Delta$

- As $\|p(\lambda)\|$ is monotone decreasing for $\lambda > -\lambda_2$ I must have $\lambda^* < -\lambda_1$ but this will not satisfy the condition that $B + \lambda^* I$ is positive semidefinite which requires $\lambda^* \geq -\lambda_1$.
- The resolution to this apparent contradiction is to set $\lambda^* = -\lambda_1$.
- I still have a contradiction as $\|p(\lambda^*)\| < \Delta$ but I can fix it by adding a term to $p(\lambda^*)$.
- As the $q_1$-term is missing, I have

$$p^* \equiv p(-\lambda_1) = -\sum_{j:\lambda_j \neq \lambda_1} \frac{q_j^T g}{-\lambda_1 + \lambda_j} q_j. \tag{7.25}$$

- But $\|p(-\lambda_1)\| < \Delta$??

- However, there is an extra degree of freedom in $p$.
- To ensure that $\|p^*\| = \Delta$ we may add **any** multiple of $q_1$ to $p^*$ without affecting the equation $(B + \lambda^* I)p^* = -g$ or (of course) the fact that $B + \lambda^* I$ is positive semidefinite.
- So the final version of $p^*$ is

$$p^* \equiv p(-\lambda_1) = -\sum_{j:\lambda_j \neq \lambda_1} \frac{q_j^T g}{-\lambda_1 + \lambda_j} q_j + \tau q_1. \tag{7.26}$$

- The constant $\tau$ is fixed by requiring that $p^*$ satisfy $\|p^*\| = \Delta$, i.e.

$$\|p^*\|^2 \equiv \sum_{j:\lambda_j \neq \lambda_1} \frac{(q_j^T g)^2}{(-\lambda_1 + \lambda_j)^2} + \tau^2 = \Delta^2 \tag{7.27}$$

## 7.3 Global Convergence

It is possible to prove that trust region algorithm has the vital property of **global convergence**, i.e. that the sequence of gradients $\{g_k\}$ generated by Alg. 7.1 converges to zero (if $\eta > 0$).

The details are technical and are discussed in Appendix A.16.

## 7.4 Exercises

1. Let $f(x) = (1 - x_1)^2 + 10(x_2 - x_1^2)^2$. At $x = (0, -1)$, use Maple or Matlab to draw the contour lines of the quadratic model 7.1, taking B to be the Hessian of f. Draw the family of solutions of 7.2 as the trust region radius varies from $\Delta = 0$ to $\Delta = 2$. Repeat at $x = (0, 0.5)$.

2. Write a Matlab m-file which implements the dogleg method. Choose $B_k$ to be the exact Hessian. Apply it to solve Rosenbrock's problem (previous exercise). Experiment with the update rule for the trust region by changing the constants in Alg. 7.1.

3. Try the more difficult problem

$$\min f(x) = \sum_{i=1}^{n} \left[(1 - x_{2i-1})^2 + 10(x_{2i} - x_{2i-1}^2)^2\right]$$

for $n = 10$.

4. Explain why $m_k(\tau p_k^s)$ decreases monotonically when $g_k{}^T B_k g_k \leq 0$ (see Slide 229).

5. The Cauchy-Schwarz inequality states that for any vectors $u$ & $v$,

$$(u^T v)^2 \leq (u^T u)\,(v^T v)$$

with equality only when $u$ and $v$ are parallel. When $B$ is positive definite, use this inequality to show that

$$\frac{\|p\|^4}{(p^T B p)(p^T B^{-1} p)} \leq 1.$$

6. Derive the solution of the two-dimensional subspace minimisation problem when B is positive definite. Hints:
   - Write $p$ as $p = \alpha g + \beta g_1$ where $g_1 = g + \gamma B^{-1} g$ and $\gamma$ is chosen so that $g^T g_1 = 0$.
   - Show that the equation $\|p\|^2 = \Delta$ is an ellipse in the $\alpha$–$\beta$ plane.
   - Parameterise $\alpha$ & $\beta$ appropriately.
   - Express $m(p)$ as a quadratic in $\alpha$ & $\beta$.
   - Finally, use the parameterised form of $\alpha$ and $\beta$ to express $m(p)$ in terms of a single angle — $t$, say, where $0 \leq t \leq 2\pi$.
   - Plot $m(p)$ as a function of $t$.

- Use any numerical technique you wish to find the optimal value of $\mathtt{t}$.
- The equation to be solved can be reduced to a quartic in either $\sin \mathtt{t}$ or $\cos \mathtt{t}$. A quartic has an exact solution but the formula is so complicated that it is much easier to find a numerical solution.

7. Show that if $B$ is any symmetric matrix then there exists $\lambda \geq 0$ such that $B + \lambda I$ is positive definite.

8. Show the update rule (7.24) (based on the Newton root-finding method) is equivalent to Alg. 7.2 on Slide 257; using the following (just assemble the pieces, easier than it looks):

$$\begin{aligned}\frac{d}{d\lambda}\left(\frac{1}{\|p(\lambda)\|}\right) &= \frac{d}{d\lambda}\left(\|p(\lambda)\|^2\right)^{-1/2} \\ &= -\frac{1}{2}\left(\|p(\lambda)\|^2\right)^{-3/2}\frac{d}{d\lambda}\|p(\lambda)\|^2\end{aligned}$$

$$\frac{d}{d\lambda}\|p(\lambda)\|^2 = -2\sum_{j=1}^{n}\frac{(q_j^T g)^2}{(\lambda+\lambda_j)^3}.$$

Finally,

$$\|q\|^2 = \|(R^T)^{-1}p\|^2 = p^T(B+\lambda I)^{-1}p = \sum_{j=1}^{n}\frac{(q_j^T g)^2}{(\lambda+\lambda_j)^3}.$$

# 8 Introduction to Constrained Optimisation

The remainder of these notes are concerned with minimising functions subject to constraints on the variables. A general formulation for these problems is

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{subject to} \begin{cases} c_i(x) = 0 & i \in \mathcal{E}, \\ c_i(x) \geq 0 & i \in \mathcal{I} \end{cases} \tag{8.1}$$

where $f$ and the functions $c_i$ are all smooth, real-valued functions on a subset of $\mathbb{R}^n$, and $\mathcal{E}$ and $\mathcal{I}$ are two finite sets of indices. As before, I call $f$ the **objective** function, while $c_i, i \in \mathcal{E}$ are the **equality constraints** and $c_i, i \in \mathcal{I}$ are the **inequality constraints** .

I define the **feasible set** $\Omega$ to be the set of points $x$ that satisfy the constraints; that is,

$$\Omega = \{x | c_i(x) = 0, i \in \mathcal{E};\ c_i(x) \geq 0, i \in \mathcal{I}\}, \tag{8.2}$$

so that I can rewrite 8.1 more compactly as

$$\min_{x \in \Omega} f(x). \tag{8.3}$$

- Of course, “hiding” the constraints in this way doesn’t make them go away.
- Or make it easier to solve the proble,

- In Section 8.1 I look at a couple of examples of **constrained problems** and try to infer optimality conditions (analogous to those for unconstrained problems — zero gradient and positive definite/semi-definite Hessian) to characterise the solutions of constrained optimisation problems.
- I'll go through these examples quickly as they are very simple.
- In Section 8.2 I will formally state and prove a set of first-order necessary conditions for optimality.
- In Section 8.3 I will derive second order necessary and sufficient conditions for optimality.

## 8.1 Characterising Optimal Points

- Remember that for the unconstrained optimisation problem of Chapter 3, I characterised solution points $x^*$ in the following way:
  - Necessary conditions: Local minima of unconstrained problems have $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ positive semidefinite.
  - Sufficient conditions: Any point $x^*$ at which $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive definite is a strong local minimiser of $f$.
- In this Section I look at a succession of **constrained problems** and try to infer the corresponding conditions to characterise the solutions of constrained optimisation problems.

### 8.1.1 Examples

- To introduce the basic principles behind the characterisation of solutions of constrained optimisation problems, I work through three simple examples.
- The ideas discussed here will be made rigorous in Sections 8.2 and 8.3.
- I start by noting one item of terminology that I will often need: at a feasible point $x$, the inequality constraint $i \in \mathcal{I}$ is said to be **active** if $c_i(x) = 0$ and **inactive** if the strict inequality $c_i(x) > 0$ is satisfied.

Example **8.1 (Single Equality Constraint)** *My first example is a two-variable problem with a single equality constraint:*

$$\min x_1 + x_2 \quad s.t. \quad x_1^2 + x_2^2 - 2 = 0 \tag{8.4}$$

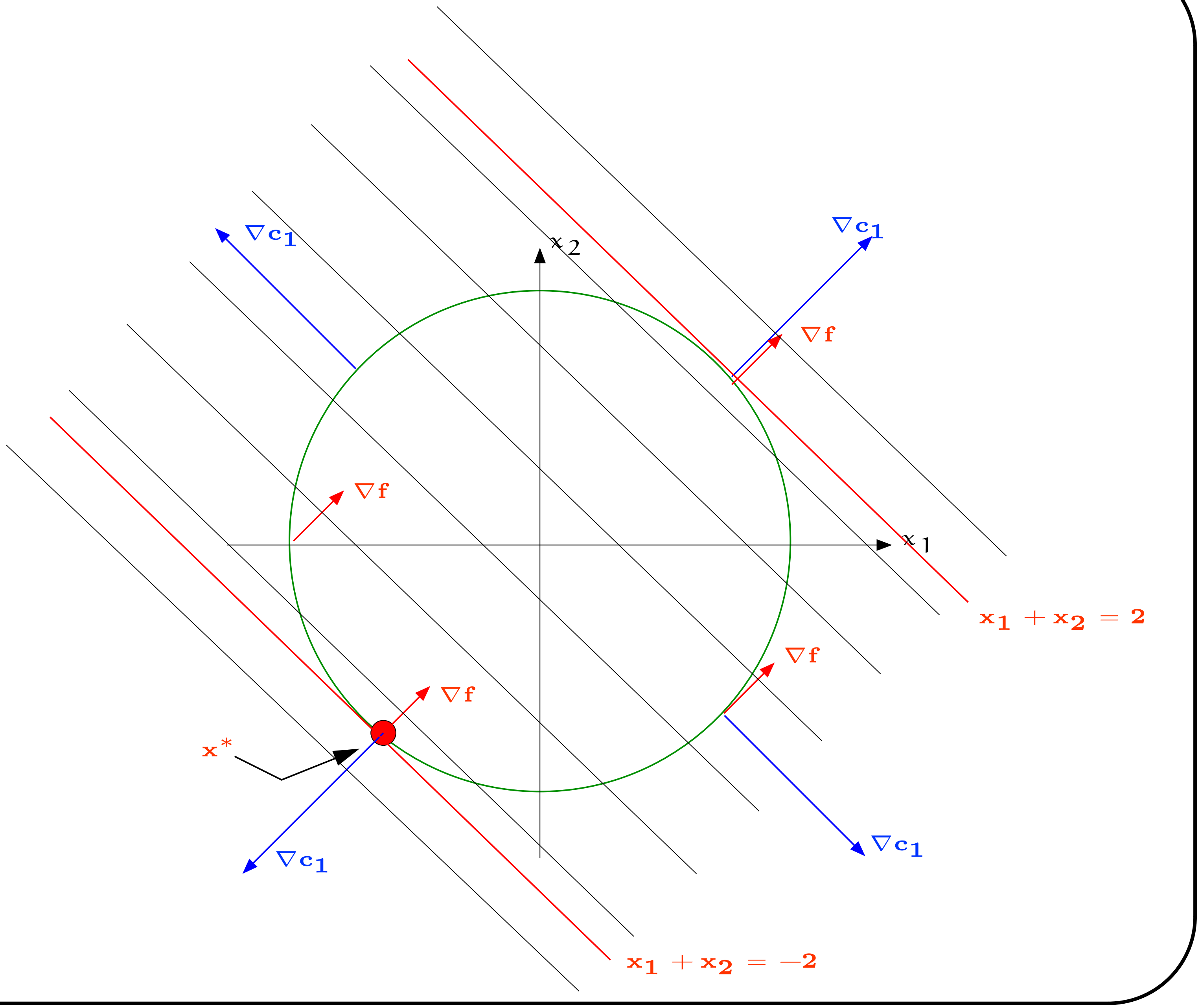


*Figure 15: Constraint and function gradients at feasible points.*

*In the language of (8.1), I have* $f(x) = x_1 + x_2$, $,\mathcal{I} = \emptyset$, $\mathcal{E} = \{1\}$, *and* $c_1(x) = x_1^2 + x_2^2 - 2$. *I can see by inspection that the feasible set for this problem is the circle of radius* $\sqrt{2}$ *centered at the origin — just the boundary of this circle, not its interior.*

*The solution* $x^*$ *is obviously* $(-1,-1)^T$. *From any other point on the circle, it is easy to find a way to move that* **stays feasible** *(that is, remains on the circle) while* **decreasing** $f$. *For instance, from the point* $x = (\sqrt{2}, 0)^T$ *any move in the clockwise direction around the circle has the desired effect.*

*I also see from Figure 15 that at the solution* $x^*$, *the* **constraint normal** $\nabla c_1(x^*)$ *is parallel to* $\nabla f(x^*)$. *That is, there is a scalar* $\lambda_1^*$ *such that*

$$\nabla f(x^*) = \lambda_1^* \nabla c_1(x^*). \tag{8.5}$$

*(In this particular case, I have* $\lambda_1^* = -\frac{1}{2}$*.)*

- I can "explain" (8.5) by examining first-order Taylor series approximations to the objective and constraint functions.
- To retain feasibility with respect to the function $c_1(x) = 0$, I require that $c_1(x+d) = 0$; that is $0 = c_1(x+d) \approx c_1(x) + \nabla c_1(x)^T d = \nabla c_1(x)^T d$.
- So the direction $d$ retains feasibility with respect to $c_1$, to first order, when it satisfies

$$\nabla c_1(x)^T d = 0. \tag{8.6}$$

- Similarly, a direction of improvement must produce decrease in $f$, so that $0 > f(x+d) - f(x) \approx \nabla f(x)^T d$ or, to first order,

$$\nabla f(x)^T d < 0. \tag{8.7}$$

- If there exists a direction $d$ that satisfies both (8.6) and (8.7), obviously improvement on my current point $x$ is possible.
- It follows that a **geometric necessary condition** for optimality for the problem (8.5) is that there exist **no direction d satisfying both** (8.6) and (8.7).
- In other words, at a solution point **there must be no feasible descent directions**.
- Using a sketch, it is easy to convince yourself that the only way such a direction can **not** exist is if $\nabla f(x)$ and $\nabla c_1(x)$ are parallel, that is, if the condition $\nabla f(x) = \lambda_1 \nabla c_1(x)$ holds at $x$, for some scalar $\lambda_1$.

- If this **algebraic** condition is **not** satisfied, the direction defined by

$$d = -\left(I - \frac{\nabla c_1(x)\nabla c_1(x)^T}{\|\nabla c_1(x)\|^2}\right)\nabla f(x) \tag{8.8}$$

satisfies both conditions (8.6) and (8.7). (See Exercise 1).

- By introducing the **Lagrangian function**

$$\mathcal{L}(x, \lambda_1) = f(x) - \lambda_1 c_1(x), \tag{8.9}$$

and noting that $\nabla_x \mathcal{L}(x, \lambda_1) = \nabla f(x) - \lambda_1 \nabla c_1(x)$, I can state the condition (8.5) equivalently as follows: at the solution $x^*$, there is a scalar $\lambda_1^*$ such that

$$\nabla_x \mathcal{L}(x^*, \lambda_1^*) = 0. \tag{8.10}$$

- This observation suggests that I can search for solutions of the equality-constrained problem (8.1) by searching for stationary points of the Lagrangian function.
- The scalar quantity $\lambda_1$ in (8.9) is called a **Lagrange multiplier** for the constraint $c_1(x) = 0$.

- Though the condition (8.5) — equivalently, (8.10) — appears to be **necessary** for an optimal solution of the problem (8.1), it is clearly not **sufficient**.

- For instance, in Example 8.1, (8.5) is satisfied at the point $x = (1, 1)^T$ (with $\lambda_1^* = \frac{1}{2}$), but this point is obviously not a solution — in fact, it **maximises** the function $f$ on the circle.

- In the case of equality-constrained problems, I cannot turn the condition (8.5) into a sufficient condition simply by placing some restriction on the sign of $\lambda_1$.

- To see this, consider replacing the constraint $x_1^2 + x_2^2 - 2 = 0$ by its negative $2 - x_1^2 - x_2^2 = 0$ in Example 8.1.

- The solution of the problem is not affected, but the value of $\lambda_1^*$ that satisfies the condition (8.5) changes from $\lambda_1^* = -\frac{1}{2}$ to $\lambda_1^* = \frac{1}{2}$.

**Example 8.2 (A Single Inequality Constraint)** *This is a slight modification of Example 8.1, in which the equality constraint is replaced by an inequality.*

*Consider the problem:*

$$\min x_1 + x_2 \quad \text{s.t.} \quad 2 - x_1^2 - x_2^2 \geq 0, \tag{8.11}$$

*for which the feasible region consists of the circle of problem 8.4 and its interior — see Figure 16.*

*Note that the constraint normal* $\nabla c_1$ *points toward the interior of the feasible region at each point on the boundary of the circle. By inspection, I see that the solution is still* $(-1, -1)$ *(the yellow circle in the Figure) and that the condition* (8.5) *holds for the value* $\lambda_1^* = \frac{1}{2}$.

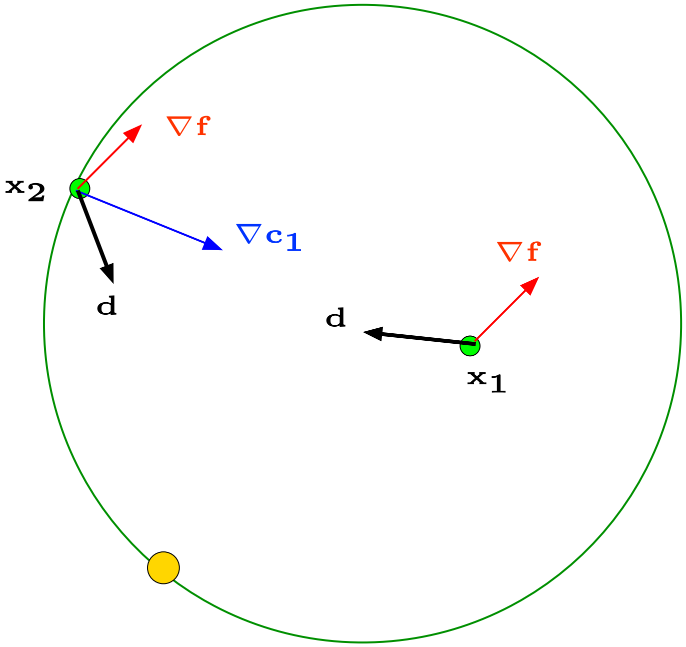


*Figure 16: Improvement directions* **d** *from two feasible points.*

*However, this inequality-constrained problem differs from the equality-constrained problem 8.4 of Example 8.1 in that the sign of the Lagrange multiplier plays a significant role, as I now explain.*

- As before, it seems reasonable that a given feasible point $x$ is **not** optimal if I can find a step $d$ that both retains feasibility and decreases the objective function $f$ to first order.
- The main difference between this problem and the equality-constrained version comes in the handling of the feasibility condition $c_1(x) \geq 0$.
- As in (8.7) the direction $d$ improves the objective function, to first order, if $\nabla f(x)^T d < 0$.
- Meanwhile, the direction $d$ retains feasibility if

$$0 \leq c_1(x+d) \approx c_1(x) + \nabla c_1(x)^T d, \tag{8.12}$$

so, to first order, feasibility is retained if

$$c_1(x) + \nabla c_1(x)^T d \geq 0. \tag{8.13}$$

- In determining whether a direction $\mathbf{d}$ exists that satisfies both (8.7) and (8.13) , I consider the following two cases, which are illustrated in Figure 16.

1. Consider first the case in which $x = x_1$ lies **strictly inside** the circle, so that the strict inequality $c_1(x) > 0$ holds.
   - In this case, *any* vector $d$ satisfies the condition (8.13), provided only that its length is sufficiently small.
   - In particular, whenever $\nabla f(x) \neq 0$, I can obtain a direction $d$ that satisfies both (8.7) and (8.13) by setting (see Exercise 2).

$$d = -\frac{c_1(x)}{\|\nabla c_1(x)\|} \frac{\nabla f(x)}{\|\nabla f(x)\|}. \tag{8.14}$$

- The only situation in which such a direction fails to exist is when $\nabla f(x) = 0$ or $\nabla c_1(x) = 0$.
- In the latter case (8.13) holds for all $\mathbf{d}$ so the factor $\|\nabla c_1(x)\|$ in the denominator can be omitted.
- **If $\|\nabla f(x)\| = 0$ at a point $x$ where $c_1(x) > 0$ then no feasible descent direction $d$ exists at $x$ and $x$ is an unconstrained minimum point.**
- **Summarising; in this case an optimal feasible point is one where $\nabla f(x) = 0$ or equivalently $\nabla_x \mathcal{L}(x^*, \lambda_1^*) = 0$ with $\lambda_1 = 0$.**

2. Consider now the case in which $x$ lies on the boundary of the circle, so that $c_1(x) = 0$.
   - The conditions (8.7) and (8.13) therefore become

$$\nabla f(x)^T d < 0, \quad \nabla c_1(x)^T d \geq 0.$$

   - The first of these conditions defines an open half-space, while the second defines a closed half-space, as illustrated in Figure 17.
   - It is clear from this figure that the two regions fail to intersect only when $\nabla f(x)$ and $\nabla c_1(x)$ point in the same direction, that is, when

$$\nabla f(x) = \lambda_1 \nabla c_1(x), \quad \text{for some} \quad \lambda_1 \geq 0. \tag{8.15}$$

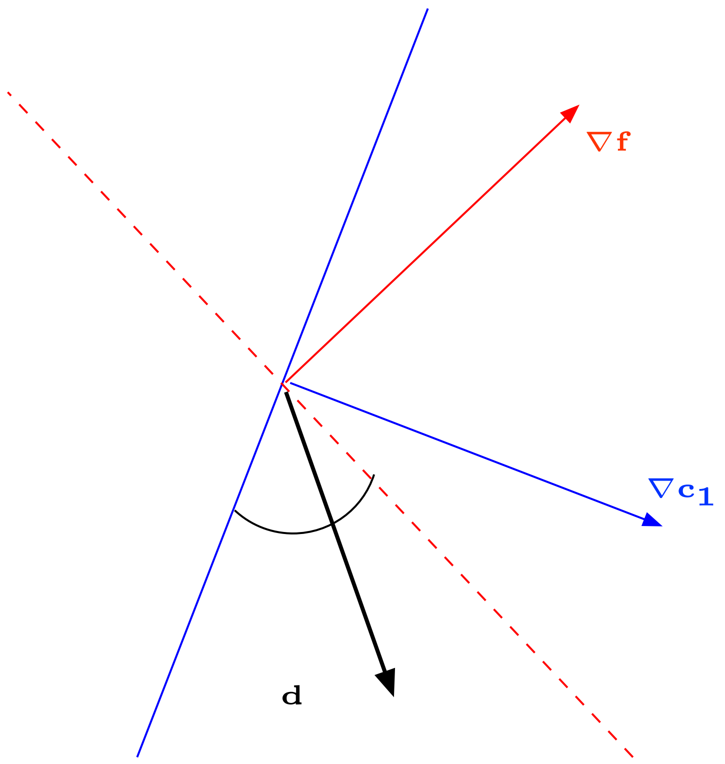


Figure 17: A direction $\mathbf{d}$ that satisfies (8.7) and (8.13).

- Note that the sign of the multiplier is significant here.
- If (8.15) were satisfied with a *negative* value of $\lambda_1$, then $\nabla f(x)$ and $\nabla c_1(x)$ would point in opposite directions, and I see from Figure 17 that the set of directions that satisfy both (8.7) and (8.13) would make up an entire open half-plane.
- The optimality conditions for both the interior and the boundary point cases can again be summarised neatly with reference to the Lagrangian function.
- When no first-order feasible descent direction exists at some point $x^*$, I have that

$$\nabla_x \mathcal{L}(x^*, \lambda_1^*) = 0, \quad \text{for some} \quad \lambda_1^* \geq 0, \tag{8.16}$$

Note that in this case (a feasible point on the boundary, $c_1(x) = 0$), $\lambda_1$ need not be zero at an optimal point.

- I can summarise the fact that $\lambda_1$ must be zero at an internal feasible optimal point ($c_1(x) > 0$) but need not be at a boundary feasible optimal point ($c_1(x) = 0$) withe the so-called **complementarity condition**;

$$\lambda_1^* c_1(x^*) = 0. \tag{8.17}$$

- This condition implies that the Lagrange multiplier $\lambda_1$ can be strictly positive **only when the corresponding constraint $c_1$ is active**.
- Conditions of this type play a central role in constrained optimisation, as I see in the sections that follow.
- In the interior point case I have that $c_1(x^*) > 0$, so (8.17) requires that $\lambda_1^* = 0$.
- Hence, (8.16) reduces to $\nabla f(x^*) = 0$, consistent with the comments in **blue** following (8.14).

See App. A.29 for another worked example with two inequality constraints.

In the next section, based on these Examples, I will state a set of necessary conditions; conditions that must hold at an optimal feasible point.

## 8.2 First-Order Optimality Conditions

- The three examples above suggest that a number of conditions are necessary for a point to be a solution for (8.1).
- These include
  - the equation $\nabla_x \mathcal{L}(x, \lambda) = 0$,
  - the nonnegativity of $\lambda_i$ for all inequality constraints $c_i(x)$
  - the complementarity condition $\lambda_i c_i(x) = 0$ that is required for all inequality constraints.
- Now we'll generalise the observations made in these examples and state the first-order optimality conditions in a rigorous fashion.

- In general, the Lagrangian for the constrained optimisation problem (8.1) is defined as

$$\mathcal{L}(x, \lambda) = f(x) - \sum_{i \in \mathcal{E} \cup \mathcal{I}} \lambda_i c_i(x). \tag{8.18}$$

- The **active set** $\mathcal{A}(x)$ at any feasible $x$ is the union of the set $\mathcal{E}$ with the indices of the active inequality constraints; that is,

$$\mathcal{A}(x) = \mathcal{E} \cup \{i \in \mathcal{I} | c_i(x) = 0\}. \tag{8.19}$$

- Next, I need to give more attention to the properties of the constraint gradients.
- The vector $\nabla c_i(x)$ is often called the **normal** to the constraint $c_i$ at the point $x$,
  - it is perpendicular (normal) to the contours of the constraint $c_i$ at $x$
  - in the case of an inequality constraint, it points toward the feasible side of this constraint.
  - Can you explain why???

- It can happen that $\nabla c_i(x) = 0$ for all feasible points $x$ due to the choice of formula used for the equality constraint $c_i(x) = 0$, so that the term $\lambda_i \nabla c_i(x)$ vanishes for all values of $\lambda_i$ and does not play a role in the Lagrangian gradient $\nabla_x \mathcal{L}$.

- For instance, if I replaced the constraint in (8.4) by the equivalent condition

$$c_1(x) = \left(x_1^2 + x_2^2 - 2\right)^2 = 0$$

I would have that

  - $\nabla c_1(x) = 0$ for all feasible points $x$
  - and the condition $\nabla f(x) = \lambda_1 \nabla c_1(x)$ no longer holds at the optimal point $(-1, -1)^T$.

- I usually add an extra condition called a **constraint qualification** to ensure that such "degenerate" behavior does not occur at the value of $x$ in question.
- One such constraint qualification— often used in the design of algorithms—is the one defined as follows:

  **Definition 8.1 (LICQ)** *Given the point* $x^*$ *and the active set* $\mathcal{A}(x^*)$ *defined by* (8.19), *I say that the linear independence constraint qualification (LICQ) holds if the set of active constraint gradients* $\{\nabla c_i(x^*), i \in \mathcal{A}(x^*)\}$ *is linearly independent.*
- Note that if this condition holds, none of the active constraint gradients can be zero.
- Why????

- The LICQ is the final piece in the jigsaw.
- Now I can state the following optimality conditions for a general nonlinear programming problem (8.1).
- These conditions provide the foundation for algorithms for the solution of Constrained Optimisation problems.
- They are called **first-order conditions** because they concern themselves with properties of the gradients (first-derivative vectors) of the objective and constraint functions.

**Theorem 8.1 (First-Order Necessary Conditions)** *Suppose that $x^*$ is a local solution of (8.1) and that the LICQ (Def. 8.1) holds at $x^*$. Then there is a Lagrange multiplier vector $\lambda^*$, with components $\lambda_i^*, i \in \mathcal{E} \cup \mathcal{I}$, such that the following conditions are satisfied at $(x^*, \lambda^*)$*

*(Conditions 8.20b and 8.20c simply require that the constraints be satisfied — which is why they are green...)*

$$\nabla_x \mathcal{L}(x^*, \lambda^*) = 0, \tag{8.20a}$$

$$c_i(x^*) = 0, \quad \textit{for all} \quad i \in \mathcal{E}, \tag{8.20b}$$

$$c_i(x^*) \geq 0, \quad \textit{for all} \quad i \in \mathcal{I}, \tag{8.20c}$$

$$\lambda_i^* \geq 0, \quad \textit{for all} \quad i \in \mathcal{I}, \tag{8.20d}$$

$$\lambda_i^* c_i(x^*) = 0, \quad \textit{for all} \quad i \in \mathcal{E} \cup \mathcal{I} \tag{8.20e}$$

The conditions (8.23a–8.20e) are the **Karush-Kuhn-Tucker conditions** or **KKT conditions** for short.

- Because the complementarity condition (8.20e) implies that the Lagrange multipliers corresponding to inactive inequality constraints are zer o, I can omit the terms for indices $i \notin \mathcal{A}(x^*)$ from (8.23a) and rewrite this condition as

$$0 = \nabla_x \mathcal{L}(x^*, \lambda^*) = \nabla f(x^*) - \sum_{i \in \mathcal{A}(x^*)} \lambda_i^* \nabla c_i(x^*). \tag{8.21}$$

- A special case of complementarity is important and deserves its own definition:

**Definition 8.2 (Strict Complementarity)** *Given a local solution $x^*$ of (8.1) and a vector $\lambda^*$ satisfying (8.23a–8.20e), I say that the strict complementarity condition holds if exactly one of $\lambda_i^*$ and $c_i(x^*)$ is zero for each index $i \in \mathcal{I}$. In other words, I have that $\lambda_i^* > 0$ for each $i \in \mathcal{I} \cap \mathcal{A}(x^*)$.*

- For a given problem (8.1) and solution point $x^*$ there may be many vectors $\lambda^*$ for which the conditions (8.23a–8.20e) are satisfied.

- When the LICQ holds, however, the optimal $\lambda^*$ is unique (see the exercises).

- **Example 8.3** *A toy problem:*
  - *Let* $f(x) = x_1 + x_2 + 2$,
  - $c_1(x) = 2 - (x_1 - 1)^2 - (x_2 - 1)^2$,
  - $c_2(x) = 2 - (x_1 - 1)^2 - (x_2 + 1)^2$
  - $c_3(x) = x_1$.
  - *Sketch the problem.*
  - *It is easy to see that* $x^* = (0, 0)^T$.
  - *Find the Lagrange multipliers at* $x^*$ *and check whether the LICQ is satisfied.*

I will prove Theorem 8.1 in two steps.

- First in Thm. 8.2 I prove the special case (Problem $\mathsf{E}$) where all the constraints are equality constraints:

$$\mathsf{E}: \min_{x \in \mathbb{R}^n} f(x) \quad \text{subject to } c_i(x) = 0,\ i \in \mathcal{E}, \tag{8.22}$$

- Then I extend the result to include inequality constraints — proving Thm. 8.1.

**Theorem 8.2 (1st-Order N. Conds. — Equality Constraints)** *Suppose that $x^*$ is a local solution of (8.22) and that the LICQ (Def. 8.1) holds at $x^*$. Then there is a Lagrange multiplier vector $\lambda^*$, with components $\lambda_i^*, i \in \mathcal{E}$, such that the following conditions are satisfied at $(x^*, \lambda^*)$*

$$\nabla_x \mathcal{L}(x^*, \lambda^*) = 0, \tag{8.23a}$$

$$c_i(x^*) = 0, \quad \textit{for all} \quad i \in \mathcal{E}, \tag{8.23b}$$

### 8.2.1 Background Material for KKT Theorem

I need some new ideas before I go ahead with the proof in Section 8.2.2.

**Definition 8.3 (Limit Point)** *A vector $\bar{x} \in \mathbb{R}^n$ is a limit point of a sequence $\{x_k\}$ if there is a subsequence of $\{x_k\}$ that converges to $\bar{x}$. (The Bolzano-Weirstrass Thm — which applies to $\mathbb{R}^n$ — states that every sequence on a closed and bounded set has a convergent subsequence.)*

**Example 8.4** *The sequence $\{1, -1, 1, -1, \dots\}$ is defined on the closed bounded set $\{-1, 1\}$. It is obviously not convergent but equally obviously has convergent subsequences with converging to $1$ and $-1$ so has $1$ & $-1$ as limit points.*

**Example 8.5** *The sequence $\{1, 2, 3, \dots\}$ is* **not** *defined on a closed bounded set and has no convergent subsequences and therefore no limit points.*

As a means to the end of proving Thm. 8.2 I can define a **penalty** function $\mathsf{F}^k$ for the equality-constrained minimisation problem $\mathsf{E}$ (8.22) as follows.

**Definition 8.4 (Penalty Function)** *Let $x^*$ be a local minimum of problem $\mathsf{E}$. Then I define*

$$\mathsf{F}^k(x) = f(x) + \frac{k}{2}\|c(x)\|^2 + \frac{\alpha}{2}\|x - x^*\|^2. \tag{8.24}$$

- Note that $\|c(x)\|^2 \equiv \sum_{i\in\mathcal{E}} c_i(x)^2$.
- The $\alpha$–term is included so that $x^*$ is a **strict** local minimum of

  $$\mathsf{E}\alpha : \min_{x\in\mathbb{R}^n} f(x) + \frac{\alpha}{2}\|x - x^*\|^2 \quad \text{subject to } c_i(x) = 0,\ i \in \mathcal{E}$$

  even if $x^*$ is only a weak local minimum of the original problem $\mathsf{E}$.

- Penalty functions have the nice property that in the limit as $k \to \infty$, the **unconstrained** minimum of the penalty function is equal to the solution to the actual constrained problem (in this case $E$).
- This is intuitively reasonable as to minimise $F^k(x)$ for any given $k$ I need to reduce $\|c(x)\|^2$ towards zero (moving to the feasible region) and also minimise $f(x)$.
- It seems plausible that the bigger $k$ is, the more I "penalise" the $\|c(x)\|^2$ term in $F^k(x)$.
- The trade-off between reducing $f(x)$ and driving $\|c(x)\|^2$ to zero moves more and more towards the latter as $k$ increases.
- So I can expect that in the limit as $k \to \infty$ the minimum of $F^k(x)$ will be a feasible point that minimises $f(x)$, a solution to $E$.
- I need to prove this.

**Theorem 8.3 (Penalty Function)** *Given the equality-constrained problem* $\mathsf{E}$ (8.22), *then the unconstrained minima* $x_k$ *of* $F^k(x)$ *as defined in* (8.24) *above have the property that*

$$\lim_{k \to \infty} x_k = x^* \qquad \textit{a local minimum of problem } \mathsf{E}.$$

**Proof:**

- Saying that $x^*$ is a local minimum of $\mathsf{E}$ means that $x^*$ must be the point at which $f(x)$ takes the smallest value for all feasible points $x$ **near** $x^*$ — i.e. within a distance $\varepsilon$ for $\varepsilon > 0$ sufficiently small.
- In mathematical notation; $\exists \varepsilon > 0$ s.t. $f(x^*) \leq f(x)$ for all $x$ s.t. $c(x) = 0$ and $x \in S_\varepsilon \equiv \{x | \|x - x^*\| \leq \varepsilon\}$.
- For each $k > 0$, let $x^k$ be an optimal solution to

$$\min F^k(x) \quad \text{s.t.} \quad x \in S_\varepsilon.$$

- I know that a solution exists as $S_\varepsilon$ is closed and bounded.
- R.T.P. that $x_k \to x^*$ as $k \to \infty$.
- First I note that because $F_k$ has a minimum at $x_k$ ;

$$\forall k, F_k(x_k) \equiv f(x_k) + \frac{k}{2}\|c(x_k)\|^2 + \frac{\alpha}{2}\|x_k - x^*\|^2 \leq F_k(x^*) \equiv f(x^*).$$

- As $f(x_k)$ is bounded on $S_\varepsilon$ for all $k$ I must have $\lim_{k\to\infty} \|c(x_k)\| = 0$ as otherwise the LHS of the inequality will be unbounded in the limit due to the $\frac{k}{2}$ factor in $\frac{k}{2}\|c(x_k)\|^2$.
- So every limit point $\overline{x}$ of $\{x_k\}$ is feasible (satisfies $c(\overline{x}) = 0$).
- Also, from the above inequality I certainly have

$$\forall k, f(x_k) + \frac{\alpha}{2}\|x_k - x^*\|^2 \leq f(x^*).$$

- Pick any convergent subsequence and take the limit as $k \to \infty$.
- I have

$$f(\overline{x}) + \frac{\alpha}{2}\|\overline{x} - x^*\|^2 \le f(x^*).$$

- Since $\overline{x} \in S_\varepsilon$ (as $S_\varepsilon$ is closed) and $\overline{x}$ is feasible it follows that $f(x^*) \le f(\overline{x})$ so

$$f(\overline{x}) + \frac{\alpha}{2}\|\overline{x} - x^*\|^2 \le f(x^*) \le f(\overline{x})$$

  which requires that $\|\overline{x} - x^*\|^2 = 0$.
- So $\overline{x} = x^*$ and I have $x_k \to x^*$ as $k \to \infty$.

■

For $k$ sufficiently large $x_k$ must be an interior point of the closed sphere $S_\varepsilon$ so $x_k$ must be an unconstrained local minimum of $F_k$ for $k$ sufficiently large.

### 8.2.2 Proof of Theorem 8.2 KKT (Equality Constraints)

**Proof:**

- From the first order necessary condition Thm. 3.2 for **unconstrained** problems I have for $k$ sufficiently large (so that $x_k$ must be an unconstrained local minimum of $F_k$) that:

$$0 = \nabla F_k(x_k) = \nabla f(x_k) + k \nabla c(x_k) c(x_k) + \alpha(x_k - x^*). \quad (8.25)$$

- As $\nabla c(x^*)$ has full rank ($m$) by the LICQ, the same must be true for $\nabla c(x_k)$ when $k$ is sufficiently large as $c(x)$ is $C^1$.
- For such $k$, $\nabla c(x_k)^T \nabla c(x_k)$ must be invertible ($m \times m$).

- Now pre-multiply (8.25) by $\left[\nabla c(x_k)^T \nabla c(x_k)\right]^{-1} \nabla c(x_k)^T$ giving:

$$\begin{aligned} 0 = \left[\nabla c(x_k)^T \nabla c(x_k)\right]^{-1} \nabla c(x_k)^T \nabla f(x_k) + kc(x_k) + \\ \alpha \left[\nabla c(x_k)^T \nabla c(x_k)\right]^{-1} \nabla c(x_k)^T (x_k - x^*). \end{aligned}$$

- So $kc(x_k) = -\left[\nabla c(x_k)^T \nabla c(x_k)\right]^{-1} \nabla c(x_k)^T \{\nabla f(x_k) + \alpha(x_k - x^*)\}$.
- Now, taking the limit as $k \to \infty$ (and $x_k \to x^*$),

$$kc(x_k) \to -\lambda^* \equiv -\left[\nabla c(x^*)^T \nabla c(x^*)\right]^{-1} \nabla c(x^*)^T \nabla f(x^*) \quad (8.26)$$

- Using this result for $\lim\limits_{k\to\infty} kc(x_k)$ and taking $\lim\limits_{k\to\infty}$ in (8.25), I have $0 = \nabla f(x^*) - \nabla c(x^*)\lambda^*$.
- This is the required first-order KKT necessary condition for equality-constrained problems. ■

### 8.2.3 KKT conditions for Inequality Constrained problems

**Proof:** (of Thm 8.1) I can extend Thm. 8.2 to inequality-constrained problems in two steps.

1. - Note that inequality constraints that are **inactive** at the optimal point $x^*$ may be ignored as they cannot affect the solution.
   - Also, the the inequality constraints that are **active** at $x^*$ may be treated as equality constraints in the sense that if $x^*$ is a solution to the problem (8.1) then it is also a solution to

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{subject to} \begin{cases} c_i(x) = 0 & i \in \mathcal{E}, \\ c_i(x) = 0 & i \in \mathcal{A}(x^*) \setminus \mathcal{E} \end{cases} \tag{8.27}$$

- Of course I don't know which inequality constraints are active at $x^*$
  - so (8.27) does not give me a solution method
  - but it **does** mean that I can apply the KKT conditions for an equality-constrained problem to (8.27).

- So (for free) I get (most of) the KKT first-order necessary conditions for a general mixed problem (8.1):

$$\nabla_x \mathcal{L}(x^*, \lambda^*) = 0, \tag{8.27a}$$

$$c_i(x^*) = 0, \quad \text{for all} \quad i \in \mathcal{E}, \tag{8.27b}$$

$$c_i(x^*) \geq 0, \quad \text{for all} \quad i \in \mathcal{I}, \tag{8.27c}$$

$$\textbf{The odd one out} \longrightarrow \lambda_i^* \geq \mathbf{0}, \quad \text{for all} \quad i \in \mathcal{I}, \tag{8.27d}$$

$$\lambda_i^* c_i(x^*) = 0, \quad \text{for all} \quad i \in \mathcal{E} \cup \mathcal{I} \tag{8.27e}$$

where (as in (8.21))

$$\nabla_x \mathcal{L}(x^*, \lambda^*) \equiv \nabla f(x^*) - \sum_{i \in \mathcal{A}(x^*)} \lambda_i^* \nabla c_i(x^*).$$

- The first condition (8.27a) follows immediately from applying the (equality-constrained) version of the KKT conditions to (8.27) — I set the multipliers corresponding to inactive (at $x^*$ ) inequality constraints to zero as they cannot affect the solution.
- The second (8.27b) and third (8.27c) green conditions just require that all the constraints be satisfied at the optimal point $x^*$.
- I will have to prove the **fourth condition** (8.27d) separately as it does not follow from the (equality-constrained) version of the KKT conditions.

- It is easy to check that the (fifth) complementarity condition (8.27e) holds.
  - It is trivial for the equality constraints — it means that there is no restriction on the signs of the corresponding Lagrange multipliers.
  - For the **inactive** inequality constraints ($c_i(x^*) > 0$) it says the corresponding Lagrange multipliers must be zero as these constraints may be omitted from the Lagrangian without changing the solution.
  - Finally, it says nothing about the multipliers corresponding to the active constraints ($c_i(x^*) = 0$).

2. - I finally need to prove that the **fourth condition** (8.27d) is necessary for $x^*$ to be an optimal solution of (8.1).
   - Define the functions $c_j^-(x) = \min\{0, c_j(x)\}$, $j \in \mathcal{I}$.
   - Obviously $c_j^-(x) \le 0$, $j \in \mathcal{I}$.
   - If the $j^{th}$ constraint is satisfied then $c_j^-(x)$ is zero.
   - If the constraint is violated then $c_j^-(x) = c_j(x) < 0$.
   - For each value of $k = 0, 1, \cdots$ I define the penalty problem corresponding to (8.1):

   $$\min F^k(x) \equiv f(x) + \frac{k}{2}\sum_{i\in\mathcal{E}} c_i(x)^2 + \frac{k}{2}\sum_{i\in\mathcal{I}} \left(c_i^-(x)\right)^2 + \frac{\alpha}{2}\|x - x^*\|^2$$

   provided that $x \in S_\varepsilon$ as before.

- As in the derivation of the KKT conditions for an equality-constrained problem, I can apply the first-order necessary conditions for an unconstrained problem to this new version of $F_k$ — note that $\left(c_i^-(x)\right)^2$ is continuously differentiable with gradient $2c_i^-(x)\nabla c_i(x)$.
- A similar argument (Exercise) to that used in the proof of Thm. 8.2 leads to the conclusions (compare with (8.26)):

$$\lambda_i^* = -\lim_{k\to\infty} kc_i(x_k), i \in \mathcal{E}$$
$$\lambda_i^* = -\lim_{k\to\infty} kc_i^-(x_k), i \in \mathcal{I}.$$

- But $c_i^-(x_k) \le 0$ by definition so $\lambda_i^* \ge 0, i \in \mathcal{I}$.

$\blacksquare$

I will illustrate the KKT conditions with another example.

**Example 8.6** *Minimise the function* $\left(x_1 - \frac{3}{2}\right)^2 + \left(x_2 - \frac{1}{2}\right)^4$ *in the feasible region described by the four constraints*

$$x_1 + x_2 \le 1, \quad x_1 - x_2 \le 1, \quad -x_1 + x_2 \le 1, \quad -x_1 - x_2 \le 1.$$

*Graphically the feasible region is just the diamond-shaped region bounded by* $(1, 0)$, $(0, 1)$, $(-1, 0)$ *and* $(0, -1)$.

*By restating the constraints in the standard form of* (8.1) *the problem becomes*

$$\min_x \left(x_1 - \frac{3}{2}\right)^2 + \left(x_2 - \frac{1}{2}\right)^4 \quad \text{s.t.} \quad \begin{bmatrix} 1 - x_1 - x_2 \\ 1 - x_1 + x_2 \\ 1 + x_1 - x_2 \\ 1 + x_1 + x_2 \end{bmatrix} \geq 0. \quad (8.28)$$

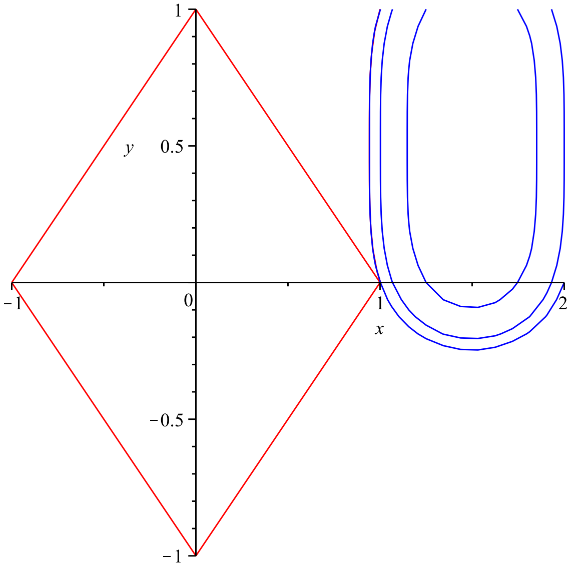


*Figure 18: Inequality-constrained Example 8.6*

*It is clear from the Figure that the solution is* $\mathsf{x}^* = (1,0)^\mathsf{T}$.

*The first and second constraints in* (8.28) *are active at this point. Denoting them by* $c_1$ *and* $c_2$ *(and the inactive constraints by* $c_3$ *and* $c_4$*), I have*

$$\nabla f(x^*) = \begin{bmatrix} -1 \\ -\frac{1}{2} \end{bmatrix}, \nabla c_1(x^*) = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \nabla c_2(x^*) = \begin{bmatrix} -1 \\ 1 \end{bmatrix}.$$

*Therefore, the KKT conditions (8.23a–8.20e) are satisfied when I set*

$$\lambda^* = \left(\frac{3}{4}, \frac{1}{4}, 0, 0\right)^T.$$

### 8.2.4 Sensitivity Analysis

That Lagrange multipliers are useful should now be clear but what do they "mean"? The value of each Lagrange multiplier $\lambda_i^*$ tells me something about the **sensitivity** of the optimal objective value $f(x^*)$ to the presence of constraint $c_i$. To put it another way, $\lambda_i^*$ indicates how hard $f$ is "pushing" or "pulling" against the particular constraint $c_i$.

I can illustrate this point with a little analysis. When I choose an inactive constraint $i \notin \mathcal{A}(x^*)$ such that $c_i(x^*) > 0$, the solution $x^*$ and function value $f(x^*)$ are quite indifferent to whether this constraint is present or not. If I perturb $c_i$ by a tiny amount, it will still be inactive and $x^*$ will still be a local solution of the optimisation problem. Since $\lambda_i^* = 0$ from (8.20e), the Lagrange multiplier indicates accurately that constraint $i$ is not significant.

Suppose instead that constraint $i$ is active, and I perturb the right-hand-side of this constraint a little, requiring, say, that $c_i(x) \geq -\varepsilon$ instead of $c_i(x) \geq 0$. Note:

- $\varepsilon > 0$ corresponds to “weakening” the constraint (making the feasible region bigger) and $\varepsilon < 0$ to “tightening” it.
- I expect that weakening the constraint will **decrease** the value of the optimal $f(x^*(\varepsilon))$ as the minimum over a larger set will be smaller (or the same).

Suppose that $\varepsilon$ is sufficiently small that the perturbed solution $x^*(\varepsilon)$ still has the same set of active constraints, and that the Lagrange multipliers are not much affected by the perturbation.

I then find that for all $j \in \mathcal{A}(x^*)$ with $j \neq i$;

$$\begin{aligned} -\varepsilon = c_i(x^*(\varepsilon)) - c_i(x^*) &\approx (x^*(\varepsilon) - x^*)^T \nabla c_i(x^*), \\ 0 = c_j(x^*(\varepsilon)) - c_j(x^*) &\approx (x^*(\varepsilon) - x^*)^T \nabla c_j(x^*). \end{aligned}$$

The value of $f(x^*(\varepsilon))$, meanwhile, can be estimated with the help of (8.23a). I have

$$\begin{aligned} f(x^*(\varepsilon)) - f(x^*) &\approx (x^*(\varepsilon) - x^*)^T \nabla f(x^*) \\ &= \sum_{j \in \mathcal{A}(x^*)} \lambda_j^* (x^*(\varepsilon) - x^*)^T \nabla c_j(x^*) \\ &\approx -\varepsilon \lambda_i^*. \end{aligned}$$

Dividing across by $\varepsilon$ and taking limits as $\varepsilon \to 0$, I see that the family of solutions $x^*(\varepsilon)$ satisfies

$$\frac{df(x^*(\varepsilon))}{d\varepsilon} = -\lambda_i^*. \qquad (8.29)$$

So my simple sensitivity analysis of this problem concludes that:

- If $\lambda_i^* \|\nabla c_i(x^*)\|$ is large, then the optimal value is sensitive to the placement of the $i$th constraint,
- If this quantity is small, the dependence is weak.
- If $\lambda_i^*$ is exactly zero for some active constraint, small perturbations to $c_i$ in some directions will hardly affect the optimal objective value at all; the change is zero, to first order.

(Of course the above argument breaks down if $\|\nabla c_i(x^*)\| = 0$ as then the LHS of $-\varepsilon \approx (x^*(\varepsilon) - x^*)^T \nabla c_i(x^*)$ is non-zero while the RHS is zero.

In other words, a first-order approximation is no longer valid.)

Notes:

- In general if I perturb an equality constrained problem
$$\min f(x) | c_i(x) = 0, i \in \mathcal{E}$$
by solving
$$\min f(x) | c_i(x) = u_i, i \in \mathcal{E}$$
then
$$\nabla_u f(x(u)) = -\lambda(u) \tag{8.30}$$
where $x(u)$ is the optimal solution to the perturbed problem and $\lambda(u)$ is the vector of Lagrange multipliers for the perturbed problem. (See App A.30 for details.)
- This result still holds for inequality-constrained problems if I add the inequality constraints that are binding at the solution to the set of equality constraints.
- Obviously, (8.30) generalises the informal result (8.29).

This discussion motivates the definition below, which classifies constraints according to whether or not their corresponding Lagrange multiplier is zero.

**Definition 8.5** *Let $x^*$ be a solution of the problem (8.1), and suppose that the KKT conditions (8.23a–8.20e) are satisfied.*

- *I say that an inequality constraint $c_i$ is* **strongly active** *or* **strongly binding** *if $i \in \mathcal{A}(x^*)$ and $\lambda_i^* > 0$ for some Lagrange multiplier $\lambda^*$ satisfying (8.23a–8.20e).*
- *I say that $c_i$ is* **weakly** *active if $i \in \mathcal{A}(x^*)$ and $\lambda_i^* = 0$ for all $\lambda^*$ satisfying (8.23a–8.20e).*

## 8.3 Second-Order Conditions

- So far, I have described the first-order necessary conditions—the KKT conditions—which tell me how the first derivatives of $f$ and the active constraints $c_i$ are related at $x^*$.
- They are not sufficient conditions.
- Although for the simple inequality constrained examples in Sec 8.1 the non-negativity condition on multipliers corresponding to active inequality constraints allowed me to distinguish maximum from minimum points, this is not always so, as I will show .
- Just as for unconstrained problems I need second-order conditions to distinguish maximum, minimum and stationary points.

- First I need to define “feasible directions” — directions that maintain feasibility:

**Definition 8.6 (Feasible Directions)** *Given a point $x^*$ and the active constraint set $\mathcal{A}(x^*)$ defined by (8.19), the set $\mathcal{F}$ of* **feasible directions** *is defined by*

$$\mathcal{F}(x^*) = \left\{ \alpha d \,\middle|\, \alpha > 0, \quad \begin{matrix} d^T \nabla c_i^* = 0, & \textit{for all} & i \in \mathcal{E}, \\ d^T \nabla c_i^* \geq 0, & \textit{for all} & i \in \mathcal{A}(x^*) \cap \mathcal{I} \end{matrix} \right\} \tag{8.31}$$

*$\mathcal{F}$ is the set of vectors that (to first order) maintain feasibility — stay “on” equality and “inside or on” inequality constraints.*

- When the (first-order) KKT conditions are satisfied, a move along a "feasible direction" ($d$ in $\mathcal{F}$) either increases the first-order approximation to the objective function (that is, $d^T \nabla f(x^*) > 0$), or else keeps this value the same (that is, $d^T \nabla f(x^*) = 0$).

- What implications does optimality have for the **second** derivatives of $f$ and the constraints $c_i$?

- I will show in this Section that these derivatives play a "tiebreaking" role.

- For the feasible directions $d \in \mathcal{F}$ for which $d^T \nabla f(x^*) = 0$, I cannot determine from first derivative information alone whether a move along this direction will increase or decrease the objective function $f$.

- Second-order conditions examine the second derivative terms in the Taylor series expansions of $f$ and $c_i$, to see whether this extra information resolves the issue of increase or decrease in $f$.
- In summary: the second-order conditions analyse the curvature of the Lagrangian function in the “undecided” directions — the directions $d \in \mathcal{F}$ for which $d^T \nabla f(x^*) = 0$.
- Since I am discussing second derivatives, stronger smoothness assumptions are needed here than in the previous Sections.
- For the purpose of this section, $f$ and $c_i, i \in \mathcal{E} \cup \mathcal{I}$, are all assumed to be twice continuously differentiable.

- Given $\mathcal{F}$ from Definition 8.6 and some Lagrange multiplier vector $\lambda^*$ satisfying the KKT conditions (8.23a–8.20e) , I define a restricted set of feasible directions; the set of **critical** directions $\mathcal{C}(\lambda^*)$ a subset of $\mathcal{F}$ by

$$\mathcal{C}(\lambda^*) = \left\{ d \in \mathcal{F} | \nabla c_i(x^*)^T d = 0, \text{ for all } \ i \in \mathcal{A}(x^*) \cap \mathcal{I} \text{with} \, \lambda_i^* > 0 \right\}.$$

- Critical directions “keep **strongly** active constraints active” to first order.
- An equivalent definition is:

$$d \in \mathcal{C}(\lambda^*) \Leftrightarrow \begin{cases} \nabla c_i(x^*)^T d = 0, & \text{for all } i \in \mathcal{E}, \\ \nabla c_i(x^*)^T d = 0, & \text{for all } i \in \mathcal{A}(x^*) \mathcal{I} \text{ with } \lambda_i^* > 0, \\ \nabla c_i(x^*)^T d \geq 0, & \text{for all } i \in \mathcal{A}(x^*) \mathcal{I} \text{ with } \lambda_i^* = 0. \end{cases} \tag{8.32}$$

- From the definition (8.32) and the fact that $\lambda_i^* = 0$ for all inactive components $i \in \mathcal{I} \backslash \mathcal{A}(x^*)$, it follows immediately that

$$d \in \mathcal{C}(\lambda^*) \Rightarrow \lambda_i^* \nabla c_i(x^*)^T d = 0 \text{ for all } i \in \mathcal{E} \cup \mathcal{I}. \tag{8.33}$$

- From the first KKT condition (8.23a) and the definition (8.18) of the Lagrangian function, I have that

$$d \in \mathcal{C}(\lambda^*) \Rightarrow d^T \nabla f(x^*) = \sum_{i \in \mathcal{E} \cup \mathcal{I}} \lambda_i^* d^T \nabla c_i(x^*) = 0. \tag{8.34}$$

- So the set $\mathcal{C}(\lambda^*)$ contains directions from $\mathcal{F}$ for which it is not clear from first derivative information alone whether $f$ will increase or decrease — for this reason $\mathcal{C}$ is often called the set of **stationary directions**.

The first theorem defines a **necessary** condition involving the second derivatives: If $x^*$ is a local solution, then the curvature of the Lagrangian along stationary directions ($d \in \mathcal{C}(\lambda^*)$) must be nonnegative.

**Theorem 8.4 (Second-Order Necessary Conditions)** *Suppose that $x^*$ is a local solution of (8.1) and that the LICQ (Def. 8.1) constraint qualification is satisfied. Let $\lambda^*$ be a Lagrange multiplier vector such that the KKT conditions (8.23a–8.20e) are satisfied, and let $\mathcal{C}(\lambda^*)$ be defined as above. Then*

$$d^T \nabla_{xx} \mathcal{L}(x^*, \lambda^*) d \geq 0, \quad \textit{for all stationary directions } d \in \mathcal{C}(\lambda^*). \tag{8.35}$$

- The general proof of Thm. 8.4 is quite technical so I will just give the proof for the special case of equality constraints.
- Note that (8.32) implies that for equality-constrained problems,

$$\mathcal{C} = \left\{ d | \nabla c_i(x^*)^T d = 0, \quad \text{for all } i \in \mathcal{E} \right\} \tag{8.36}$$

- I state and prove a slightly weaker and easily proved version of the general result in Thm. 8.6 below.
- See App. A.23 for a proof of the general result for inequality-constrained problems.

So RTP that

**Theorem 8.5 (Second-Order N. C.'s — E. Constr.)** *Suppose that* $x^*$ *is a local solution of* (8.22) *and that the LICQ (Def. 8.1) constraint qualification is satisfied. Let* $\lambda^*$ *be a Lagrange multiplier vector such that the first-order (KKT) necessary conditions are satisfied, and let* $\mathcal{C}(\lambda^*)$ *be defined as above. Then*

$$d^T \nabla_{xx} \mathcal{L}(x^*, \lambda^*) d \geq 0,$$

*for all vectors* $d$ *such that* $d \perp \nabla c_i(x^*)$, *for each* $i \in \mathcal{E}$.

*i.e. for all vectors* $d \in \mathcal{C}$.

**Proof:**

- Just as in the proof of Thm. 8.2, I appeal to the corresponding second-order necessary conditions Thm. 3.4 for an unconstrained problem and apply them to the penalty function $F_k$ defined in Def. 8.24.
- So the Hessian of $F_k$;
$$\nabla^2 F_k(x_k) = \nabla^2 f(x_k) + k\nabla c(x_k)\nabla c^T(x_k) + \\ k\sum_{i=1}^{m} c_i(x_k)\nabla^2 c_i(x_k) + \alpha I$$
must be positive semidefinite. for $k$ sufficiently large and for $\alpha > 0$.
- (As for $k$ sufficiently large $x_k$ must be an interior point of the closed sphere $S_\varepsilon \equiv \{x | \|x - x^*\| \le \varepsilon\}$ so $x_k$ must be an unconstrained local minimum of $F_k$ for $k$ sufficiently large.)

- Pick any fixed stationary direction $\mathbf{d} \in \mathcal{C}$ (i.e. $\mathbf{d} \perp \nabla c_i(x^*)$ for all $i \in \mathcal{E}$) and define $\mathbf{d}_k$ to be the projection of $\mathbf{d}$ onto the null space of $\nabla c(x_k)^T$;

$$\mathbf{d}_k \equiv \mathbf{d} - \nabla c(x_k) \left[\nabla c(x_k)^T \nabla c(x_k)\right]^{-1} \nabla c(x_k)^T \mathbf{d}.$$

- (Check that $\mathbf{d}_k \perp \nabla c(x_k)$.) It follows that

$$\begin{aligned} 0 \le {\mathbf{d}_k}^T \nabla^2 F_k(x_k) \mathbf{d}_k = \\ {\mathbf{d}_k}^T \left( \nabla^2 f(x_k) + k \sum_{i=i^m} c_i(x_k) \nabla^2 c_i(x_k) \right) \mathbf{d}_k \\ + \alpha \|\mathbf{d}_k\|^2. \end{aligned} \tag{8.37}$$

- I know from (8.26) that $kc_i(x_k) \to \lambda_i^*$ as $k \to \infty$.
- Check that, by definition of $\mathbf{d}_k$, $\mathbf{d}_k \to \mathbf{d}$ as $k \to \infty$.

- So
$$0 \le d^T \left( \nabla^2 f(x^*) + \sum_{i=i^m} \lambda_i^* \nabla^2 c_i(x^*) \right) d + \alpha \|d\|^2,$$
for all stationary directions $d \in \mathcal{C}$.
- But $\alpha > 0$ is arbitrary so I can drop the $+\alpha\|d\|^2$ term (why?).

And I finally have
$$0 \le d^T \mathcal{L}(x^*, \lambda^*) d$$
for all stationary directions $d \in \mathcal{C}$ as required. ■

### 8.3.1 Weaker Necessary Conditions

It is worth noting that a weaker version of the second order KKT necessary conditions for an inequality constrained problem is easily proved, namely:

**Theorem 8.6 (Second-Order Weak Necessary Conditions)** *Suppose that $x^*$ is a local solution of (8.1) and that the LICQ (Def. 8.1) constraint qualification is satisfied. Let $\lambda^*$ be a Lagrange multiplier vector such that the KKT conditions (8.23a–8.20e) are satisfied, and let $\mathcal{C}(\lambda^*)$ be defined as above. Then*

$$\begin{aligned} d^T \nabla_{xx} \mathcal{L}(x^*, \lambda^*) d \geq 0, \quad & \textit{for all directions } d \textit{ s.t.} \\ & \nabla c_i(x^*)^T d = 0, \textit{for all } i \in \mathcal{A}(x^*). \end{aligned} \tag{8.38}$$

(Note that $\mathcal{A}(x^*) \equiv \mathcal{E} \cup (\mathcal{I} \cap \mathcal{A}(x^*))$.)

**Proof:**

Just as I proved the inequality-constrained version Thm. 8.1 of the first-order necessary conditions based on the equality-constrained version Thm. 8.2, I can use the same trick here, now based on the proof for the equality-constrained case Thm. 8.5.

I use the same observation as in the proof of Thm. 8.1:

- Inequality constraints that are **inactive** at the optimal point $x^*$ may be ignored as they cannot affect the solution.
- The inequality constraints that are **active** at $x^*$ may be treated as equality constraints in the sense that if $x^*$ is a solution to the problem (8.1) then it is also a solution to the **equality-constrained** problem

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{subject to}\, c_i(x) = 0, i \in \mathcal{A}(x^*). \tag{8.39}$$

- So applying Thm. 8.5.

$$0 \leq y^\mathsf{T} \mathcal{L}(x^*, \lambda^*) y$$

for all directions $y$ satisfying $y^\mathsf{T} \nabla c_i(x^*) = 0$, $i \in \mathcal{A}(x^*)$ as required.

$\blacksquare$

## Comments on Weak Necessary Conditions

- The difference between (8.38) and (8.35) is in the treatment of the weakly active constraints — those for which $c_i(x^*) = 0$ and $\lambda_i^* = 0$.

- The conditions are **weak** in the sense that the set of vectors $d$ satisfying

$$\begin{cases} \nabla c_i(x^*)^T d = 0, & \text{for all } i \in \mathcal{E}, \\ \nabla c_i(x^*)^T d = 0, & \text{for all } i \in \mathcal{A}(x^*)\mathcal{I}. \end{cases}$$

  is a proper subset of $\mathcal{C}(x^*, \lambda^*)$ so the weak necessary conditions may be satisfied at a point (which is not a local minimum) while the stronger version of the second-order necessary conditions may not.

I illustrate this with an example.

**Example 8.7** *(I will work in $\mathbb{R}^3$so will use the familiar $(x, y, z)$ notation and also use $x$ to refer to the vector $(x, y, z)$ where convenient.)*

*Let $f(x) = z - (x^2 - y^2)$ so that the level surfaces $f(x) = c$ take the form $z = x^2 - y^2 + c$.*

*I impose two inequality constraints:*

$$\begin{aligned} c_1(x) &= 1 - x^2 - y^2 - (z-1)^2 \geq 0 \\ c_2(x) &= ax + by + cz \geq 0. \end{aligned}$$

*The constraint $c_1(x) \geq 0$ corresponds to the interior of a unit sphere centred at $(0, 0, 1)$ and $c_2(x) \geq 0$ to a plane through the origin with normal vector $(a, b, c)^T$. At the point $x^* = (0, 0, 0)^T$, $f(x^*) = 0$ and both constraints are active. However $x^*$ is not a local minimum as I will show.*

*First I check the first order KKT conditions at* $x^*$ *:*

$$\nabla \mathcal{L}(x^*, \lambda) = \begin{bmatrix} -2x \\ 2y \\ 1 \end{bmatrix}_{(0,0,0)} - \lambda_1 \begin{bmatrix} -2x \\ -2y \\ -2(z-1) \end{bmatrix}_{(0,0,0)} - \lambda_2 \begin{bmatrix} a \\ b \\ c \end{bmatrix} = 0$$

$$= \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} - \lambda_1 \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix} - \lambda_2 \begin{bmatrix} a \\ b \\ c \end{bmatrix} = 0$$

*So the first-order KKT necessary conditions are satisfied at* $x^*$ *if* $\lambda_1 = \frac{1}{2}$ *and* $\lambda_2 = 0$.

*Now consider the second-order necessary conditions. The critical directions* $\mathcal{C}(x^*, \lambda^*)$ *are the directions* $d$ *s.t.* $d^T \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix} = 0$ *and*

$d^T \begin{bmatrix} a \\ b \\ c \end{bmatrix} \geq 0$ *so I must have* $d = \begin{bmatrix} \alpha \\ \beta \\ 0 \end{bmatrix}$ *with* $\alpha a + \beta b \geq 0$.

*The Hessian of the Lagrangian* $\nabla^2 \mathcal{L}(x^*, \lambda^*) = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

*The second-order necessary condition is that* $d^T\nabla^2\mathcal{L}(x^*,\lambda^*)d \geq 0$ *for all* $d \in \mathcal{C}$. *With* $d = \begin{bmatrix}\alpha\\ \beta\\ 0\end{bmatrix}$ *the conditions to be satisfied by* $\alpha, \beta$ *are* $d^T\nabla^2\mathcal{L}(x^*,\lambda^*)d = -\alpha^2 + 3\beta^2 \geq 0$ *together with* $\alpha a + \beta b \geq 0$.

*But I can easily construct* $d$ *s.t.*$-\alpha^2 + 3\beta^2 < 0$ *or equivalently* $3\beta^2 < \alpha^2$ *while still satisfying* $\alpha a + \beta b \geq 0$. *Just choose* $\text{sign}(\alpha) = \text{sign}(a)$ *and* $\text{sign}(\beta) = \text{sign}(b)$ *and (for example)* $|\alpha| = 2|\beta|$.

*So the condition* $d^T\nabla^2\mathcal{L}(x^*,\lambda^*)d \geq 0$ *is* **not** *true for all* $d \in \mathcal{C}$ *and therefore* $x^*$ *is* **not** *a local minimum point.*

*But…*

*The weak necessary conditions of Thm. 8.6* **are** *satisfied!*

*The vector* $d$ *must satisfy*

$d^T \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix} = 0$ *and* $d^T \begin{bmatrix} a \\ b \\ c \end{bmatrix} = 0$ *so I must have* $d = \begin{bmatrix} \alpha \\ \beta \\ 0 \end{bmatrix}$ *with* $\alpha a + \beta b = 0$.

*The latter condition is satisfied by any* $\alpha, \beta$ *s.t.* $\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = K \begin{bmatrix} b \\ -a \end{bmatrix}$ *for any* $K \in \mathbb{R}$.

*As before I have* $d^T \nabla^2 \mathcal{L}(x^*, \lambda^*) d = -\alpha^2 + 3\beta^2$. *Substituting for* $\alpha, \beta$ *I have* $-\alpha^2 + 3\beta^2 = K^2 \left(-b^2 + 3a^2\right)$ *which is non-negative for all choices of* $a, b$ *satisfying* $3a^2 \geq b^2$.

*So for certain choices of the weakly active constraint (namely* $3a^2 \geq b^2$*) the weak version of the necessary conditions is satisfied at* $x^*$ *, even though it is not in fact a local minimum.*

- The weak version of the necessary condition is weak precisely because it is easier to “pass the test” — as the example illustrates.

- The set of vectors ($\mathcal{D}$, say) $\mathbf{d}$ that are orthogonal to the gradients of the active constraints (the set specified in the weak necessary conditions of Thm. 8.6) is a subset of $\mathcal{C}$ so $\mathbf{d}^T \nabla^2 \mathcal{L}(x^*, \lambda^*) \mathbf{d}$ may be non-negative for $\mathbf{d} \in \mathcal{D}$ but not for all $\mathbf{d} \in \mathcal{C}$.

- A final comment: weakly active constraints ($c_i(x^*) = 0$ and $\lambda_i^* = 0$) may be ignored when examining the first-order KKT necessary conditions but — as I have just seen — this is not the case for the second-order necessary conditions.

**Sufficient conditions** are conditions on $f$ and $c_i, i \in \mathcal{E} \cup \mathcal{I}$, that **ensure** that $x^*$ is a local solution of the problem (8.1). The second-order sufficient condition stated in the next theorem looks very much like the necessary condition just discussed, but it differs in that the LICQ constraint qualification is not required, and the inequality in (8.35) is replaced by a strict inequality.

**Theorem 8.7 (Second-Order Sufficient Conditions)** *Suppose that for some feasible point $x^* \in \mathbb{R}^n$ there is a Lagrange multiplier vector $\lambda^*$ such that the KKT conditions (8.23a–8.20e) are satisfied. Suppose also that*

$$d^T \nabla_{xx} \mathcal{L}(x^*, \lambda) d > 0, \qquad \text{for all stationary directions } d \in \mathcal{C}(\lambda^*), d \neq 0. \tag{8.40}$$

*Then $x^*$ is a strict local solution for (8.1).*

**Proof:** See App. A.24.

**Example 8.8 (Example 8.2 One Last Time)** *I now return to Example 8.2 to check the second-order conditions for problem* (8.11)*. In this problem I have* $f(x) = x_1 + x_2$, $c_1(x) = 2 - x_1^2 - x_2^2$, $\mathcal{E} = \emptyset$, *and* $\mathcal{I} = \{1\}$*. The Lagrangian is*

$$\mathcal{L}(x, \lambda) = (x_1 + x_2) - \lambda_1 (2 - x_1^2 - x_2^2),$$

*and I saw that the KKT conditions (8.23a–8.20e) are satisfied by* $x^* = (-1, -1)^T$, *with* $\lambda_1^* = \frac{1}{2}$*. The Lagrangian Hessian at this point is*

$$\nabla_{xx} \mathcal{L}(x^*, \lambda^*) = \begin{bmatrix} 2\lambda_1^* & 0 \\ 0 & 2\lambda_1^* \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

*This matrix is positive definite, that is, it satisfies* $w^T \nabla_{xx} \mathcal{L}(x^*, \lambda^*) w > 0$ *for all* $w \neq 0$, *so it certainly satisfies the conditions of Theorem 8.7. I conclude that* $x^* = (-1, -1)^T$ *is a strict local solution for* (8.11)*.*

**Example 8.9** *For an example in which the issues are more complex, consider the problem*

$$\min x_2^2 - \frac{1}{10}(x_1 - 4)^2 \quad s.t. \quad x_1^2 + x_2^2 - 1 \geq 0, \tag{8.41}$$

*in which I seek to minimise a nonconvex function over the* **exterior** *of the unit circle. Obviously, the objective function is not bounded below on the feasible region, since I can take the sequence of feasible points* $(10, 0)$, $(20, 0)$, $(30, 0)$, $\dots$ *and note that* $f(x)$ *approaches* $-\infty$ *along this sequence. Therefore, no global solution exists, but it may still be possible to identify a strict local solution on the boundary of the constraint. I search for such a solution by using the KKT conditions (8.23a–8.20e) and the second-order sufficient conditions of Theorem 8.7.*

*By defining the Lagrangian for (8.41) in the usual way, it is easy to verify that*

$$\nabla_x \mathcal{L}(x, \lambda) = \begin{bmatrix} -0.2(x_1 - 4) - 2\lambda x_1 \\ 2x_2 - 2\lambda x_2 \end{bmatrix}, \tag{8.42}$$

$$\nabla_{xx} \mathcal{L}(x, \lambda) = \begin{bmatrix} -0.2 - 2\lambda & 0 \\ 0 & 2 - 2\lambda \end{bmatrix}. \tag{8.43}$$

*The point $x^* = (1, 0)^T$ satisfies the KKT conditions with $\lambda_1^* = 0.3$ and the active set $\mathcal{A}(x^*) = \{1\}$.*

*To check that the second-order sufficient conditions are satisfied at this point,*

*I note that* $\nabla c_1(x^*) = \begin{bmatrix} 2 \\ 0 \end{bmatrix}$ *so that the space* $\mathcal{C}$ *defined in* (8.32) *is simply* $\mathcal{C}(\lambda^*) = \{w | w_1 = 0\} = \{(0, w_2)^T | w_2 \in \mathbb{R}\}$.

*Now, by substituting* $x^*$ *and* $\lambda^*$ *into* (8.43) *we have for any* $w \in \mathcal{C}$ *with* $w \neq 0$ *that*

$$w^T \nabla_{xx} \mathcal{L}(x^*, \lambda^*) w = \begin{bmatrix} 0 & w_2 \end{bmatrix} \begin{bmatrix} -0.4 & 0 \\ 0 & 1.4 \end{bmatrix} \begin{bmatrix} 0 \\ w_2 \end{bmatrix} = 1.4 w_2^2 > 0.$$

*Hence, the second-order sufficient conditions are satisfied, and I conclude from Theorem 8.7. that* $(1, 0)$ *is a strict local solution for* (8.41)*.*

## 8.4 Exercises

1. Prove that the direction $\mathbf{d}$ given in (8.8) satisfies both conditions (8.6) and (8.7)
2. Prove that the direction $\mathbf{d}$ given in (8.14) satisfies both (8.7) and (8.13).
3. Does problem (8.28) have a finite or infinite number of local solutions? Use the first-order optimality conditions (8.23a–8.20e) to justify your answer.
4. Sketch the feasible region defined by the constraint set:

$$x_2 \leq x_1^3, \quad x_2 \geq 0.$$

Check the LICQ (Def. 8.1) doesn't hold at $x^* = (0, 0)$.

5. Show that for the feasible region defined by

$$\begin{aligned} (x_1 - 1)^2 + (x_2 - 1)^2 &\leq 2, \\ (x_1 - 1)^2 + (x_2 + 1)^2 &\leq 2, \\ x_1 &\geq 0, \end{aligned}$$

the MFCQ is satisfied at $x^* = (0, 0)$ but the LICQ is not satisfied.

6. Prove that when the KKT conditions (8.23a–8.20e) and the LICQ are satisfied at a point $x^*$, the Lagrange multiplier $\lambda^*$ in the KKT condition is unique.

7. Consider the problem

$$\min_{x \in R^2} f(x) = -2x_1 + x_2 \quad \text{subject to} \begin{cases} (1 - x_1)^3 - x_2 & \geq 0 \\ x_2 + 0.25x_1^2 - 1 & \geq 0. \end{cases} \tag{8.44}$$

The optimal solution is $x^* = (0, 1)^T$, where both constraints are active. Does the LICQ hold at this point? Are the KKT conditions satisfied?

8. Does the problem 8.41 have any other KKT points (other than $(1, 0)^T$? Do these points (if any) satisfy the second-order necessary or sufficient conditions?

# 9 Linear Programs

In this short Chapter I look at a familiar (from O.R. 1) problem — solving Linear Programs (LP's). They are very important in their own right and also allow me to exercise the KKT necessary conditions on a problem of significant complexity.

(This Chapter is based almost entirely on the first few Sections of Ch.13 of Nocedal & Wright.)

## 9.1 LP Definitions

- Linear programs have a linear objective function and linear constraints, which may include both equalities and inequalities.
- The feasible set is a **polytope**, a convex, connected set with flat, polygonal faces.

- The level surfaces (contours in $\mathbb{R}^2$) of the objective function are planes $c^T x = K$ (more correctly referred to as affine spaces as they do not in general pass through the origin).
- In two dimensions it is easy to use graphical methods — the optimal point will be either a vertex, an edge or perhaps the problem is unbounded or infeasible.
- You should draw some rough sketches to illustrate all these possible cases.
- In higher dimensions, the set of optimal points can be a single vertex, an edge or face, or even the entire feasible set.
- The problem has no solution if the feasible set is empty (the infeasible case) or if the objective function is unbounded below on the feasible region (the unbounded case).

- Linear programs are usually stated and analyzed in the following standard form:

$$\min c^T x, \text{ subject to } Ax = b,\ x \geq 0. \tag{9.1}$$

where $c$ and $x$ are vectors in $\mathbb{R}^n$, $b$ is a vector in $\mathbb{R}^m$, and $A$ is an $m \times n$ matrix with $m \leq n$.

- It is easy to transform any linear program to this form.
- For instance, given the problem

$$\min c^T x, \text{subject to} \, Ax \leq b$$

(without any bounds on $x$), I can convert the inequality constraints to equalities by introducing a vector of slack variables $z$ and writing

$$\min c^T x, \text{ subject to } Ax + z = b,\ z \geq 0.$$

- This form is still not quite standard, since not all the variables are constrained to be nonnegative.
- I deal with this by splitting $x$ into its nonnegative and nonpositive parts, $x = x^{+} - x^{-}$ , where $x^{+} = \max(x, 0) \geq 0$ and $x^{-} = \max(-x, 0) \geq 0$.
- The problem (9.1) can now be written as

$$\min \begin{bmatrix} c \\ -c \\ 0 \end{bmatrix}^{T} \begin{bmatrix} x^{+} \\ x^{-} \\ z \end{bmatrix} \text{ s.t. } \begin{bmatrix} A & -A & I \end{bmatrix} \begin{bmatrix} x^{+} \\ x^{-} \\ z \end{bmatrix} = b \ , \begin{bmatrix} x^{+} \\ x^{-} \\ z \end{bmatrix} \geq 0. \tag{9.2}$$

which clearly has the same form as (9.1).

- Inequality constraints of the form $Ax \leq b$ or $Ax \geq b$ can always be converted to equality constraints by adding or subtracting slack variables to make up the difference between the left- and right-hand sides.
- So $Ax \leq b \Leftrightarrow Ax + s = b, s \geq 0$ and $Ax \geq b \Leftrightarrow Ax - s = b, s \geq 0$.
- When I subtract the variables from the left hand side, as in the second case, they are sometimes known as surplus variables.
- I can also convert a “maximize” objective $\max c^T x$ into the “minimize” form of (9.1) by simply negating $c$.

- I say that the linear program (9.1) is infeasible if the feasible set is empty.
- I say that the problem (9.1) is unbounded if the objective function is unbounded below on the feasible region, that is, there is a sequence of points $x_k$ feasible for (9.1) such that $c^T x_k \downarrow -\infty$.
- Of course, unbounded problems have no solution.
- For the standard formulation (9.1), I will assume throughout that $m < n$.
- Otherwise, the system $Ax = b$ contains redundant rows, or is infeasible, or defines a unique point.
- When $m \geq n$, factorisations such as the QR or LU factorisation can be used to transform the system $Ax = b$ to one with a coefficient matrix of full row rank.

## 9.2 Optimality Conditions

- Optimality conditions for the standard form problem (9.1) can be derived from the theory of Chapter 8.
- Only the (KKT) Karush Kuhn Tucker first order necessary conditions are needed as the problem is linear.
- Convexity of the problem ensures that these conditions are sufficient for a global minimum.
- I do not need to refer to the second-order conditions from Chapter 8, which are "empty" as the Hessian of the Lagrangian for (9.1) is zero.
- The theory that I developed in Chapter 8 makes derivation of optimality and duality results for linear programming much easier than in other treatments, where this theory is developed more or less from scratch.

- The KKT conditions were stated in Theorem 8.1.
- As stated in Chapter 8, this theorem requires linear independence of the active constraint gradients (LICQ).
- However, the Theorem continues to hold for dependent constraints provided they are linear, as is the case here.
- I partition the Lagrange multipliers for the problem (9.1) into two vectors $\lambda$ and $s$, where $\lambda \in \mathbb{R}^m$ is the multiplier vector for the $m$ equality constraints $Ax = b$, while $s \in \mathbb{R}^n$ is the multiplier vector for the bound (inequality) constraints $x \geq 0$.
- Using the definition (8.9) I can write the Lagrangian function for (9.1) as

$$\mathcal{L}(x, \lambda, s) = c^T x - \lambda^T (Ax - b) - s^T x. \tag{9.3}$$

- Applying the KKT conditions to (9.3), the first-order necessary conditions that $x^*$ be a solution to problem (9.1) are that there exist $\lambda$ and $s$ such that:

$$
\begin{aligned}
A^T\lambda + s &= c && (9.4a)\\
Ax &= b && (9.4b)\\
x &\geq 0 && (9.4c)\\
s &\geq 0 && (9.4d)\\
x_i s_i &= 0,\ i = 1, \ldots, n. && (9.4e)
\end{aligned}
$$

- The complementarity condition (9.4e) can also be written $x^T s = 0$ as $x$ and $s$ are non-negative.
- Let $x^*, \lambda^*, s^*$ satisfy Eqs. 9.4a–9.4e.
- Then

$$c^T x^* = (A^T \lambda^* + s^*)^T x^* = (Ax^*)^T \lambda^* = b^T \lambda^*. \tag{9.5}$$

- As I'll show in a moment, $b^T\lambda$ is the objective function for the **dual problem** to (9.1) so (9.5) indicates that the primal and dual objectives are equal for vector triples $(x, \lambda, s)$ that satisfy Eqs. 9.4a–9.4e.

- It is easy (though unnecessary as noted above) to show directly that the KKT conditions Eqs. 9.4a–9.4e are **sufficient** for $x^*$ to be a global solution of (9.1).
- Let $\bar{x}$ be any other feasible point, so that $A\bar{x} = b$ and $\bar{x} \geq 0$.
- Then

$$c^T\bar{x} = (A^T\lambda^* + s^*)^T\bar{x} = b^T\lambda^* + \bar{x}^Ts^* \geq b^T\lambda^* = c^Tx^*. \qquad (9.6)$$

- So no other feasible point can have a lower objective value than $c^Tx^*$.
- I can say more:
  - The feasible point $\bar{x}$ is optimal if and only if $\bar{x}^Ts^* = 0$ since otherwise the inequality in (9.6) is strict.
  - In other words, when $s_i^* > 0$, then I must have $\bar{x}_i = 0$ for all solutions $\bar{x}$ of (9.1).

## 9.3 The Dual Problem

- Given the data $c, b$ and $A$, which defines the problem (9.1) , I can define another, closely related, problem as follows:

$$\max b^T\lambda, \text{ subject to } A^T\lambda \le c. \tag{9.7}$$

- This problem is called the **dual problem** for (9.1).
- The original problem (9.1) is often referred to as the **primal**.
- I can restate (9.7) in a slightly different form by introducing a vector of dual slack variables $s$ and writing

$$\max b^T\lambda, \text{ subject to } A^T\lambda + s = c, s \ge 0. \tag{9.8}$$

- The variables $(\lambda, s)$ in this problem are sometimes jointly referred to collectively as **dual variables**.

- The primal and dual problems present two different viewpoints on the same data.
- Their close relationship becomes evident when I write down the KKT conditions for (9.7).
- First let's restate (9.7) in the form

  $$\min -b^T\lambda \quad \text{s.t.} \;\; c - A^T\lambda \geq 0,$$

  to fit the formulation (8.1) in Ch 8.
- By using $x$ for the Lagrange multipliers for the constraints $A^T\lambda \leq c$, I see that the Lagrangian function is $\bar{\mathcal{L}}(\lambda, x) = -b^T\lambda - x^T(c - A^T\lambda)$.

- Using Theorem 8.1 again, I find the first-order necessary conditions for $\lambda$ to be optimal for (9.7) is that there exists $x$ such that

$$Ax = b \tag{9.9a}$$

$$A^T\lambda \leq c \tag{9.9b}$$

$$x \geq 0 \tag{9.9c}$$

$$x_i(c - A^T\lambda)_i = 0, \quad i = 1, \dots, n. \tag{9.9d}$$

- Defining $s = c - A^T\lambda$ (as in (9.8)), I find that the conditions Eqs. 9.4a–9.4e and Eqs. 9.9a–9.9d are identical!
- The optimal Lagrange multipliers $\lambda$ in the primal problem are the optimal variables in the dual problem, while the optimal Lagrange multipliers $x$ in the dual problem are the optimal variables in the primal problem.

- Analogously to (9.6), I can show that Eqs. 9.9a–9.9d are in fact sufficient conditions for a solution of the dual problem (13.7).
- Given $x^*$ and $\lambda^*$ satisfying these conditions (so that the triple $(x, \lambda, s) = (x^*, \lambda^*, c - A^T\lambda^*)$ satisfies Eqs. 9.4a–9.4e, I have for any other dual feasible point $\bar{\lambda}$ (with $A^T\bar{\lambda} \le c$) that

$$\begin{aligned} b^T\bar{\lambda} &= x^{*T}A^T\bar{\lambda} \\ &= x^{*T}(A^T\bar{\lambda} - c) + c^Tx^* \\ &\le c^Tx^* \quad \text{as } A^T\bar{\lambda} - c \le 0 \text{ and } x^* \ge 0. \\ &= b^T\lambda^* \quad \text{from } 9.5 \end{aligned}$$

- Hence $\lambda^*$ achieves the maximum of the dual objective $b^T\lambda$ over the dual feasible region $A^T\lambda \le c$, so it solves the dual problem (9.7).

- The primal-dual relationship is symmetric; by taking the dual of the dual problem (9.7), I recover the primal problem (9.1) — an interesting Exercise.

- Given a feasible vector $x$ for the primal (satisfying $Ax = b$ and $x \geq 0$) and a feasible point $(\lambda, s)$ for the dual (satisfying $A^T\lambda + s = c, s \geq 0$), I have as in (9.6) that

$$c^T x - b^T \lambda = (c - A^T\lambda)^T x = s^T x \geq 0. \tag{9.10}$$

- Therefore I have $c^T x \geq b^T \lambda$ (that is, the dual objective is a lower bound on the primal objective) when both the primal and dual variables are feasible—a result known as weak duality.

The following strong duality result is fundamental to the theory of linear programming.

**Theorem 9.1 (Strong Duality)** *I have a "dichotomy":*

*(i) If either the primal* (9.1) *or dual* (9.7) *problem has a finite solution then so does the other and the objective values are equal.*

*(ii) If either problem is unbounded then the other problem is infeasible.*

**Proof:**

- For (i), suppose that (9.1) has a finite optimal solution $x^*$ .
- It follows from Theorem 8.1 that there are vectors $\lambda^*$ and $s^*$ such that $(x^*, \lambda^*, s^*)$ satisfies (9.4a)–(9.4e)
- I noted above that (9.4a)–(9.4e) and (9.9a)–(9.9d) are equivalent, and that (9.9a)–(9.9d) are sufficient conditions for $\lambda^*$ to be a solution of the dual problem (9.7).
- Moreover, it follows from (9.6) that $c^T x^* = b^T \lambda$ , as claimed.
- A symmetric argument holds if I start by assuming that the dual problem (9.7) has a solution.

- To prove (ii), suppose that the primal is unbounded, that is, there is a sequence of points $x_k, k = 1, 2, 3, \ldots$ such that $c^T x_k \downarrow -\infty$, $Ax_k = b$ and $x_k \geq 0$.
- Suppose too that the dual (9.7) is feasible, that is, there exists a vector $\bar{\lambda}$ such that $A^T \bar{\lambda} \leq c$.
- From the latter inequality together with $x_k \geq 0$, I have that $\bar{\lambda}^T A x_k \leq c^T x^k$ and therefore $\bar{\lambda}^T b = \bar{\lambda}^T A x_k \leq c^T x_k \downarrow -\infty$.
- So I have a contradiction.
- Hence, the dual must be infeasible.
- A similar argument can be used to show that unboundedness of the dual implies infeasibility of the primal.

$\blacksquare$

## 9.4 Geometry Of The Feasible Set

From now on I assume that the matrix $A$ has full row rank.

In practice, a preprocessing phase is applied to the user-supplied data to remove some redundancies from the given constraints and eliminate some of the variables. Reformulation by adding slack, surplus, and artificial variables can also result in A having full row rank.

**Definition 9.1 (Basic Feasible Point)** *A vector $x$ is a basic feasible point if it is feasible and if there exists a subset $\mathcal{B}$ of the index set $\{1, 2, \dots, n\}$ such that*

- *$\mathcal{B}$ contains exactly $m$ indices ($m < n$).*
- *$i \notin \mathcal{B} \Rightarrow x_i = 0$ (i.e. the bound $x_i \geq 0$ is inactive only if $i \in \mathcal{B}$).*
- *The $m \times m$ matrix $B$ defined by $B = [A_i]_{i \in \mathcal{B}}$ is non-singular ($A_i$ is the $i^{th}$ column of $A$).*

- Each iterate generated by the Simplex method is a basic feasible point of (9.1).
- A set $\mathcal{B}$ satisfying these properties is called a basis for the problem (9.1).
- The corresponding matrix B is called the basis matrix.
- The Simplex method's strategy of examining only basic feasible points will converge to a solution of (9.1) only if

  (a) the problem has basic feasible points and

  (b) at least one such point is a basic optimal point, that is, a solution of (9.1) that is also a basic feasible point.
- Happily, both (a) and (b) are true under reasonable assumptions, as the following result (sometimes known as the fundamental theorem of linear programming) shows.

**Theorem 9.2** *The following cases arise:*

*(i)* *If* (9.1) *has a nonempty feasible region, then there is at least one basic feasible point;*

*(ii)* *If* (9.1) *has solutions, then at least one such solution is a basic optimal point.*

*(iii)* *If* (9.1) *is feasible and bounded, then it has an optimal solution.*

**Proof:**

(i)

- Among all feasible vectors $x$, choose one with the minimal number of nonzero components and denote this number by $p$.
- Without loss of generality, assume that the nonzeros are $x_1, x_2, \ldots, x_p$ , so I have $\sum_{i=1}^{p} A_i x_i = b$.
- Suppose first that the columns $A_1, A_2, \ldots, A_p$ are linearly dependent.
- Then I can express one of them ( $A_p$ , say) in terms of the others, and write $A_p = \sum_{i=1}^{p-1} z_i A_i$ for some scalars $z_1, z_2, \ldots, z_{p-1}$ .
- It is easy to check that the vector $x(\epsilon) = x + \epsilon(z_1, z_2, \ldots, z_{p-1}, -1, 0, 0, \ldots, 0)^T = x + \epsilon z$ satisfies $Ax(\epsilon) = b$ for any scalar $\epsilon$.

- In addition, since $x_i > 0$ for $i = 1, 2, \dots, p$, I also have $x_i(\epsilon) > 0$ for the same indices $i = 1, 2, \dots, p$ and all $\epsilon$ sufficiently small in magnitude.
- However, there is a value $\bar{\epsilon} \in (0, x_p]$ such that $x_i(\bar{\epsilon}) = 0$ for some $i = 1, 2, \dots, p$.
- Hence, $x(\bar{\epsilon})$ is feasible and has at most $p - 1$ nonzero components, contradicting our choice of $p$ as the minimal number of nonzeros.
- Therefore, columns $A_1, A_2, \dots, A_p$ must be linearly independent, and so $p \leq m$.

- If $p = m$, I am done, since then $x$ is a basic feasible point and $\mathcal{B}$ is simply $\{1, 2, \ldots, m\}$.
- Otherwise $p < m$ and because $A$ has full row rank, I can choose $m - p$ columns from among $A_{p+1}, A_{p+2}, \ldots, A_n$ to build up a set of $m$ linearly independent vectors.
- I construct $\mathcal{B}$ by adding the corresponding indices to $\{1, 2, \ldots, p\}$.
- (The vector $x$ still has only $p$ non-zero components.)
- So $x$ is a basic feasible point.
- The proof of (i) is complete.

(ii) The proof of (ii) is quite similar.

- Let $x^*$ be a solution with a minimal number of nonzero components $p$, and assume again that $x_1^*, x_2^*, \dots, x_p^*$ are the nonzeros.
- If the $p$ columns $A_1, A_2, \dots, A_p$ are linearly dependent, we define $x^*(\epsilon) = x^* + \epsilon z$, where $z$ is chosen exactly as above.
- It is easy to check that $x^*(\epsilon)$ will be feasible for all $\epsilon$ sufficiently small, both positive and negative.
- Hence, since $x^*$ is optimal, I must have $c^T(x^* + \epsilon z) \geq c^T x^* \Rightarrow \epsilon c^T z \geq 0$ for all $\epsilon$ sufficiently small (positive and negative).
- Therefore, $c^T z = 0$ and so $c^T x^*(\epsilon) = c^T x^*$ for all $\epsilon$.
- The same logic as in the proof of (i) can be applied to find $\bar{\epsilon} > 0$ such that $x^*(\bar{\epsilon})$ is feasible and optimal, with at most $p - 1$ nonzero components.

- This contradicts our choice of $p$ as the minimal number of nonzeros, so the columns $A_1, A_2, \ldots, A_p$ must be linearly independent.
- I can now apply the same reasoning as above to conclude that $x^*$ is already a basic feasible point and therefore a basic optimal point.

(iii) The final statement (iii) is a consequence of finite termination of the Simplex method. I comment on the latter property in the next section. ■

### 9.4.1 Vertices Of the Feasible Polytope

- The feasible set defined by the linear constraints is a **polytope** — a region bounded by affine spaces "planes" $a_i^T x = b$.
- The **vertices** of this polytope are the points that do not lie on a straight line between two other points in the set — a geometric definition.
- Basic feasible points were defined above algebraically.
- I will show that **vertices** $\equiv$ **basic feasible points**.
- I therefore have an important relationship between the algebraic and geometric viewpoints and a useful aid to understanding how the Simplex method works.

**Theorem 9.3** *All basic feasible points for (9.1) are vertices of the feasible polytope* $\{x|Ax = b, x \geq 0\}$*, and vice versa.*

**Proof:**

1. Let $x$ be a basic feasible point and assume without loss of generality that $\mathcal{B} = \{1, 2, \ldots, m\}$.
   - The matrix $B = [A_i]_{i=1,2,\ldots,m}$ is therefore nonsingular and
     $$x_{m+1} = x_{m+2} = \ldots x_n = 0. \tag{9.11}$$
   - Suppose that $x$ lies on a straight line between two other distinct feasible points $y$ and $z$ — i.e. that $x$ is not a vertex.
   - Then I can find $\alpha \in (0, 1)$ such that $x = \alpha y + (1 - \alpha)z$.
   - Because of (9.11) and the fact that $\alpha$ and $1 - \alpha$ are both positive, I must have $y_i = z_i = 0$ for $i = m+1, m+2, \ldots, n$.

- Writing $x_B = (x_1, x_2, \ldots, x_m)^T$ and defining $y_B$ and $z_B$ likewise, I have from $Ax = Ay = Az = b$ that $Bx_B = By_B = Bz_B = b$ and so, by nonsingularity of B, we have $x_B = y_B = z_B$ .
- Therefore, $x = y = z$, contradicting our assertion that $y$ and $z$ are two distinct feasible points other than $x$.
- Therefore, $x$ is a vertex.

2. Conversely, let $x$ be a vertex of the feasible polytope, and suppose that the nonzero components of $x$ are $x_1, x_2, \ldots, x_p$.
   - If the corresponding columns $A_1, A_2, \ldots, A_p$ are linearly dependent, then I can construct the vector $x(\epsilon) = x + \epsilon z$ as in (13.15).
   - Since $x(\epsilon)$ is feasible for all $\epsilon$ with sufficiently small magnitude, I can define $\hat{\epsilon} > 0$ such that $x(\hat{\epsilon})$ and $x(-\hat{\epsilon})$ are both feasible.

- Since $x = x(0)$ obviously lies on a straight line between these two points, it cannot be a vertex.
- Hence our assertion that $A_1, A_2, \ldots, A_p$ are linearly dependent must be incorrect, so these columns must be linearly independent and $p \leq m$.
- If $p < m$ then since A has full row rank, I can add $m - p$ indices to $\{1, 2, \ldots, p\}$ to form a basis $\mathcal{B}$, for which $x$ is the corresponding basic feasible point.

■

I conclude this discussion of the geometry of the feasible set with a definition of degeneracy. For the purposes of this Chapter, I use the following definition.

**Definition 9.2 (Degeneracy)** *A basis $\mathcal{B}$ is said to be degenerate if $x_i = 0$ for some $i \in \mathcal{B}$, where $x$ is the basic feasible solution corresponding to $\mathcal{B}$. A linear program (9.1) is said to be degenerate if it has at least one degenerate basis.*

**Example 9.1** *Let $A = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 1 \end{bmatrix}$ and $b = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$. Then $\mathcal{B} = \{1, 2\}$ and $\mathcal{B} = \{1, 3\}$ are both bases corresponding to the basic feasible points $(0, 1, 0)$ and $(0, 0, 1)$. Can you see that both bases are degenerate? Draw a diagram!*

## 9.5 The Simplex Method

### 9.5.1 Introduction to the method

In this Section I give a detailed description of the Simplex method for (9.1).

- There are actually a number of variants of the Simplex method; the one described here is sometimes known as the Revised Simplex method.

  As I described above, all iterates of the Simplex method are basic feasible points for (9.1) and therefore vertices of the feasible polytope.

- Most steps consist of a move from one vertex to an adjacent one for which the basis $\mathcal{B}$ differs in exactly one component.

- On most steps (but not all), the value of the primal objective function $\mathsf{c}^\mathsf{T}\mathsf{x}$ is decreased.

- Another type of step occurs when the problem is unbounded: The step is an edge along which the objective function is reduced, and along which I can move infinitely far without ever reaching a vertex.

- The major issue at each Simplex iteration is to decide which index to remove from the basis $\mathcal{B}$.

- Unless the step is a direction of unboundedness, a single index must be removed from $\mathcal{B}$ and replaced by another from outside $\mathcal{B}$.

- I can gain some insight into how this decision is made by looking again at the KKT conditions (9.4a)–(9.4e).

- From the value of $\mathcal{B}$ and from (9.4a)–(9.4e), I can derive values for not just the primal variable $x$ but also the dual variables $(\lambda, s)$ as I now show.

- First, define the nonbasic index set $\mathcal{N}$ as the complement of $\mathcal{B}$, that is,
$$\mathcal{N} = \{1, 2, \dots, n\} \backslash \mathcal{B}. \tag{9.12}$$
- Just as $\mathcal{B}$ is the basic matrix, whose columns are $A_i$ for $i \in \mathcal{B}$, I use $N$ to denote the nonbasic matrix $N = [A_i]_{i \in \mathcal{N}}$ .
- I also partition the n-element vectors x, s, and c according to the index sets $\mathcal{B}$ and $\mathcal{N}$, using the notation
$$\begin{aligned} x_B &= [x_i]_{i \in \mathcal{B}}, & x_N &= [x_i]_{i \in \mathcal{N}} \\ s_B &= [s_i]_{i \in \mathcal{B}}, & s_N &= [s_i]_{i \in \mathcal{N}} \\ c_B &= [c_i]_{i \in \mathcal{B}}, & c_N &= [c_i]_{i \in \mathcal{N}} \end{aligned}$$
- From the KKT condition (9.4b), I have that
$$Ax = Bx_B + Nx_N = b.$$

- The primal variable $x$ for this Simplex iterate is defined as

$$x_B = B^{-1}b, \quad x_N = 0. \tag{9.13}$$

- Since I am dealing only with basic feasible points, I know that $B$ is nonsingular and that $x_B \geq 0$, so this choice of $x$ satisfies two of the KKT conditions: the equality constraints (9.4b) and the nonnegativity condition (9.4c).

- I choose $s$ to satisfy the complementarity condition (9.4e) by setting $s_B = 0$.

- The remaining variables $\lambda$ and $s_N$ can be found by partitioning the first KKT condition (9.4a) into $c_B$ and $c_N$ components and using $s_B = 0$ to obtain

$$B^T\lambda = c_B, \quad N^T\lambda + s_N = c_N \tag{9.14}$$

- Since B is square and nonsingular, the first equation uniquely defines $\lambda$ as
$$\lambda = B^{-T} c_B. \tag{9.15}$$
- The second equation in (9.14) implies a value for $s_N$ :
$$s_N = c_N - N^T \lambda = c_N - (B^{-1} N)^T c_B. \tag{9.16}$$
- Computation of the vector $s_N$ is often referred to as pricing.
- The components of $s_N$ are often called the reduced costs of the nonbasic variables $x_N$ .

- The only KKT condition that I have not enforced explicitly is the nonnegativity condition $s \geq 0$.
- The basic components $s_B$ certainly satisfy this condition, by our choice $s_B = 0$.
- If the vector $s_N$ defined by (9.16) also satisfies $s_N \geq 0$, I have found an optimal vector triple $(x, \lambda, s)$, so the algorithm can terminate and declare success.
- Usually, however, one or more of the components of $s_N$ are negative.

- The new index to enter the basis $\mathcal{B}$—the entering index—is chosen to be one of the indices $q \in \mathcal{N}$ for which $s_q < 0$.
- As I show below, the objective $c^T x$ will decrease when I allow $x_q$ to become positive if and only if
  (i) $s_q < 0$
  (ii) it is possible to increase $x_q$ away from zero while maintaining feasibility of x.

- The procedure for altering $\mathcal{B}$ and changing x and s can be described accordingly as follows:
  - allow $x_q$ to increase from zero during the next step;
  - fix all other components of $x_N$ at zero, and figure out the effect of increasing $x_q$ on the current basic vector $x_B$ , given that I want to stay feasible with respect to the equality constraints $Ax = b$
  - keep increasing $x_q$ until one of the components of $x_B$ ($x_p$ , say) is driven to zero, or determining that no such component exists (the unbounded case);
  - remove index p (known as the leaving index) from $\mathcal{B}$ and replace it with the entering index q.

- This process of selecting entering and leaving indices, and performing the algebraic operations necessary to keep track of the values of the variables $x$, $\lambda$ and $s$, is sometimes known as pivoting.
- I now formalise the pivoting procedure in algebraic terms.
- Since both the new iterate $x^+$ and the current iterate $x$ should satisfy $Ax = b$, and since $x_N = 0$ and $x_i^+ = 0$ for $i \in \mathcal{N} \backslash \{q\}$, I have $Ax^+ = Bx_B^+ + A_q x_q^+ = Bx_B = Ax$.
- By multiplying this expression by $B^{-1}$ and rearranging, I obtain
$$x_B^+ = x_B - B^{-1} A_q x_q^+. \tag{9.17}$$
- Geometrically speaking, (9.17) is usually a move along an edge of the feasible polytope that decreases $c^T x$.
- I continue to move along this edge until a new vertex is encountered.

- At this vertex, a new constraint $x_p \geq 0$ must have become active, that is, one of the components $x_p$ for some $p \in \mathcal{B}$, has decreased to zero.

- I then remove this index $p$ from the basis $\mathcal{B}$ and replace it by $q$.

- I now show how the step defined by (9.17) affects the value of $c^T x$. From (9.17), we have

$$c^T x^+ = c_B^T x_B^+ + c_q x_q^+ = c_B^T x_B - c_B^T B^{-1} A_q x_q^+ + c_q x_q^+. \quad (9.18)$$

- From (9.15) I have $c_B^T B^{-1} = \lambda^T$ , while from the second equation in (9.14), since $q \in \mathcal{N}$, I have $A_q^T \lambda = c_q - s_q$ .

- Therefore,

$$c_B^T B^{-1} A_q x_q^+ = \lambda^T A_q x_q^+ = (c_q - s_q) x_q^+,$$

so by substituting in (9.18) I obtain

$$c^T x^+ = c_B^T x_B - (c_q - s_q) x_q^+ + c_q x_q^+ = c^T x + s_q x_q^+. \qquad (9.19)$$

- Since $q$ was chosen so that $s_q < 0$, it follows that the step (9.17) produces a decrease in the primal objective function $c^T x$ whenever $x_q^+ > 0$.
- It is possible that I can increase $x_q$ to $+\infty$ without ever encountering a new vertex.
- In other words, the constraint $x_B^+ = x_B - B^{-1} A_q x_q^+ \geq 0$ may hold for all positive values of $x_q$ .
- This can happen if $B^{-1} A_q \leq 0$.

- When this happens, the linear program is unbounded; the Simplex method has identified a ray that lies entirely within the feasible polytope along which the objective $c^T x$ decreases to $-\infty$.

- If the basis $\mathcal{B}$ is nondegenerate (see Definition 9.2), then I am guaranteed that $x_q^+ > 0$, so I can be assured of a strict decrease in the objective function $c^T x$ at this step.

- If the problem (9.1) is nondegenerate, we can ensure a decrease in $c^T x$ at every step, and can therefore prove the following result concerning termination of the Simplex method.

**Theorem 9.4** *Provided that the linear program* (9.1) *is nondegenerate and bounded, the Simplex method terminates at a basic optimal point.*

**Proof:**

- The Simplex method cannot visit the same basic feasible point x at two different iterations, because it attains a strict decrease at each iteration.
- Since the number of possible bases $\mathcal{B}$ is finite (there are only a finite number of ways to choose a subset of m indices from 1, 2, . . . , n), and since each basis defines a single basic feasible point, there are only a finite number of basic feasible points.

- Hence, the number of iterations is finite.
- Moreover, since the method is always able to take a step away from a nonoptimal basic feasible point, and since the problem is not unbounded, the method must terminate at a basic optimal point. ■

This result gives me a proof of Theorem 9.2 (iii) in the case in which the linear program is nondegenerate.

The proof of finite termination is considerably more complex when nondegeneracy of (9.1) is not assumed.

### 9.5.2 A Single Step of the Simplex Method

I have covered most of the details of taking a single step of the Simplex method. Now I can formulate a step of the Simplex method in pseudo-code for clarity.

## Algorithm 9.1

```
Given $\mathcal{B}, \mathcal{N}, x_B = B^{-1}b \geq 0, x_N = 0$;
Solve $B^T\lambda = c_B$ for $\lambda$
$s_N = c_N - N^T\lambda$;  ( pricing )
if $s_N \geq 0$
   then STOP; ( optimal point found )
fi
Select $q \in \mathcal{N}$ with $s_q < 0$ as the entering index
 Solve $Bd = A_q$ for d;
if $d \leq 0$
   then STOP; ( problem is unbounded )
fi
Calculate $x_q^+ = \min_{i \in \mathcal{B} | d_i > 0} (x_B)_i / d_i$ and use $p$ to denote the minimizing $i$;
Update $x_B^+ = x_B - dx_q^+$, $x_N^+ = (0, \ldots, 0, x_q^+, 0, \ldots, 0)^T$ ;
 Change $\mathcal{B}$ by adding $q$ and removing the basic variable $j_p$
 corresponding to column $p$ of B.
(The $p^{th}$ column of B is the $j_p^{th}$ column of $A$.)
```

Finally, an example!

**Example 9.2** *Consider the problem*

$$\begin{aligned} \min -3x_1 - 2x_2 \, & \textit{subject to} \\ x_1 + x_2 + x_3 \quad & = 5, \\ 2x_1 + (1/2)x_2 + \quad x_4 & = 8, \\ x & \geq 0. \end{aligned}$$

*Suppose I start with the basis* $\mathcal{B} = \{3, 4\}$*, for which I have*

$$x_B = \begin{bmatrix} x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 5 \\ 8 \end{bmatrix}, \lambda = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, s_N = \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} -3 \\ -2 \end{bmatrix}$$

*and an objective value of* $c^T x = 0$.

1. - *Since both elements of* $s_N$ *are negative, I could choose either* $1$ *or* $2$ *to be the entering variable.*
   - *Suppose I choose* $q = 1$*. Check that I obtain* $d = (1, 2)^T$ *, so I cannot (yet) conclude that the problem is unbounded.*
   - *By performing the ratio calculation, I find that* $p = 2$ *(corresponding to* $j_p = 4$*) and* $x_1^+ = 4$*.*
   - *I update the basic and nonbasic index sets to* $\mathcal{B} = \{3, 1\}$ *and* $\mathcal{N} = \{4, 2\}$*, and move to the next iteration.*

2. - *At the second iteration, I have*

$$x_B = \begin{bmatrix} x_3 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ 4 \end{bmatrix}, \lambda = \begin{bmatrix} 0 \\ -3/2 \end{bmatrix}, s_N = \begin{bmatrix} s_4 \\ s_2 \end{bmatrix} = \begin{bmatrix} 3/2 \\ -5/4 \end{bmatrix}$$

   *with an objective value of* $-12$*.*

   - *I see that* $s_N$ *has one negative component, corresponding to the index* $q = 2$ *so I select this index to enter the basis.*
   - *I obtain* $d = (3/4, 1/4)^T$ *, so again I do not detect unboundedness.*
   - *Continuing, I find that the maximum value of* $x_2^+$ *is* $4/3$*, and that* $p = 1$*, which indicates that index* $j_p = 3$ *will leave the basis* $\mathcal{B}$ *.*
   - *I update the index sets to* $\mathcal{B} = \{2, 1\}$ *and* $\mathcal{N} = \{4, 3\}$ *and continue.*

*3. At the start of the third iteration, I have*

$$x_B = \begin{bmatrix} x_2 \\ x_1 \end{bmatrix} = \begin{bmatrix} 4/3 \\ 11/3 \end{bmatrix}, \lambda = \begin{bmatrix} -5/3 \\ -2/3 \end{bmatrix}, s_N = \begin{bmatrix} s_4 \\ s_3 \end{bmatrix} = \begin{bmatrix} 2/3 \\ 5/3 \end{bmatrix}$$

*with an objective value of* $c^T x = -41/3$*. I see that* $s_N \geq 0$*, so the optimality test is satisfied, and I terminate.*

## 9.6 What Was Not Mentioned

- What to do if the “all-slack” solution is not feasible?
  - It is convenient to revert to the non-standard form $Ax \leq b, x \geq 0$.
  - Define an auxiliary problem: $\min x_0$ st $\begin{bmatrix} -\mathbf{1} & A \end{bmatrix} \begin{bmatrix} x_0 \\ x \end{bmatrix} \leq b$.
  - Define a real number $b_0 = 0$ if the vector $b \geq 0$ and $b_0 = -\min(b)$ otherwise so that $b_0 \geq 0$.
  - Clearly choosing $x_0 = b_0$ and $x = 0$ gives a feasible solution to our non-standard problem.
  - Introducing slacks $z$ gives $\begin{bmatrix} -\mathbf{1} & A & I \end{bmatrix} \begin{bmatrix} x_0 \\ x \\ z \end{bmatrix} = b$ is in standard form and $z = b - \min(b) \geq 0$ is feasible.

- The initial basis is just $\{1, j_1, \ldots, j_{m-1}\}$ where $j_1, \ldots, j_{m-1}$ are the columns of $\begin{bmatrix} -\mathbf{1} & A & I \end{bmatrix}$ corresponding to the positive elements of $z$.
- The matrix $\begin{bmatrix} -\mathbf{1} & I_{m,m-1} \end{bmatrix}$ ($I_{m,m-1}$ is the submatrix of I consisting of the columns corresponding to the positive elements of $z$) is certainly invertible (why?) — so I have a b.f.p. for the auxiliary problem.
- (What happens if more than one element of the vector $b$ is equal to $\min(b)$?)
- Now that I have a b.f.p. for the auxiliary problem, minimising $x_0$ while keeping $x_0 \geq 0$ produces a value for x that satisfies $Ax \leq b + x_0, x \geq 0$.
- If the optimal value for $x_0$ is positive then the original problem is infeasible, otherwise I have found a b.f.p. for the original problem.

- What happens if a problem is degenerate?
  - Answer: the Simplex Method is not guaranteed to converge.
  - It may "cycle" through a set of vertices forever.
  - To prevent this a more subtle choice of the leaving variable at each iteration is needed.
  - With this choice it can be shown that the Simplex Method converges.

## 9.7 Exercises

1. Convert the following linear program to standard form:

   $$\max_{x,y} c^T x + d^T y \quad \text{s.t.}\, A_1 x = b_1, A_2 x + B_2 y \le b_2, l \le y \le u,$$

   where there are no explicit bounds on x.

2. Verify that the dual of the dual is the primal.

3. Show that the dual of the linear program
   $\min c^T x \quad \text{s.t.} Ax \geq b, x \geq 0$ is $\max b^T \lambda \quad \text{s.t.}\, A^T \lambda \leq c, \lambda \geq 0$.

4. Consider the following linear program:

$$\begin{aligned} \min -5x_1 - x_2 \quad & \text{subject to} \\ x_1 + x_2 &\le 5 \\ 2x_1 + (1/2)x_2 &\le 8, \\ x &\ge 0. \end{aligned}$$

   (a) Add slack variables $x_3$ and $x_4$ to convert this problem to standard form.

   (b) Using Alg. 9.1, solve this problem using the Simplex method, showing at each step the basis and the vectors $\lambda$, $s_N$ and $x_B$ and the value of the objective function. (The initial choice of $\mathcal{B}$ for which $x_B \ge 0$ should be obvious once you have added the slacks in part (a).)

# 10 Quadratic Programs

- In this final Chapter I examine Quadratic Programs — the simplest kind of **non-linear** constrained optimisation problem.
- Geometrically, Quadratic and Linear Programs are similar, differing only in that QP's have quadratic objective functions.
- The constraints are linear in both cases.
- Most of the ideas encountered when studying QP's carry over to the case of a general non-linear program.

## 10.1 QP Definitions

A **quadratic program** QP is a constrained optimisation problem with a quadratic objective function and linear constraints:

$$\min_x q(x) = \frac{1}{2} x^T Q x + x^T d, \tag{10.1a}$$

$$\text{subject to} \quad a_i^T x = b_i, \quad i \in \mathcal{E} \tag{10.1b}$$

$$a_i^T x \geq b_i, \quad i \in \mathcal{I}, \tag{10.1c}$$

where $Q$ is a symmetric $n \times n$ matrix; $\mathcal{E}$ and $\mathcal{I}$ are finite sets of indices and $d$, $x$ and $\{a_i\}$, $i \in \mathcal{E} \cap \mathcal{I}$ are vectors in $\mathbb{R}^n$and $\{b_i\}$, $i \in \mathcal{E} \cap \mathcal{I}$ are scalars.

- QP's can always be solved (or shown to be infeasible) in a finite number of iterations.
- It is not hard to see that a general nonlinear constrained optimisation problem can be approximated by a QP **at each iteration**.
- So by solving a succession of QP's we can solve a general nonlinear constrained problem.
- This leads to the standard method for solving constrained problems, Sequential Quadratic Programming (SQP).
- I won't say more about SQP in this course but a good understanding of methods for QP's is clearly the key to SQP.

## 10.2 Example QP's

**Example 10.1 (Portfolio Optimisation)**

- *Suppose an investor has a set of* $n$ *possible investments with returns* $r_i$*,* $i = 1, \dots, n$*.*
- *The returns* $r_i$ *are not usually known a priori but are often modelled as random variables from a normal distribution with known means* $\mu_i = E[r_i]$ *and variances* $\sigma_i^2 = E[r_i^2] - \mu_i^2$*.*
- *A "portfolio" is just an allocation of a fraction of resources to each of the* $n$ *investments, say* $x_i$*,* $i = 1, \dots, n$ *where* $\sum_{i=1}^{n} x_i = 1$ *and* $x_i \geq 0$*.*
- *The "return" is just* $R = \sum_{i=1}^{n} x_i r_i$ *and the expected return is* $E[R] = x^T \mu$*.*

- *The variance of the return depends on the covariances between each pair of investments,*

$$\rho_{ij} = \frac{E[(r_i - \mu_i)(r_j - \mu_j)]}{\sigma_i \sigma_j}.$$

- *The variance of the portfolio* $R$ *is then given by*

$$E[(R - E[R])^2] = \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j \sigma_i \sigma_j \rho_{ij} = x^T Q x,$$

- *Here* $Q = \Sigma P \Sigma$ *is an* $n \times n$ *symmetric and positive definite matrix — (how do I know that* $Q$ *is positive definite?).*

- *The strategy is usually to choose portfolios where the expected return* $x^T \mu$ *is large and the variance* $x^T Q x$ *is small (simultanaeously greedy and cowardly!).*

- *To combine these two apparently contradictory objectives I can form a single combined objective function a using a "risk tolerance" parameter $\kappa \ge 0$ — a conservative investor will choose a large value for $\kappa$ — and solve the following QP to find the "best" portfolio:*

$$\max x^T \mu - \kappa x^T Q x, \textit{ subject to } \sum_{i=1}^{n} x_i = 1 \textit{ and } x_i \ge 0.$$

- ***Exercise* 10.1** *Write down the KKT necessary conditions for this problem.*

  ***Exercise* 10.2** *Are the second-order necessary and sufficient conditions satisfied?*

  ***Exercise* 10.3** *Can you find the solution $x$ directly from the KKT conditions?*

Now for a more complex and practical Example.

**Example 10.2** *Each day the Electricity Regulator assigns to each of the 100+ Irish electricity generators large and small a production schedule for the next day based on projected demand.*

- *The is a (hard) mixed Integer Linear program.*
  - *Fortunately I don't need to solve it.*
- *The Regulator does this on the basis of (honestly) stated costs fixed and marginal costs provided by the generators.*
- *After each day (more or less) the Regulator is required to apply a correction to the half-hourly electricity price for the day to ensure that all generators at least have their fixed and marginal costs met — as the actual amounts provided may not equal those specified in the schedule.*

*See Appendix A.31 for the full details.*

The remainder of the Chapter falls into two sections.

First I look at the important special case where (unlike the examples above) **all** the constraints are equality constraints. For this case, the KKT necessary conditions lead to a straightforward matrix equation for the optimal solution $x^*$.

For the general case of equality & inequality constraints; **if** I knew which inequality constraints were binding at the solution (which I don't) then I could reduce the problem to that of solving an equality-constrained problem consisting of the given equality constraints together with the active (at the optimal point) inequality constraints **treated as equality constraints**. In the final section I will see how to build an algorithm (reminiscent of the Simplex method from Ch. 9) for the general case of equality & inequality constraints based on this idea.

## 10.3 Equality Constrained QP's

Assume that I have $m$ constraints $m < n$ and write the QP as

$$\min_x q(x) = \frac{1}{2}x^T Qx + x^T d, \tag{10.2a}$$

$$\text{subject to} \quad Ax = b, \tag{10.2b}$$

where $A$ is the $m \times n$ "Jacobian" matrix of constraints — so the columns of $A^T$ are the vectors $a_i$. (Remember our convention that all vectors are columns.)

Assume that the matrix $A$ has "full row rank", i.e. that its rows are linearly independent (just the LICQ) and that the problem has a solution.

It is easy to check that the first-order necessary conditions (the KKT conditions (8.23a–8.20e) ) for $x^*$ to be a solution of (10.2a–10.2b) mean there must be a vector $\lambda^*$ of Lagrange multipliers such that the following system of equations is satisfied:

$$\begin{bmatrix} Q & -A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} x^* \\ \lambda^* \end{bmatrix} = \begin{bmatrix} -d \\ b \end{bmatrix} \tag{10.3}$$

The matrix in (10.3) is not symmetric — making solution more difficult.

It is trivial to rewrite the equation as

$$\begin{bmatrix} Q & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} -x^* \\ \lambda^* \end{bmatrix} = \begin{bmatrix} d \\ -b \end{bmatrix} \tag{10.4}$$

where the coefficient matrix is now symmetric.

It is convenient to write $x^* = x_0 + p$, where $x_0$ is **any** estimate of the solution and $p$ the required step to the solution. Now the matrix equation can be re-written as:

$$\begin{bmatrix} Q & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} -p \\ \lambda^* \end{bmatrix} = \begin{bmatrix} g \\ r \end{bmatrix} \tag{10.5}$$

where the residual $r = Ax_0 - b$, $g = d + Qx_0$ (the gradient of $f(x_0)$) and $p = x^* - x_0$.

Because of its source, the matrix in (10.5) is called the KKT matrix and the following Theorem tells us when it is non-singular.

**Lemma 10.1** *Let* $A$ *have full row rank and let* $Z$ *be the* $n \times (n-m)$ *matrix whose columns form a basis for the null space of* $A$*. (*$AZ = 0$*.) Then, if the "reduced-Hessian" matrix* $Z^T QZ$ *is positive definite, the KKT matrix*

$$K = \begin{bmatrix} Q & A^T \\ A & 0 \end{bmatrix} \tag{10.6}$$

*is non-singular and so there is a unique* $(x^*, \lambda^*)$ *satisfying* (10.3)*.*

**Proof:** Suppose that there are vectors $r$ and $s$ such that

$$\begin{bmatrix} Q & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} r \\ s \end{bmatrix} = 0. \tag{10.7}$$

As $A r = 0$, I have from (10.7) that

$$0 = \begin{bmatrix} r & s \end{bmatrix} \begin{bmatrix} Q & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} r \\ s \end{bmatrix} = r^T Q r.$$

As $r$ is in the null space of $A$, I can write $r = Z u$ for some vector $u$ in $\mathbb{R}^{n-m}$. So I have

$$0 = r^T Q r = u^T Z^T Q Z u,$$

which as $Z^T Q Z$ is positive definite means that $u = 0$. Therefore $r = 0$ and so $A^T s = 0$ by (10.7). As $A$ has full row rank conclude that $s = 0$. So $K \begin{bmatrix} r \\ s \end{bmatrix} = 0$ implies that $\begin{bmatrix} r \\ s \end{bmatrix} = 0$ and so $K$ is non-singular. ■

### 10.3.1 Second-Order Necessary & Sufficient Conditions for Equality-constrained Problems

- The general second-order necessary conditions were discussed in the previous Chapter and stated and proved in Thm. 8.4.

- Fortunately in the case of an equality-constrained problem they are quite simple.

- First I note that the set $F_2$ of stationary directions is just:

$$F_2 = \left\{ d | d^T a_i = 0, i \in \mathcal{E} \right\},$$

  the directions perpendicular to all the constraints, so $F_2$ is just the null space of $A$.

- As usual, I can construct a matrix $Z$ whose columns form a basis for the null space of $A$.

- Then the necessary condition (8.35) reduces (as the Hessian of the Lagrangian is just $\mathsf{Q}$) to requiring that $\mathsf{Z}^\mathsf{T}\mathsf{Q}\mathsf{Z}$ be positive semi-definite.
- Similarly, the second-order sufficient conditions (8.40) just require that $\mathsf{Z}^\mathsf{T}\mathsf{Q}\mathsf{Z}$ be positive definite.
- Exactly as I found in Lemma 10.1!

### 10.3.2 Solving the KKT equation

I know that the matrix $K$ is non-singular provided the "reduced-Hessian" matrix $Z^T QZ$ is positive definite, so having chosen an arbitrary initial guess for the solution, $x$, the matrix equation (10.5) can be solved for $p$ and $\lambda$, Then $x^* = x + p$. The matrix $K$ is $(n + m) \times (n + m)$ where $m$ is the number of equality constraints. This can give rise to a very large set of equations.

It can be shown that if $Z^T QZ$ is positive definite then the KKT matrix **always** has $n$ positive and $m$ negative eigenvalues, so standard methods for solving large linear systems that depend on the matrix being positive definite (such as Cholesky decomposition) will not work.

An interesting alternative is the "Range-Space" method, where I eliminate $p$ from (10.5) and then solve for $\lambda$.

**Range-Space Method** I can dissect (10.5) into two matrix equations:

$$-Qp + A^T\lambda = g \tag{10.8}$$

$$Ap = -r \tag{10.9}$$

where $r = Ax_0 - b$, $g = d + Qx_0$ (the gradient of $q(x_0)$) as before — and $x_0$ is an arbitrary starting guess for the solution $x^*$.

Now (provided $Q$ is non-singular) I multiply (10.8) by $AQ^{-1}$ and cancel the term in $p$ with (10.9), giving a matrix equation for $\lambda^*$ alone:

$$\left(AQ^{-1}A^T\right)\lambda^* = \left(AQ^{-1}g - r\right). \tag{10.10}$$

Once $\lambda^*$ has been found, I can solve (10.8) for $p$.

As the method requires that I work with $Q^{-1}$ it is most efficient when one or more of the following holds:

- $Q$ is diagonal or block-diagonal (easy to invert).
- $Q^{-1}$ available at no extra cost using a quasi-Newton method like BFGS.
- The number of equality constraints $m$ is small, so that the linear system (10.10) is easily solved.

**Example 10.3** *Consider the problem*

$$\min q(x) = 3x_1^2 + 2x_1x_2 + x_1x_3 + 2.5x_2^2 + 2x_2x_3 + 2x_3^2 - 8x_1 - 3x_2 - 3x_3,$$

$$\textit{subject to the constraints } x_1 + x_3 = 3 \textit{ and } x_2 + x_3 = 0.$$

*I can write this problem in the format (10.2a–10.2b) by defining*

$$Q = \begin{bmatrix} 6 & 2 & 1 \\ 2 & 5 & 2 \\ 1 & 2 & 4 \end{bmatrix}, d = \begin{bmatrix} -8 \\ -3 \\ -3 \end{bmatrix}, A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}, b = \begin{bmatrix} 3 \\ 0 \end{bmatrix}.$$

*The solution* $x^*$ *and optimal Lagrange multiplier vector* $\lambda^*$ *are:*

$$x^* = \begin{bmatrix} 2 \\ -1 \\ 1 \end{bmatrix}, \lambda^* = \begin{bmatrix} 3 \\ -2 \end{bmatrix}$$

*The matrix* $\mathsf{Q}$ *is positive definite and the null-space basis matrix can be taken to be* $\mathsf{Z} = (-1, -1, 1)^\mathsf{T}$*.*

*First check that* $\mathsf{Z}^\mathsf{T}\mathsf{Q}\mathsf{Z}$ *is positive definite so the KKT matrix is guaranted to be invertible and the second-order sufficient condition is satisfied (so the solution to the KKT matrix equation is guaranteed to be a strong local solution to the problem).*

***Exercise*** ***10.4*** *Using Matlab (or Maple), solve the problem using both the direct method and the Range-Space method.*

## 10.4 Inequality Constrained QP's

Most Quadratic Programs (and constrained problems in general) include inequality constraints. The following is a simple treatment which omits much of the underlying theory but allows an algorithm to be presented.

### 10.4.1 Optimality Conditions for Inequality Constrained QP's

Start by writing the Lagrangian for (10.1a–10.1c) as

$$\mathcal{L}(x,\lambda) = \frac{1}{2}x^T Q x + x^T d - \sum_{i \in \mathcal{I} \cup \mathcal{E}} \lambda_i (a_i^T x - b_i). \tag{10.11}$$

As before, define the **active set** $\mathcal{A}(x^*) = \mathcal{E} \cup \{i \in \mathcal{I} : a_i^T x^* = b_i\}$ at the optimal point $x^*$ to be the set of indices where equality holds.

The KKT conditions (8.23a–8.20e) when applied to (10.11) take the form:

$$Qx^* + d - \sum_{i \in \mathcal{A}(x^*)} \lambda_i^* a_i = 0 \tag{10.12a}$$

$$a_i^T x^* = b_i, \quad \text{for all } i \in \mathcal{A}(x^*), \tag{10.12b}$$

$$a_i^T x^* > b_i, \quad \text{for all } i \in \mathcal{I} \backslash \mathcal{A}(x^*), \tag{10.12c}$$

$$\lambda_i^* \geq 0, \quad \text{for all } i \in \mathcal{I} \cap \mathcal{A}(x^*). \tag{10.12d}$$

$$\lambda_i^* (a_i^T x^* - b_i) = 0, \quad \text{for all } i \in \mathcal{I} \cup \mathcal{E}. \tag{10.12e}$$

- The last condition is just complementarity — requiring that that inactive inequality constraints have zero multipliers.
- As the constraints are linear I do not need the LICQ.

- The second-order sufficient conditions are simply that $Z^T Q Z$ be positive definite where $Z$ is defined as usual to be a null-space basis matrix for the matrix whose columns are the active (at $x^*$ ) constraint vectors $a_i, i \in \mathcal{A}(x^*)$.
- I will consider only the convex case — where $Q$ is positive semidefinite.
- So the second-order sufficient conditions automatically hold.

### 10.4.2 Active Set Methods for Convex Inequality Constrained QP's

- If $\mathcal{A}(x^*)$ were known in advance the problem (10.1a–10.1c) reduces to just solving an equality-constrained QP with the active constraints treated as equality constraints.
- In practice, of course, I do not know the active set a priori so we must use an iterative method where constraints are added to & removed from a **working** set $\mathcal{W}_k$ at each iteration.
- I expect/hope that at the final iteration $\mathcal{W}_k = \mathcal{A}(x^*)$ and so the solution of this final equality-constrained problem will be the solution to (10.1a–10.1c) .
- I start at a point $x_0$ that is feasible with respect to **all** the constraints — this will be true at every iteration.
- I will look at the choice of starting point in Section 10.4.5.

- Suppose that I have an iterate $x_k$ and a working set $\mathcal{W}_k$ at the $k^{th}$ iteration.
- First I check whether $x_k$ minimises the quadratic $q$ in the subspace defined by $\mathcal{W}_k$ .
- If not, I calculate a step $p$ by solving an equality-constrained QP subproblem in which the constraints in $\mathcal{W}_k$ are treated as equalities and all other constraints are (temporarily) ignored.
- To express the sub-problem in terms of the step $p$, I define

  $$p = x - x_k, \quad g_k = Qx_k + d$$

  and by substituting for $x$ into the objective $q(x)$ (10.1a) I get

  $$q(x) = q(x_k + p) = \frac{1}{2}p^T Qp + g_k^T p + c$$

  where $c = \frac{1}{2}{x_k}^T Qx_k + d^T x_k$ is a constant and so may be ignored.

- So the QP sub-problem to be solved at the $k^{th}$ iteration is:

$$\min_p \frac{1}{2} p^T Q p + g_k^T p \tag{10.13a}$$

$$\text{subject to } a_i^T p = 0 \text{ for all } i \in \mathcal{W}_k. \tag{10.13b}$$

- Call the solution to this problem $p_k$.
- If $x_k + p_k$ is feasible with respect to all constraints, then set $x_{k+1} = x_k + p_k$.

- If not I need to decide "how far" to move along the direction $p_k$ , i.e. what value of $\alpha$ to choose in the formula $x_{k+1} = x_k + \alpha p_k$.
  - For each $i \in \mathcal{W}_k$, the term $a_i^T x$ doesn't change as we move along $p_k$ from $x_k$ because $a_i^T(x_k + p_k) = a_i^T x_k = b_i$ — so since the constraints in $\mathcal{W}_k$ were satisfied at $x_k$ , they are still satisfied at $x_k + \alpha p_k$ for any $\alpha$.
  - So focus on the "non-working" set $\mathcal{W}_k^c$.
    - $*$ The only non-working constraints $i \in \mathcal{W}_k^c$ that can "enter" the working set are those for which $a_i^T p_k$ is negative .
    - $*$ To see this, note that if $a_i^T p_k \geq 0$ for some $i \in \mathcal{W}_k$ then for any $\alpha_k \geq 0$ I have $a_i^T(x_k + \alpha_k p_k) \geq a_i^T x_k \geq b_i$ — so this constraint is satisfied for all non-negative values of the step-length $\alpha_k$.

- On the other hand, suppose that $a_i^T p_k < 0$ for some $i \in \mathcal{W}_k^c$. Then $a_i^T(x_k + \alpha_k p_k) \geq b_i$ only if

$$\alpha_k \leq \frac{b_i - a_i^T x_k}{a_i^T p_k}.$$

- As I want $\alpha_k$ to be as large as possible in the range $[0, 1]$ subject to maintaining feasibility, I conclude that

$$\alpha_k \equiv \min\left(1, \min_{i \in \mathcal{W}_k^c, a_i^T p_k < 0} \frac{b_i - a_i^T x_k}{a_i^T p_k}\right). \tag{10.14}$$

- In other words, choose $\alpha$ as large as possible so that **all** the constraints hold at $x_k + \alpha p_k$.
- Or equivalently, find the first "blocking" constraint index $i \in \mathcal{W}_k^c$ — for which $a_i^T(x_k + \alpha_k p_k) = b_i$.

- Call the constraints where the minimum is attained **blocking** constraints.
  - If $\alpha_k = 1$ and no new constraints are active at $x_k + \alpha_k p_k$ then there are no blocking constraints (an unconstrained step).
  - If $\alpha_k < 1$ — the step along $p_k$ is blocked by one or more constraints not in the working set $\mathcal{W}_k$ — I construct a **new** working set $\mathcal{W}_{k+1}$ by adding one of the blocking constraints to $\mathcal{W}_k$ .
  - The special case $\alpha_k = 0$ corresponds to $a_i^T p_k < 0$ for one or more constraints active at $x_k$ $(a_i^T x_k = b_i)$ but not in the working set — again construct a working set $\mathcal{W}_{k+1}$ by adding one of these blocking constraints to $\mathcal{W}_k$ .
- Now set $x_{k+1} = x_k + \alpha_k p_k$ and solve the QP-subproblem (10.13a), (10.13b) with $\mathcal{W}_k$ replaced by $\mathcal{W}_{k+1}$ and $g_k$ by $g_{k+1}$.

- I continue in this manner; adding constraints to the working set, moving to the new point and re-solving until I reach a point $\hat{x}$ that minimises the quadratic objective function over the current working set $\hat{\mathcal{W}}$ .

- I know when I have arrived at such a point as the sub-problem has solution $p = 0$.

- As $p = 0$ satisfies the KKT conditions (10.5) for (10.13a–10.13b), I have

$$\sum_{i \in \hat{\mathcal{W}}} a_i \hat{\lambda}_i = g \equiv Q\hat{x} + d. \tag{10.15}$$

- To make $\hat{x}$ & $\hat{\lambda}$ satisfy the first KKT condition (8.23a), just define the multipliers corresponding to the inequality constraints not in the working set to be zero.

- Because of the restrictions on the stepsize to ensure feasibility, $\hat{x}$ is feasible wrt **all** the constraints so the second & third KKT conditions (8.20b) and (8.20c) are satisfied by $\hat{x}$ .
- Now examine the **sign** of the multipliers corresponding to the inequality constraints in the working set ($i \in \hat{\mathcal{W}} \cap \mathcal{I}$).
  - If the multipliers are all nonnegative, then the fourth KKT condition (8.20d) is satisfied so I conclude that $\hat{x}$ is a KKT point for the original QP (10.1a–10.1c) .
    - As I have taken Q to be positive semidefinite, in fact we have that $\hat{x}$ is a local minimum point.
    - If Q is positive definite then $\hat{x}$ is a global minimum point.

- On the other hand, if one of the multipliers $\hat{\lambda}_j$ for $j$ in $\hat{\mathcal{W}} \cap \mathcal{I}$ is negative, then (8.20d) is not satisfied and the objective $q(x)$ may be reduced by dropping this constraint (allowing movement to the "interior" or feasible side of this constraint).
  - I remove the index $j$ corresponding to the most negative multiplier and solve a new sub-problem for this working set.
  - I will see in Sec 10.4.3 that this strategy produces a direction $p$ at the next iteration that is feasible wrt the dropped constraint.

I can now state an algorithm for solving (convex) QP's.

**Algorithm 10.1 (Active-Set Method for convex QP's)**

```
begin
  Compute a feasible start point $x_0$
  set $\mathcal{W}_0$ to be a subset of the active constraints at $x_0$
  while $k \geq 0$ do
      Solve (10.13a–10.13b) to find $p_k$.
      if $p_k = 0$
        then Compute Lagrange multipliers $\hat{\lambda}_i$ that satisfy (10.15)
            set $\hat{\mathcal{W}} = \mathcal{W}_k$
            if $\hat{\lambda}_i \geq 0$ for all $i \in \mathcal{W}_k \cap \mathcal{I}$
              then stop with solution $x^* = x_k$ (FINISHED)
              else
                  set $j = \arg \min_{j \in \mathcal{W}_k \cap \mathcal{I}} \hat{\lambda}_j$ (constraint corr. to most neg. $\lambda_j$)
                  set $x_{k+1} \leftarrow x_k$
                  set $\mathcal{W}_{k+1} \leftarrow \mathcal{W}_k \setminus \{j\}$ (drop constraint $j$ )
            fi
        else (p_k not zero.)
            Compute $\alpha_k$ from (10.14)
            $x_{k+1} \leftarrow x_k + \alpha_k p_k$
            if there are blocking constraints
              then Get $\mathcal{W}_{k+1}$ by adding one of the blocking constraints to $\mathcal{W}_k$
              else $\mathcal{W}_{k+1} \leftarrow \mathcal{W}_k$
            fi
      fi
      $k \leftarrow k + 1$;
  end (while)
end
```

### 10.4.3 Maintenance of Feasibility After Constraint is Dropped

I need to check that, when a constraint corresponding to a negative multiplier is dropped at line 12 in Algorithm 10.1, the direction $\mathbf{p}$ found at the next iteration is guaranteed to be feasible wrt the dropped constraint.

(So I do not "cycle" back and forth; alternatively adding and removing the same constraints to & from the working set.)

It is convenient to state the result as a Theorem.

**Theorem 10.2** *Suppose that the point $\hat{x}$ satisfies the first-order KKT conditions for the equality-constrained sub-problem with working set $\hat{\mathcal{W}}$; so* (10.15) *is satisfied together with $a_i^T \hat{x} = b_i$ for all $i \in \hat{\mathcal{W}}$. Suppose that the constraint gradients are linearly independent and that the multiplier for one of the indices $j \in \hat{\mathcal{W}}$ is negative ($\hat{\lambda}_j < 0$). Let $p$ be the solution to the "next" sub-problem:*

$$\min_{p} \frac{1}{2} p^T Q p + (Q\hat{x} + d)^T p, \tag{10.16a}$$

$$\textit{subject to } a_i^T p = 0, \textit{ for all } i \in \hat{\mathcal{W}} \backslash \{j\}. \tag{10.16b}$$

*Then $p$ is a feasible direction for constraint $j$ ($a_j^T p \geq 0$).*

*Also, if $p$ satisfies second-order sufficient conditions for the "next" sub-problem, I have $a_j^T p > 0$.*

**Proof:** Since the direction $p$ is the solution to (10.16a) & (10.16b), I know that for all $i \in \hat{\mathcal{W}} \backslash \{j\}$ there are multipliers $\tilde{\lambda}_i$ such that

$$\sum_{i \in \hat{\mathcal{W}} \backslash \{j\}} \tilde{\lambda}_i a_i = Qp + (Q\hat{x} + d). \tag{10.17}$$

I also have (from the second-order necessary conditions) that if — as usual — $Z$ is a matrix whose columns form a basis for the null space of the submatrix of $A^T$ corresponding to $\hat{\mathcal{W}} \backslash \{j\}$ namely

$$[a_i]_{i \in \hat{\mathcal{W}} \backslash \{j\}}$$

then $Z^T Q Z$ is positive semi-definite. As $p = Zp_Z$ for some vector $p_Z$ it follows that $p^T Q p \geq 0$.

I assumed that $\hat{x}$ and $\hat{\mathcal{W}}$ satisfy (10.15) so subtracting that equation from (10.17) above I get:

$$\sum_{i \in \hat{\mathcal{W}} \backslash \{j\}} (\tilde{\lambda}_i - \hat{\lambda}_i) a_i - \hat{\lambda}_j a_j = Qp. \tag{10.18}$$

Taking inner products of both sides with $p$ and using $a_i^T p = 0$ for all $i \in \hat{\mathcal{W}} \backslash \{j\}$, I get

$$-\hat{\lambda}_j a_j^T p = p^T Q p. \tag{10.19}$$

As $p^T Q p \geq 0$ and $\hat{\lambda}_j < 0$ by assumption, it follows that $a_j^T p \geq 0$ as required.

Finally, if the second-order sufficient conditions are satisfied, I have that $Z^T Q Z$ as defined above is positive definite. I have just shown that $a_j^T p \geq 0$. Now assume that $a_j^T p = 0$. By (10.19) above I have that $p^T Q p = 0$. But $p = Z p_Z$ so $p_Z^T Z^T Q Z p_Z = 0$. As $Z^T Q Z$ is positive definite I must have $p_Z = 0$ and so $p = 0$. The last step is to substitute $p = 0$ into (10.18). By linear independance of $\hat{\mathcal{W}}$, I find that $\hat{\lambda}_j = 0$ which contradicts the fact that $j \in \hat{\mathcal{W}}$ was chosen so that $\hat{\lambda}_j < 0$. So $a_j^T p > 0$. ■

### 10.4.4 Reduction in $q$ at each iteration

- It is easy to see that — if the solution $p_k$ to (10.13a) & (10.13b) is nonzero and satisfies second-order sufficient conditions for the current $\mathcal{W}_k$ — $p_k$ is a strict descent direction for $q(\cdot)$.
- Argue as follows: $p_k$ is the unique global solution of (10.13a) & (10.13b) (as second-order sufficient conditions hold).
- But $p = 0$ is obviously feasible (satisfies (10.13b)) so its objective value of zero must be bigger than that of $p_k$ so

$$\frac{1}{2}{p_k}^T Q p_k + g_k^T p_k < 0.$$

- I have $Q$ positive semi-definite so $g_k^T p_k < 0$.

- So, for $\alpha$ sufficiently small:

$$q(x_k + \alpha pk) = q(x_k) + \alpha g_k^T + \frac{1}{2}\alpha^2 p_k{}^T Q p_k < q(x_k).$$

**Linear Independence of the Working Set** I have assuned in the above that the constraint gradients in the working set $\mathcal{W}_k$ are linearly independent at each iteration. See Exercise 6 at the end of this Chapter for a justification.

### 10.4.5 Choosing a Start Point

Probably the most conceptually straightforward method to generate a starting feasible point $x_0$ is the "big $M$" method that includes a measure of infeasibility in the objective that is guaranteed to be zero at the solution. (The method is sometime used to generate starting feasible points for LP)

I introduce a scalar “artificial” variable $t$ to measure the constraint violation and solve the problem:

$$
\begin{aligned}
& \min_{x,t} \frac{1}{2} x^T Q x + x^T d + M t, && \text{(10.20a)} \\
\text{subject to } & t \geq \left(a_i^T x - b_i\right) \quad \text{for all } i \in \mathcal{E}. && \text{(10.20b)} \\
& t \geq -\left(a_i^T x - b_i\right) \quad \text{for all } i \in \mathcal{E}. && \text{(10.20c)} \\
& t \geq -a_i^T x + b_i \quad \text{for all } i \in \mathcal{I}. && \text{(10.20d)} \\
& t \geq 0 && \text{(10.20e)}
\end{aligned}
$$

for some large positive value of $M$.

It can be shown that — if the original problem (10.1a–10.1c) is feasible and if the constant $M$ is large enough — the solution of (10.20a–10.20e) will have $t = 0$, with a vector $x$ that is feasible for the original problem. An initial solution to (10.20a–10.20e) can be just some "guess" for $x$ and a value of $t$ large enough to ensure that all the constraints are satisfied.

**Example 10.4** *In this final example I will solve the following problem:*

$$\begin{aligned} \min_x q(x) &= (x_1 - 1)^2 + (x_2 - 2.5)^2 \\ \textit{subject to } x_1 - 2x_2 + 2 &\geq 0, \\ -x_1 - 2x_2 + 6 &\geq 0, \\ -x_1 + 2x_2 + 2 &\geq 0, \\ x_1 &\geq 0, \\ x_2 &\geq 0. \end{aligned}$$

*So* $Q = 2I$ *and* $d = \begin{bmatrix} -2 \\ -5 \end{bmatrix}$. *The constraint gradients are:*

$a_1 = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$, $a_2 = \begin{bmatrix} -1 \\ -2 \end{bmatrix}$, $a_3 = \begin{bmatrix} -1 \\ 2 \end{bmatrix}$, $a_4 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ *and* $a_5 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$.

*In the following I will use subscripts to indicate components and superscripts to indicate iteration number. In the diagram below the five constraints are numbered (1–5, circled) in the order they appear in the problem statement above.*

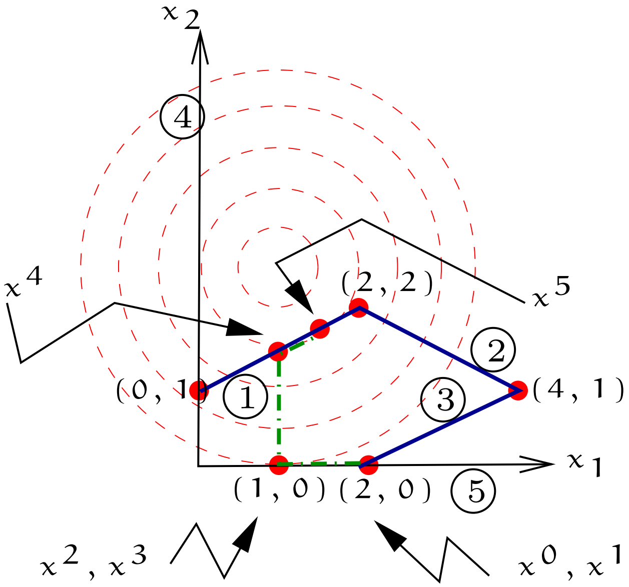


*Figure 19: Final Example*

*For a "toy" problem like this it is easy to pick a vertex as a starting feasible point — I take* $x^0 = (2, 0)$*. (If you want a challenge, try starting at the infeasible point* $(-1, -1)$ *and use the big-M method above to generate a stating feasible point — you will need to use Matlab as the algebra is considerable.)*

- *At* $x^0$*, constraints 3 & 5 are active and I set* $\mathcal{W}_0 = \{3, 5\}$*. (Other choices include* $\mathcal{W}_0 = \{3\}$ *,* $\mathcal{W}_0 = \{5\}$ *and* $\mathcal{W}_0 = \{\}$*. Each results in a different sequence of points leading to the solution.)*

  *As* $x^0$ *is a vertex of the feasible region (as may be seen from the diagram), it obviously minimises the objective function wrt the working set* $\mathcal{W}_0$ *— so the solution to (10.13a–10.13b) with* $k = 0$ *is* $p^0 = 0$*. The feasible region for the sub-problem is the point* $x^0 = (2, 0)$ *— a zero-dimensional space!*

  *Of course if I had taken* $\mathcal{W}_0 = \{3\}, \{5\}$ *or* $\{\}$ *we would have had to do some work at this iteration.*

- *Now I need to use* (10.15) *to find the multipliers* $\hat{\lambda}_3$ *and* $\hat{\lambda}_5$ *corresponding to the active constraints. Substituting for the current problem into* (10.15) *gives:*

$$\begin{bmatrix} -1 \\ 2 \end{bmatrix} \hat{\lambda}_3 + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \hat{\lambda}_5 = \begin{bmatrix} 2 \\ -5 \end{bmatrix},$$

  *which has the solution* $\hat{\lambda}_3 = -2$ *and* $\hat{\lambda}_5 = -1$.

- *Now remove constraint 3 from the working set as it has the most negative multiplier — setting* $\mathcal{W}_1 = \{5\}$*. Begin iteration 1 by finding the solution of (10.13a–10.13b) with* $k = 1$ *— which is* $p^1 = (-1, 0)^T$*. (Check by hand —easy!) The step-length formula* (10.14) *gives us* $\alpha_1 = 1$ *(in other words I take the max step as there is no blocking constraint — see Fig. 19). The new point is* $x^2 = x^1 + \alpha_1 p^1 = (1, 0)$.

- *There are no blocking constraints so* $\mathcal{W}_2 = \mathcal{W}_1 = \{5\}$ *and I find at the start of iteration 2 that the solution to (10.13a–10.13b) is* $p^2 = 0$.

  *From* (10.15) *I find the multiplier for the single working constraint is* $\hat{\lambda}_5 = -5$. *As this is negative, I drop 5 from the working set to finish with* $\mathcal{W}_3 = \emptyset$.

- *The third iteration starts by solving the unconstrained problem — whose solution is* $p^3 = (0, 2.5)$. *The step-length formula* (10.14) *gives us* $\alpha_1 = 0.6$, *a new iterate* $x^4 = (1, 1.5)$ *and a new blocking constraint; constraint 1. So* $\mathcal{W}_4 = \{1\}$.

- *The solution to (10.13a–10.13b) is* $p^4 = (0.4, 0.2)$ *and the new step length is 1. There are no blocking constraints at this step so the working set is unchanged;* $\mathcal{W}_5 = \{1\}$. *The new iterate is* $x^5 = x^4 + p^4 = (1.4, 1.7)$.

- *Finally I solve (10.13a–10.13b) for* $k = 5$ *— the solution is* $p^5 = 0$*. The multiplier from* (10.15) *is* $\hat{\lambda}_1 = 0.8$ *so I have a solution. Set* $x^* = (1.4, 1.7)$ *and STOP!*

*The second-order sufficient condition is as usual* $Z^T QZ$ *positive definite where* $Z$ *satisfies* $a_1^T Z = 0$ *so* $Z = \begin{bmatrix} -2 \\ 1 \end{bmatrix}$ *and* $Z^T QZ$ *is just the scalar* $10$ *which is certainly positive definite so I know that* $x^* = (1.4, 1.7)$ *is a strict local minimum of the problem.*

### 10.4.6 Final Comments on Constrained Optimisation

Of course Quadratic Programs (QP's) are a very special case of general non-linear constrained optimisation. The method described here (improved so it works for non-convex problems) forms the basis for Sequential Quadratic Programming (SQP).

Consider first a general equality-constrained problem:

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{subject to } c_i(x) = 0, \quad i \in \mathcal{E} \tag{10.21}$$

Without going into details; SQP (for an equality-constrained problem) solves the following sub-problem **at each iteration**

$$\begin{bmatrix} W_k & -A_k^T \\ A_k & 0 \end{bmatrix} \begin{bmatrix} p_k \\ \lambda_{k+1} \end{bmatrix} = \begin{bmatrix} -g_k \\ -c_k \end{bmatrix}$$

which is essentially (10.3) for QP's where now $W$ is the Hessian of the Lagrangian $\mathcal{L}(x, \lambda) = f(x) - \sum_{i=1}^{n} \lambda_i c_i(x)$ at the current $(x_k, \lambda_k)$ estimate; $A_k$ is the matrix given by

$$A_k^T = \left[ \nabla c_1(x_k), \nabla c_2(x_k), \ldots, \nabla c_n(x_k) \right],$$

$c_k$ is the vector of constraints at the current point $x_k$ and $x_{k+1} = x_k + p_k$.

Provided $A_k$ is full rank at each iteration and the matrix $W_k$ satisfies $d^T W_k d > 0$ for all non-zero "feasible directions" $d$ (ie directions such that $A_k d = 0$), then (just as for QP's) this linear system has a unique solution.

This (in principle) gives us an algorithm for general non-lineal equality constrained problems.

For an inequality-constrained problem a working set method similar to that described above must be used.

## 10.5 Exercises

1. Solve the following QP and (using Matlab/Maple or by hand) describe it geometrically.

$$\begin{aligned} \min_x \mathsf{q}(x) = 2x_1 + 3x_2 + 4x_1^2 + 2x_1x_2 + x_2^2 \\ \text{subject to } x_1 - x_2 \geq 0, \\ x_1 + x_2 \leq 4, \\ x_1 \leq 3 \end{aligned}$$

2. The problem of finding the shortest distance from a point $x_0$ to the hyperplane $\{x|Ax = b\}$ (where $A$ is an $m \times n$ matrix with full row rank $m$) can be formulated as the following QP:

$$\begin{aligned} \min \frac{1}{2}(x - x_0)^T(x - x_0) \\ \text{s.t. } Ax = b. \end{aligned}$$

(a) Show that the optimal multiplier vector is:

$$\lambda^* = \left(AA^T\right)^{-1}(b - Ax_0)$$

and that the solution is

$$x^* = x_0 - A^T\left(AA^T\right)^{-1}(b - Ax_0)\,.$$

(b) In the special case where $A$ is a row vector ( a single constraint), show that the shortest distance from $x - x_0$ to $x^*$ is

$$\frac{|b - Ax_0|}{\|A\|}.$$

3. Consider the general equality-constrained QP (10.2a) & (10.2a). Suppose that the "projected Hessian" $Z^T Q Z$ has a negative eigenvalue, i.e. that $u^T Z^T Q Z u < 0$ for some vector $u$. Show that if there is a pair $x^*$ and $\lambda^*$ that satisfies the KKT matrix equation (10.3), then the point $x^*$ is only a stationary point of (10.2a) & (10.2a) — not a local minimiser. (Hint: consider the function $q(x^* + \alpha Z u)$ for $\alpha \neq 0$.)

4. For each of the possible choices of starting working set for Example 10.4, work through the first two iterations of the active-set algorithm.

5. Write Matlab code to implement Algorithm 10.1. Use it to solve the following QP:

$$\min_{x} q(x) = x_1^2 + 2x_2^2 - 2x_1 - 6x_2 - 2x_1x_2$$

$$\begin{aligned} \text{subject to } \frac{1}{2}x_1 + \frac{1}{2}x_2 &\le 1, \\ -x_1 + 2x_2 &\le 2, \\ x_1, x_2 &\ge 3. \end{aligned}$$

Sketch the feasible region to obtain a starting point.

6. Using (10.13a) & (10.13b) together with (10.14) show that the gradient of a blocking constraint cannot be a linear combination of the constraint gradients in the current working set $\mathcal{W}_k$. In particular, suppose that the initial working set $\mathcal{W}_0$ in Algorithm 10.1 is chosen so that the constraint gradients corresponding to this working set are linearly independent. Show that the step selection rule (10.14) guarantees that constraint gradients corresponding to all subsequent working sets remain linearly independent.

# A Supplementary Material

## A.1 Introduction to Convexity

Informally;

- The concept of **convexity** is important in optimisation — if it holds then it usually means that the problem has a unique solution and that the solution is more easily found.
- The term **convex** may be applied to both sets and functions.
- The definition is geometric: “For any $x$ and $y$ in $\mathbb{R}^n$, the function value a fraction $\alpha$ along the line joining $x$ and $y$ is less than or equal to the intermediate value a fraction $\alpha$ between $f(x)$ and $f(y)$.”
- More precisely:

  **Definition A.1** *A function* $f$ *is convex if for any* $x$ *and*

$y \in \mathbb{R}^n$

$$f\left(\alpha x + (1-\alpha)y\right) \leq \alpha f(x) + (1-\alpha)f(y) \quad \textit{for all } \alpha \in [0,1].$$

- The (imprecise) equivalence of convexity and the function having a non-negative second derivative or, if multivariate, a positive semi-definite Hessian, will often be all we need.
- The Hessian matrix and the term positive semi-definite are defined in the next Chapter, see Defs. 3.5 and 3.6 below.
- When the second derivative is not everywhere defined our naive definition can be re-stated as requiring that the slope is non-decreasing.
- See Sec. A.2 for a full discussion.

## A.2 Convexity

More formally;

**Definition A.2 (Convex Set)** *A set* $S \in \mathbb{R}^n$ *is* **convex** *if the straight line segment connecting any two points in* $S$ *lies entirely inside* $S$*. Formally; if for any two points* $x, y \in S$*, we have* $\alpha x + (1 - \alpha)y \in S$ *for all* $\alpha \in [0, 1]$ *then the set* $S$ *is convex.*

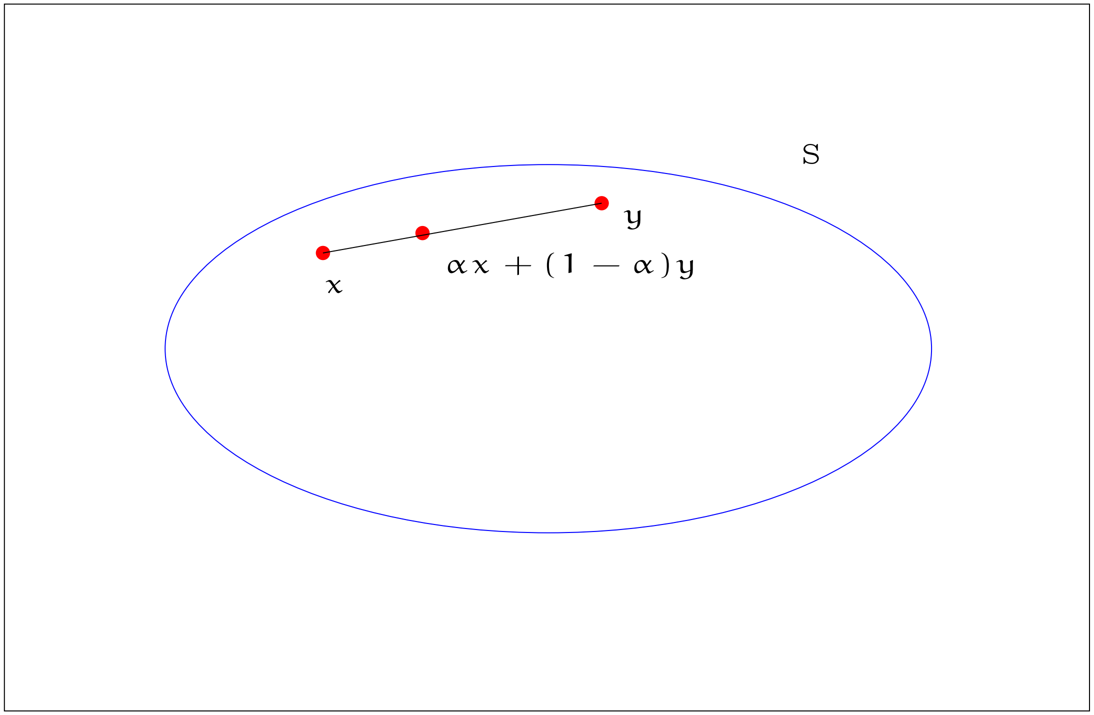


Figure 20: A convex set $S$

**Definition A.3 (Convex Function)** *A (real-valued) function* $f$ *is* **convex** *if its domain is a convex set and if for* **any** *two points* $x$ *and* $y$ *in its domain, the graph of* $f$ *lies below the straight line connecting the point* $(x, f(x))$ *to the point* $(y, f(y))$ *in the space* $\mathbb{R}^{n+1}$*. Precisely; we have*

$$f\,(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y) \quad \textit{for all } \alpha \in [0, 1].$$

*Simpler than it looks: — see Fig 21.*

- $\alpha x + (1-\alpha)y$ *is a point a fraction* $\alpha$ *along the line joining* $x$ *and* $y$ *in* $\mathbb{R}^n$.
- $\alpha f(x) + (1-\alpha)f(y)$ *is a fraction* $\alpha$ *along the interval* $[f(x), f(y)]$.

*It is easy to miss the* **any** *qualification! . See Fig 22. A convex curve lies below the line segment joining* **any** *two points on the curve.*

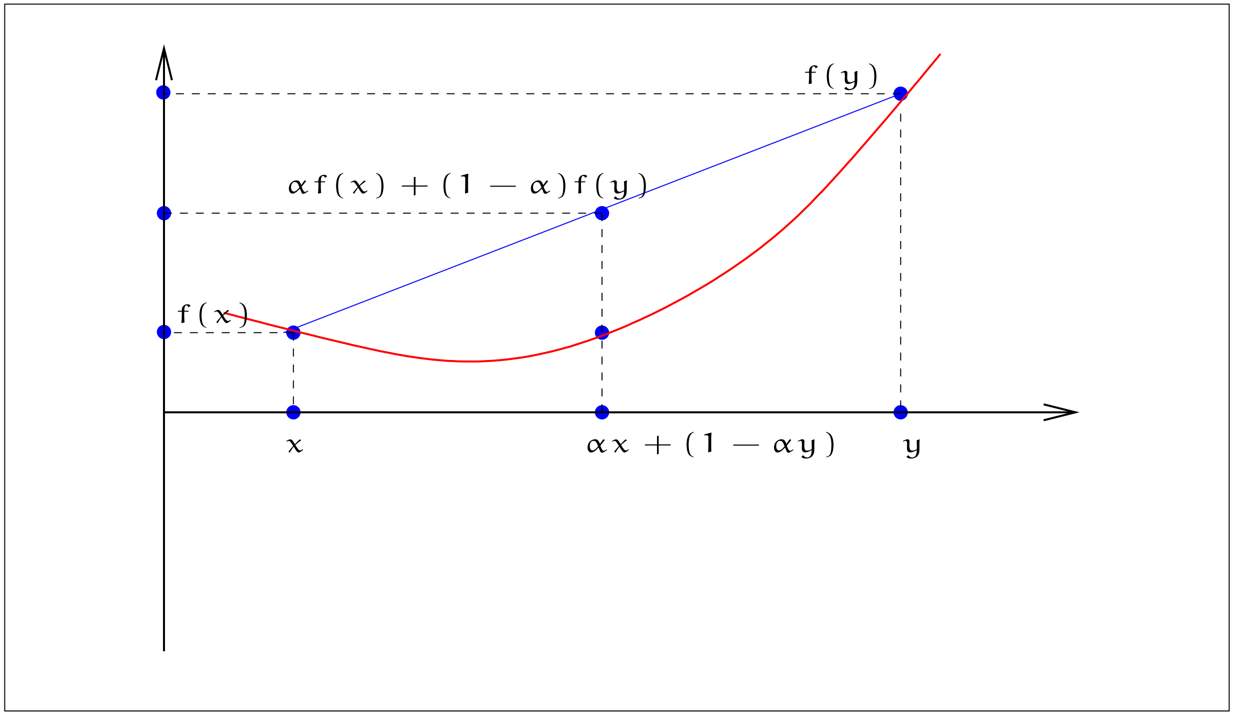


Figure 21: A convex function $f$

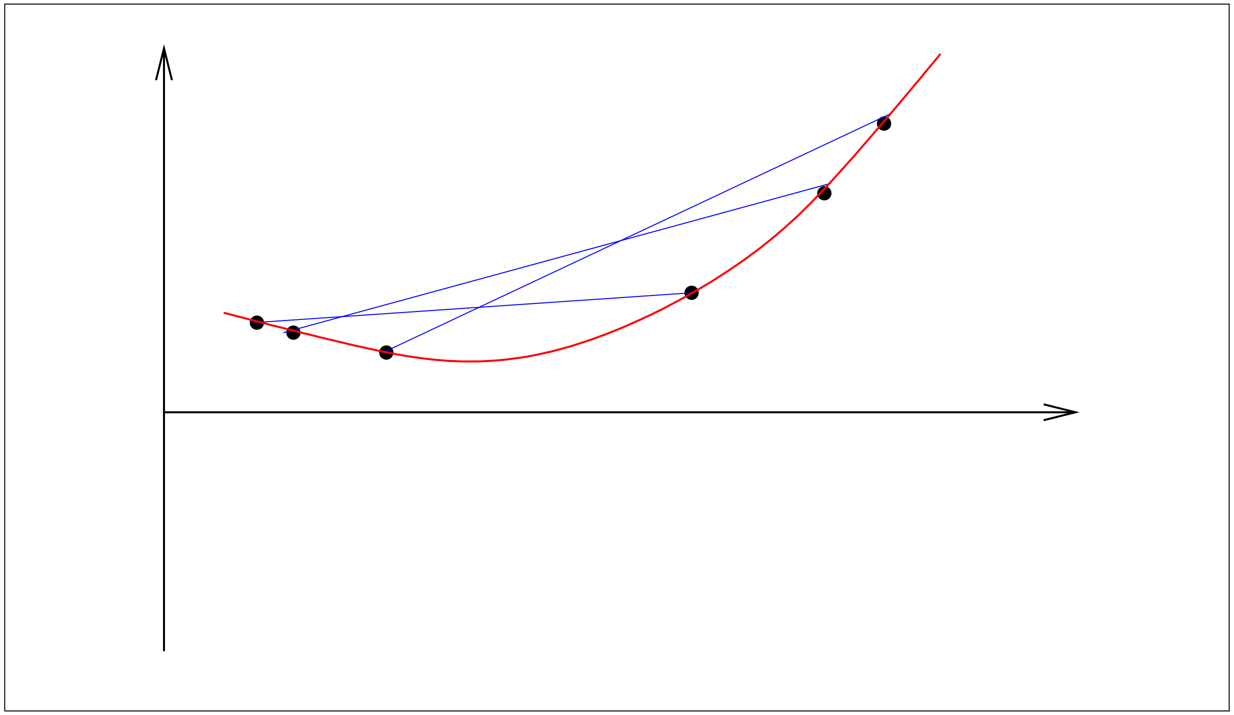


Figure 22: Convex function f — multiple endpoints

**Definition A.4** *A function $f$ is said to be concave if $-f$ is convex.*

In Appendix A.3 we state & prove some theorems concerning convex functions. The conclusion will be that (if the function $f$ is differentiable at the points in question) then $f$ is convex on $\mathbb{R}$ iff it has non-decreasing slope. Also if the second derivative is defined and continuous at the points in question then $f$ is convex iff $f'' \geq 0$.

A similar result holds on $\mathbb{R}^n$:

**Theorem A.1** *Let $f$ be a function $f : \mathbb{R}^n \to \mathbb{R}$ with continuous second derivative on $\mathbb{R}^n$. Then $f$ is convex if and only if $\nabla^2 f(x)$ is positive semi-definite for all $x$. (Note: the* **gradient** *$g(x)$ and the Hessian $\nabla^2 f(x)$ are defined in Eqs. 3.4 and 3.5 below.)*

**Proof:** As the proof is algebraically involved and uses ideas from the next Chapter, it is included in Section A.5 of the Appendix.

## A.3 Convexity Results

The following results are included here for reference.

**Theorem A.2 (Three-Chord)** *Suppose that a function* $f : \mathbb{R} \to \mathbb{R}$ *is convex on an interval* $I = [a, b] \subseteq \mathbb{R}$. *Let* $x < y < z \in \mathbb{R}$. *Then (see Fig. 23)*

$$\frac{f(y) - f(x)}{y - x} \le \frac{f(z) - f(x)}{z - x} \le \frac{f(z) - f(y)}{z - y}. \tag{A.1}$$

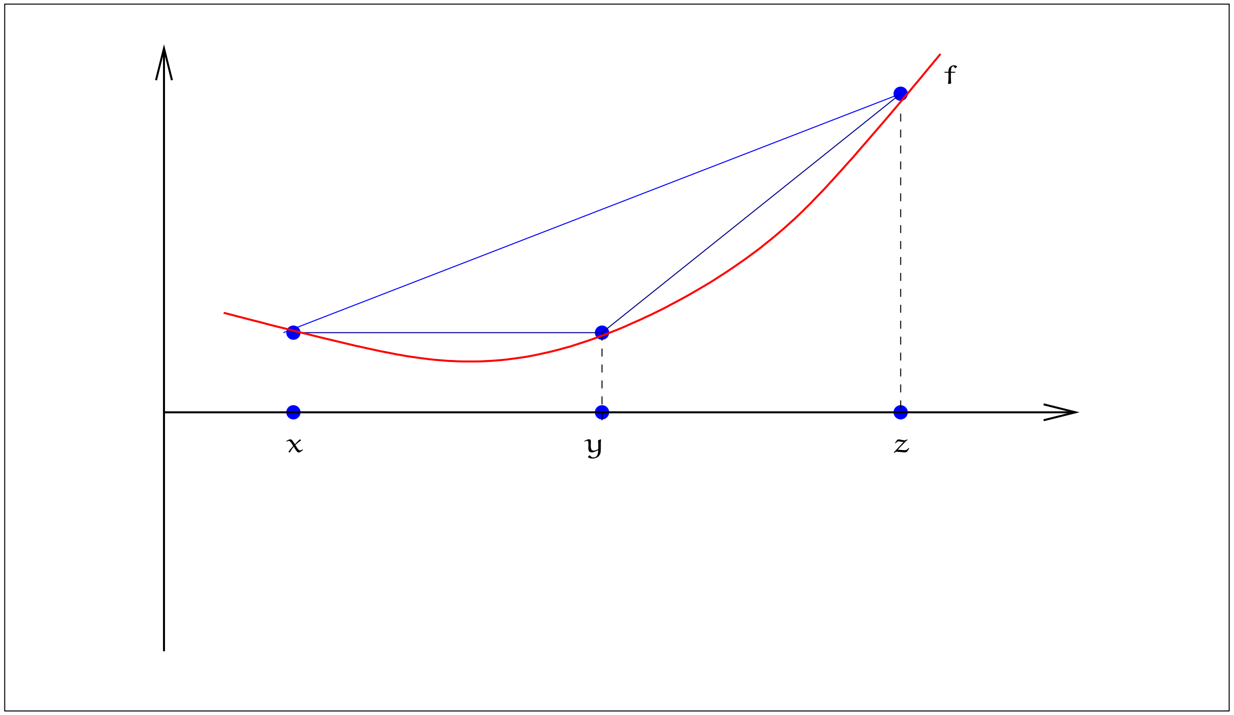


Figure 23: The 3-chord property

**Proof:** We have $x < y < z$ so $\exists t \in (0,1)$ s.t. $y = x + t(z - x)$ for some $t \in (0,1)$. So (using convexity of $f$ in the second line)

$$\begin{aligned} f(y) - f(x) &= f(tz + (1-t)x) - f(x) \\ &\le tf(z) + (1-t)f(x) - f(x) \\ &= t(f(z) - f(x)). \end{aligned}$$

Dividing across by $t(z - x)$ gives us

$$\frac{f(y) - f(x)}{t(z-x)} \le \frac{f(z) - f(x)}{(z-x)}.$$

Noting that $t(z - x) = y - x$ gives the first inequality in (A.1).

Similarly

$$\begin{aligned} f(z) - f(y) &= f(z) - f(tz + (1-t)x) \\ &\ge f(z) - tf(z) - (1-t)f(x) \\ &= (1-t)(f(z) - f(x)). \end{aligned}$$

So dividing across by $(1-t)(z-x)$ g

$$\frac{f(z)-f(y)}{(1-t)(z-x)} \geq \frac{f(z)-f(x)}{(z-x)}.$$

But $z-y=z-(tz+(1-t)x)=(1-t)(z-x)$ which gives us the second inequality in (A.1). ■

We want to show that the slope $f'(x)$ (when it exists) is non-decreasing. With this end in mind we will first prove that the "secant slope" corresponding to a convex function $f$ defined by $s(w;\ c) = \frac{f(w)-f(c)}{(w-c)}$; $(w \neq c, w \in I = [a, b] \subseteq \mathbb{R})$ is non-decreasing as a function of $w$ — in other words; the slope of the secant joining $(c, f(c))$ and $(w, f(w))$ increases as $w$ increases.

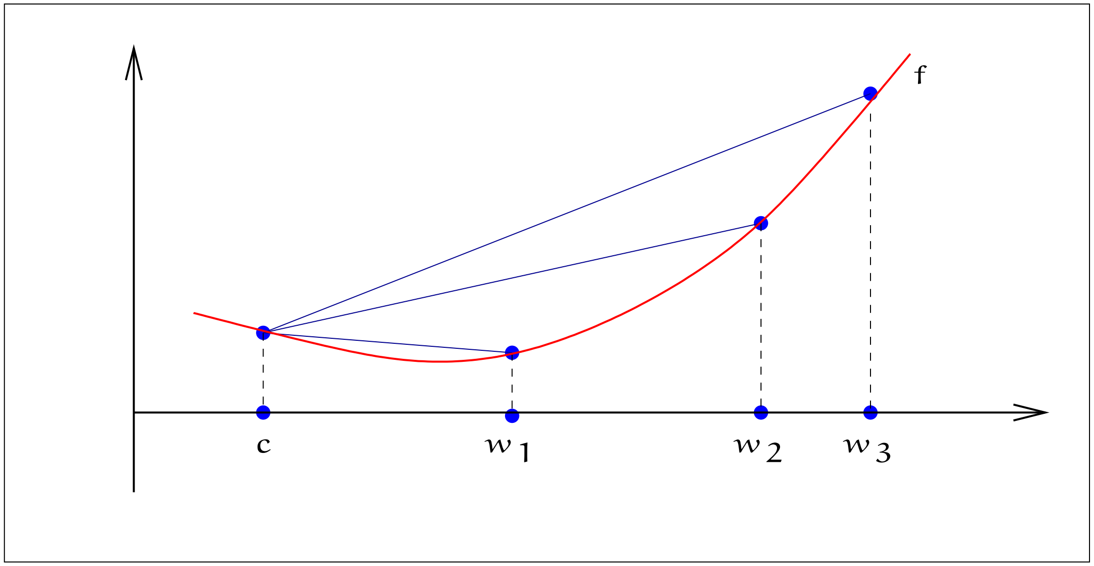


Figure 24: Non-decreasing secant slopes

**Theorem A.3** *Let* $f$ *be convex on* $I = [a, b] \subseteq \mathbb{R}$*. Then for each* $c \in I$*; the secant slope* $s(w;\ c)$ *is a non-decreasing function of* $w$*.*

**Proof:** RTP that if $w_1 < w_2$ then $s(w_1;\ c) \leq s(w_2;\ c)$.

$$\text{Either} \quad \begin{cases} \text{Case 1} & c < w_1 < w_2, \\ \text{Case 2} & w_1 < w_2 < c, \quad \text{or} \\ \text{Case 3} & w_1 < c < w_2. \end{cases}$$

Just use the two inequalities from Theorem A.2.

**[Case 1]** $\frac{f(w_1)-f(c)}{w_1-c} \leq \frac{f(w_2)-f(c)}{w_2-c}$

**[Case 2]** $\frac{f(c)-f(w_1)}{c-w_1} \leq \frac{f(c)-f(w_2)}{c-w_2}$ (reverse signs of numerators and denominators to get result)

**[Case 3]** $\frac{f(c)-f(w_1)}{c-w_1} \leq \frac{f(w_2)-f(c)}{w_2-c}$ (reverse signs of numerators and denominators to get result.)

■

Informally; we need to show that increasing functions always have "limits from the right and left" — see the Figure.

**Lemma A.4** *If a function* $f$ *is non-decreasing on an interval* $I = [a, b] \subseteq \mathbb{R}$ *then* $\lim_{s \to x_+} f(s) = R$ *at all points* $x$ *such that* $a \le x < b$ — *(and similarly* $\lim_{s \to x_-} f(s) = L$ *for* $a < x \le b$ *).*

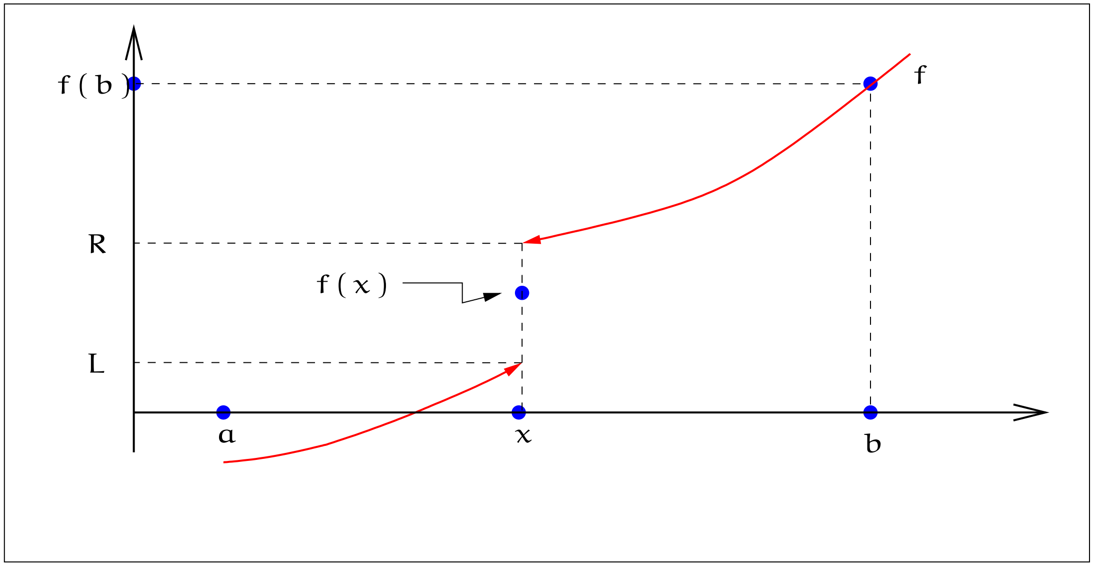


Figure 25: Limit from the right for $f$ non-decreasing

**Proof:** See Appendix A.4. ■

Now we can prove our main result — the left and right-hand derivatives of a convex function are non-decreasing:

**Theorem A.5** *Let* $f$ *be convex on* $I = [a, b] \subseteq \mathbb{R}$. *Then for each* $c < d \in I$*;*

$$f'_{-}(c) \le f'_{+}(c) \le f'_{-}(d) \le f'_{+}(d), \tag{A.2}$$

*where*

$$f'_{-}(c) = \lim_{h \to 0-} \frac{f(c+h) - f(c)}{h} \quad \textit{the left-hand derivative} \tag{A.3}$$

$$f'_{+}(c) = \lim_{h \to 0+} \frac{f(c+h) - f(c)}{h} \quad \textit{the right-hand derivative.} \tag{A.4}$$

(When the left- and right-hand derivatives at a point are equal then of course the function is differentiable at that point.)

**Proof:** First of all: we know (for any $c \in (a, b)$ and for any $x \in I = [a, b]$) that the secant slope $s(x;\ c)$ is a non-decreasing function of $x$ by Theorem A.3. By Lemma A.4 we know that the left and right-hand limits of $s(x;\ c)$ as $x \to c$ exist — these are $f'_-(c)$ and $f'_+(c)$ respectively. It follows from the definitions of left and right-hand limits (and the fact that the secant slope $s(x;\ c)$ is a non-decreasing function of $x$) that $f'_-(c) \leq f'_+(c)$.

Now let $c < d$ be two points in $(a, b)$ and let $h > 0$ be such that $c < d - h$ and $c + h < d$. Then by the 3-chord Theorem A.2,

$$\frac{f(c+h) - f(c)}{h} \leq \frac{f(d) - f(c)}{d - c} \leq \frac{f(d-h) - f(d)}{-h} \equiv \frac{f(d) - f(d-h)}{h}.$$

Now let $h \to 0^+$ in the first and third terms – we find that $f'_+(c) \leq f'_-(d)$. Combining this with $f'_-(c) \leq f'_+(c)$ for any $c \in (a, b)$ we have that $f'_-(c) \leq f'_+(c) \leq f'_-(d) \leq f'_+(d)$. ■

We noted above that when the left- and right-hand derivatives at a point are equal then of course the function is differentiable at that point — in particular it follows from Theorem A.5 that if a convex function is differentiable then its derivative is non-decreasing.

The first half of our final result follows immediately — because of its importance we state it as a Theorem.

**Theorem A.6** *Suppose that a function* $f : I = [a, b] \subseteq \mathbb{R} \to \mathbb{R}$ *is convex and twice differentiable on* $I$*. Then* $f''(x) \geq 0$ *for all* $x$ *in* $[a, b]$*.*

**Proof:** It is a standard result from first-year calculus that $g$ non-decreasing iff $g' \geq 0$. Apply this to $g = f'$. ■

So we have that $f$ convex and twice differentiable on $I$ implies that $f''(x) \geq 0$ for all $x$ in $I$. To prove the converse ( $f''(x) \geq 0$ for all $x$ in $I$ implies that $f$ is convex) is not so straightforward.

In fact we prove this as one half of Theorem A.8 which states that $f''$ non-negative **and** continuous on $I \subseteq \mathbb{R}$ implies that $f$ is convex on $I$.

But the first part of the Theorem does not require that $f''$ be continuous so we do in fact have the important result:

**Theorem A.7 (Convexity/Second Derivative)** *Suppose that a function* $f : I = [a, b] \subseteq \mathbb{R} \to \mathbb{R}$ *is twice differentiable on* $I = [a, b] \subseteq \mathbb{R}$. *Then* $f''(x) \geq 0$ *for all* $x$ *in* $[a, b]$ *iff* $f$ *is convex on* $I$.

**Proof:** Follows from Thm A.6 and the first half of Thm A.8. ■

The following Theorem is more limited than Theorem A.7 as it requires that $f$ be $C^2$ but as noted above the first half of the proof does not in fact use this property.

**Theorem A.8** *Suppose that a function $f : \mathbb{R} \to \mathbb{R}$ has continuous second derivative on $\mathbb{R}$. Then $f$ is convex on an interval $[a, b]$ if and only if $f''(x) \geq 0$ for all $x$ in $[a, b]$.*

(We have already proved the second half of the Theorem in Theorem A.6 above. )

**Proof:** For any $x$ and $y$ in $[a, b]$ define

$$F_{xy}(\alpha) = \alpha f(x) + (1-\alpha)f(y) - f(\alpha x + (1-\alpha)y), \quad \alpha \in [0, 1].$$

Obviously $f$ is convex on $[a, b]$ if and only if for any interval $[x, y] \subseteq [a, b]$, $F_{xy}(\alpha) \geq 0$ for $0 \leq \alpha \leq 1$. It is easy to check that

$$F_{xy}{}'(\alpha) = (f(x) - f(y)) - (x - y)f'(\alpha x + (1-\alpha)y)$$

and that

$$F_{xy}{}''(\alpha) = -(x-y)^2 f''(\alpha x + (1-\alpha)y).$$

It follows immediately that $f''(z)$ and $F_{xy}{}''(\alpha)$ have opposite signs where $z = \alpha x + (1-\alpha)y$ — we can ignore the case $x = y$.

1. First show that $f''$ non-negative ($\equiv {F_{xy}}''(\alpha) \leq 0$) on $[a, b]$ implies that $f$ is convex on $[a, b]$, or equivalently that $F_{xy}(\alpha) \geq 0$ for any interval $[x, y] \subseteq [a, b]$.

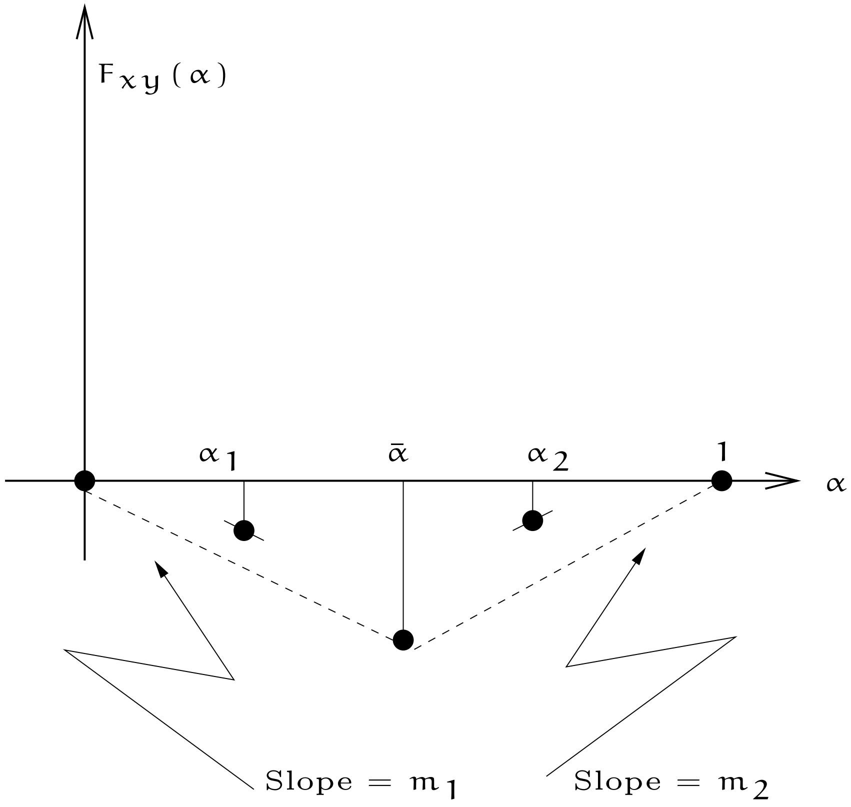


Figure 26: $f'' \geq 0$ implies $f$ convex

Suppose that the result is false. Then for some $x, y, \in \mathbb{R}$, with $x < y$, there exists $\bar{\alpha}, \quad 0 < \bar{\alpha} < 1$ such that $F_{xy}(\bar{\alpha}) < 0$.

Now refer to Fig. 26. Let

$$\begin{aligned} m_1 &= \frac{F_{xy}(\bar{\alpha}) - F_{xy}(0)}{\bar{\alpha} - 0} = \frac{F_{xy}(\bar{\alpha})}{\bar{\alpha}} < 0 \\ m_2 &= \frac{F_{xy}(1) - F_{xy}(\bar{\alpha})}{1 - \bar{\alpha}} = \frac{-F_{xy}(\bar{\alpha})}{1 - \bar{\alpha}} > 0 \end{aligned}$$

By the Mean Value Theorem, $\exists \alpha_1 \in (0, \bar{\alpha})$ such that $F'(\alpha_1) = m_1$ and $\alpha_2 \in (\bar{\alpha}, 1)$ such that $F'(\alpha_2) = m_2$. But $m_1 < m_2$ — this contradicts the assumption that the slope of $F_{xy}$ is decreasing (${F_{xy}}'' \leq 0$ on $(0, 1)$).

So $f''$ non-negative on $[a, b]$ implies that $f$ is convex on $[a, b]$ — as required.

$\square \leftarrow$

## A.4 Proof of Lemma A.4

The formal definition of $\lim_{s \to x_+} f(s) = R$ is

$$\exists R \quad \text{such that} \quad \forall \varepsilon > 0, \exists \delta > 0 \quad \text{such that} \\ \forall s, 0 < s - x < \delta \Rightarrow |f(s) - R| < \varepsilon. \qquad \text{(A.5)}$$

Let $\mathcal{F} = \{f(s) | x < s \leq b\}$, referring to Figure 25, define $R = \text{glb}\mathcal{F}$ — the **greatest lower bound** of the set $\mathcal{F}$.

The glb of any set $\mathcal{G}$ is just the largest number $b$ such that $b \leq g$ for all $g \in \mathcal{G}$. It is a fundamental property of the real numbers $\mathbb{R}$ that any set $\mathcal{F}$ of real numbers that is bounded below has a glb. The set $\mathcal{F}$ is bounded below by $f(x)$ (as $f$ is non-decreasing) so is must have a glb.

Note that the glb is not necessarily **in** $\mathcal{F}$ — in Figure 25, the glb of $\mathcal{F}$ is $R$ but $\mathcal{F} = (R, f(b)]$ as $f$ is non-decreasing.

Back to the proof; assume that $R$ is **not** the RH limit of $f(s)$ at $x$ – i.e. that (A.5) is **false**.

So

$$\exists \epsilon > 0 \quad \text{such that} \quad \forall \delta > 0, \quad \exists s_0 \quad \text{such that}$$
$$0 < s_0 - x < \delta \quad \textbf{and} \quad |f(s_0) - R| \geq \epsilon. \quad (A.6)$$

We therefore have $x < s_0 < x + \delta$ and $f(s_0) \geq R + \epsilon$ as $R$ is a lower bound for $\mathcal{F}$ and $f(s_0) \in \mathcal{F}$. As $f$ is non-decreasing this implies that $f(x + \delta) \geq R + \epsilon$ for **all** $\delta > 0$.

As $\delta > 0$ is arbitrary this means that $R + \epsilon$ is a lower bound for $\mathcal{F}$ which contradicts the definition of $R$ as the **greatest** lower bound of $\mathcal{F}$. ■

Now, prove that $f$ convex on an interval $[a, b]$ implies $f'' \geq 0$ on $[a, b]$. We use a proof by contradiction. Assume that $f$ is convex on an interval $[a, b]$ and that $f''(x_0) < 0$ at some $x_0 \in [a, b]$. By continuity, $f''$ remains negative in some interval containing $x_0$, say $[x_1, x_2]$. As above, define

$$F_{x_1 x_2}(\alpha) = \alpha f(x_1) + (1 - \alpha) f(x_2) - f(\alpha x_1 + (1 - \alpha) x_2), \quad \alpha \in [0, 1].$$

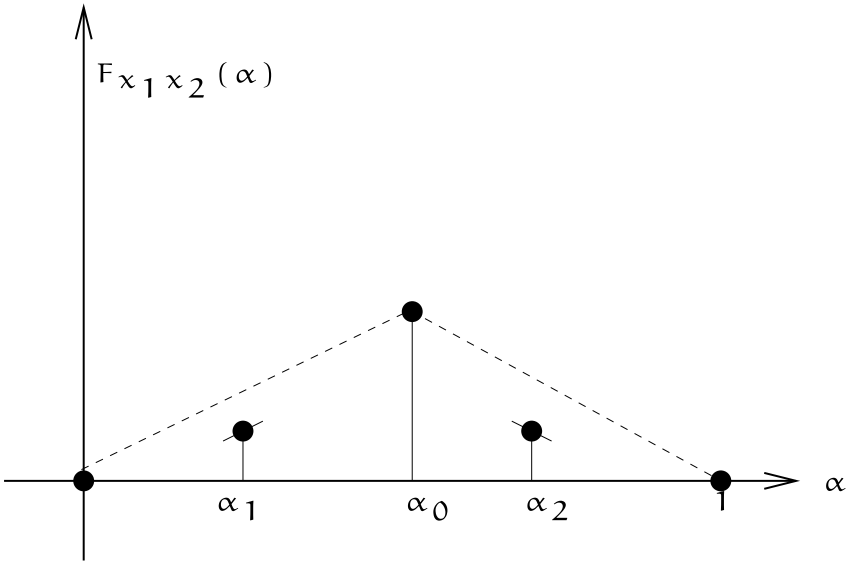


Figure 27: $f$ convex implies $f'' < 0$

Then ${F_{x_1x_2}}''(\alpha)$ is positive for $\alpha \in [0, 1]$ and of course convexity gives us $F_{x_1x_2}(\alpha) \geq 0$ for $0 \leq \alpha \leq 1$. Obviously $F_{x_1x_2}(0) = F_{x_1x_2}(1) = 0$. Now $F_{x_1x_2}$ cannot be identically zero on $[0, 1]$ as this contradicts ${F_{x_1x_2}}'' > 0$. So pick some $\alpha_0$ such that $F_{x_1x_2}(\alpha_0) > 0$. (See Figure 27.)

Now use an argument similar to that in part (1) of the proof to show that there is a point $\alpha_1$ in $[0, \alpha_0]$ and a point $\alpha_2$ in $[\alpha_0, 1]$ such that ${F_{x_1x_2}}'(\alpha_1) > 0$ and ${F_{x_1x_2}}'(\alpha_2) < 0$. But this contradicts ${F_{x_1x_2}}''(\alpha) > 0$. ■

## A.5 Proof of Theorem A.1

1. Assume that $f$ is convex. RTP $\nabla^2 f(x)$ is positive semi-definite for all $x \in \mathbb{R}^n$. Use a proof by contradiction: assume the conclusion is false so there exist $x, z \in \mathbb{R}^n$ such that $z^T \nabla^2 f(x) z < 0$.

Define $\phi(\alpha) = f(\alpha x + (1-\alpha)y)$, where $y = x + z$. Let $x_\alpha = \alpha x + (1-\alpha)y$. Then $\phi''(\alpha) = z^T \nabla^2 f(x_\alpha) z$ and so $\phi''(1) < 0$. By continuity, $\phi''$ is negative in some neighbourhood of $\alpha = 1$. Clearly, by Thm. A.8, $\phi$ is not convex. There must exist an interval $I = (\alpha_1, \alpha_2)$ such that for $0 < \beta < 1$;

$$\phi(\beta\alpha_1 + (1-\beta)\alpha_2) > \beta\phi(\alpha_1) + (1-\beta)\phi(\alpha_2) \tag{A.7}$$

Now, let $x_1 = \alpha_1 x + (1-\alpha_1)y$ and $x_2 = \alpha_2 x + (1-\alpha_2)y$. The LHS of Eq. A.7 is just

$$f([\beta\alpha_1 + (1-\beta)\alpha_2]\, x + (1 - [\beta\alpha_1 + (1-\beta)\alpha_2])\, y)$$

and it is easy to check that

$$[\beta\alpha_1 + (1-\beta)\alpha_2]\, x + (1 - [\beta\alpha_1 + (1-\beta)\alpha_2])\, y = \beta x_1 + (1-\beta) x_2.$$

Also, $\phi(\alpha_1) = f(x_1)$ and $\phi(\alpha_2) = f(x_2)$. We therefore have

$$f(\beta x_1 + (1-\beta)x_2) > \beta f(x_1) + (1-\beta)f(x_2),$$

contradicting the assumption that $f$ is convex.

2. Assume that $\nabla^2 f(x)$ is positive semi-definite for all $x \in \mathbb{R}^n$. RTP that $f$ is convex. Assume not. Then there exist $x, y \in \mathbb{R}^n$ such that

$$f(\alpha_0 x + (1-\alpha_0)y) > \alpha_0 f(x) + (1-\alpha_0)f(y) \tag{A.8}$$

for some $\alpha_0 \in (0,1)$.

Let $\phi(\alpha) = f(\alpha x + (1-\alpha)y)$. Then as before, $\phi''(\alpha) = z^T \nabla^2 f(x) z$. As $\phi''(\alpha) \geq 0$ for all $\alpha$, we have that $\phi$ is convex. Clearly $\phi(1) = f(x)$ and $\phi(0) = f(y)$ so

$$\phi(\alpha 1 + (1-\alpha)0) \leq \alpha\phi(1) + (1-\alpha)\phi(0)$$

which leads immediately to

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

contradicting Eq. A.8.

$\blacksquare$

## A.6 Stronger Proof of First-Order Necessary Conditions

We can drop the requirement that the gradient is continuous by basing the proof on the corresponding resut for real-valued functions.

**Theorem A.9 (Real Line)** *If a real-valued function* $f(x)$ *is differentiable at* $x = x_0$ *and has a local extremum at this point, then* $f'(x_0) = 0$.

**Proof:** If $f'(x_0) > 0$ $( < 0)$, then $f$ is increasing (decreasing) at this point by Theorem.... As $f$ has an extremum at $x_0$ (so is neither increasing nor decreasing), so $f'(x_0) = 0$. ■

**Theorem A.10** *If a function* $f(x_1, x_2, ...x_n)$ *is differentiable at* $x = (a_1, a_2, ...a_n)$ *and has a local extremum at this point, then* $\frac{\partial f}{\partial x_k}(a) = 0$ *for* $k = 1, \dots, n$ *or equivalently* $\nabla f(a) = 0$.

**Proof:** The hypotheses of the theorem imply that the function

$$g(x_1) = f(x_1, a_2, ...a_n)$$

(with frozen variables $x_k = a_k$ for $k = 2, \ldots, n$) has a local extremum at $x_1 = a_1$ and is differentiable at this point. Hence, by Theorem A.9, $g_1'(a_1) = 0$, which is equivalent to

$$\frac{\partial f}{\partial x_1}(a) = 0.$$

So we obtained the desired assertion for $k = 1$. The cases of $k > 1$ are considered similarly. ■

## A.7 Why is $-\nabla f$ Perpendicular to the Tangent to Contour?

One answer — for any direction $p$;

- we have $f(x_0 + \alpha p) - f(x_0) = \alpha p^T \nabla f(x_0) + O(\alpha^2)$.
- If $p^T \nabla f(x_0) = 0$ then $f(x_0 + \alpha p) - f(x_0) = O(\alpha^2)$ and so $p \perp \nabla f(x_0)$ precisely when $p$ is a "stationary direction" (one along which $f$ is constant to first order).
- In other words stationary directions (first-order infinitesimal steps "along the contour") are $\perp$ to $\nabla f(x_0)$.
- Or just $\nabla f(x_0)$ is $\perp$ to the tangent to the contour.

Back to Slide 78.

## A.8 Hessian With Uniformly Bounded Norm Implies Lipschitz Continuity

Taylor's Thm. 3.7 is (for any $x$ and any $p$):

$$\nabla f(x+p) = \nabla f(x) + \int_0^1 \nabla^2 f(x+tp)\, p\, dt$$

so

$$\begin{aligned}
\|\nabla f(x+p) - \nabla f(x)\| &= \|\int_0^1 \nabla^2 f(x+tp)\, p\, dt\| \\
&\leq \int \|\nabla^2 f(x+tp)\, p\| dt \\
&\leq \int \|\nabla^2 f(x+tp)\| \|p\| dt \\
&\leq M\|p\|
\end{aligned}$$

So $\nabla f$ is Lipschitz continuous. Back to Slide 113.

## A.9 Proof of Theorem 4.2

If at least one iteration has taken place, then for some $\alpha'$;

$$\begin{aligned} \phi(\alpha') &< \phi(0) + c\alpha'\phi'(0) \qquad \text{(A.9)} \\ \phi(\frac{\alpha'}{\rho}) &\geq \phi(0) + \frac{c\alpha'}{\rho}\phi'(0). \qquad \text{(A.10)} \end{aligned}$$

Clearly inequality A.9 means that the first Wolfe condition Eq. 4.9a is satisfied at $\alpha'$ (and, by continuity, near $\alpha'$) with $c_1 = c$. Set $c = c_1$ in the following.

Now subtract Eq. A.9 from Eq. A.10 (keeping track of the signs of the inequalities)

$$\phi(\frac{\alpha'}{\rho}) - \phi(\alpha') > c_1 \alpha' \left(\frac{1}{\rho} - 1\right) \phi'(0). \tag{A.11}$$

By the Mean Value Theorem, the lefthand side of Eq. A.11 can be written as $\alpha' \left(\frac{1}{\rho} - 1\right) \phi'(\alpha'')$ where $\alpha' < \alpha'' < \frac{\alpha'}{\rho}$ and so (as $0 < \rho < 1$);

$$\phi'(\alpha'') > c_1 \phi'(0) \tag{A.12}$$

Now, $\phi'(0) < 0$ as $p_k$ is a descent direction — so for any $c_2 > c_1$ we have

$$\phi'(\alpha'') > c_2\phi'(0), \tag{A.13}$$

so the second Wolfe condition eq. 4.9b is satisfied at $\alpha''$. Again by continuity of $\phi'$ it must hold near $\alpha''$. Finally, for $\rho$ close enough to $1$, $\alpha' < \alpha'' < \frac{\alpha'}{\rho}$ means that $|\alpha'' - \alpha'|$ can be made as small as we like, so that the two Wolfe conditions must hold in the interval $(\alpha', \alpha'')$, as required. ■

## A.10 Solution to Question 4 on Slide 150

- We have $\cos\theta = \dfrac{-p^T g}{\|p\|\|g\|}$.
- Use $p = -B^{-1}g$ to write $\cos\theta$ as $\cos\theta = \dfrac{g^T B^{-1} g}{\|B^{-1}g\|\|g\|}$.
- We also have $\|B\| = |\lambda_{\max}|$ and $\|B^{-1}\| = 1/|\lambda_{\min}|$.
- As we are taking B to be positive definite we can drop the magnitude symbols.
- We can expand the TL in $\cos\theta$ as $\cos\theta = \sum g_i^2/\lambda_i$ where the scalars $g_i$ are just the components of $g$ along the eigenvectors of B, taken to be unit vectors.
- Therefore TL $\geq \frac{1}{\lambda_{\max}}\|g\|^2 = \frac{\|g\|^2}{\|B\|}$.
- But in the bottom line, $\|B^{-1}g\| \leq \|B^{-1}\|\|g\|$ so we have $\cos\theta \geq \frac{1}{\|B\|\|B^{-1}\|} \geq \frac{1}{M}$. ■

## A.11 Proof of Question 6 on Slide 151 — the Kantorovitch inequality

R.T.P. that

$$\frac{(x^T x)^2}{(x^T Q x)(x^T Q^{-1} x)} \geq \frac{4\lambda_{\min}\lambda_{\max}}{(\lambda_{\min} + \lambda_{\max})^2}, \tag{A.14}$$

where $\lambda_{\min}$ is the smallest eigenvalue and $\lambda_{\max}$ is the largest. (As Q is positive definite , we can write:

$$0 < \lambda_{\min} \leq \lambda_i \leq \lambda_{\max}, \quad i = 1 \dots n. \tag{A.15}$$

As Q is a real symmetric matrix, it can be written as $Q = U \Lambda U^T$ where $\Lambda$ is the diagonal matrix of eigenvalues $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$ and $U^T U = I$. Writing $y = U^t x$, the required inequality can be re-written as:

$$\frac{(y^T y)^2}{(y^T \Lambda y)(y^T \Lambda^{-1} y)} \geq \frac{4\lambda_{\min}\lambda_{\max}}{(\lambda_{\min} + \lambda_{\max})^2}. \tag{A.16}$$

Now, as $\Lambda$ is diagonal and as $y^T y$ can be taken to be $1$ (we can divide above and below by $\|y\|^2$ if necessary), we can re-write the inequality to be proved as:

$$\sum_{i=1}^{n} Y_i \lambda_i \sum_{i=1}^{n} \frac{Y_i}{\lambda_i} \leq \frac{(\lambda_{\min} + \lambda_{\max})^2}{4\lambda_{\min}\lambda_{\max}}, \tag{A.17}$$

where $Y_i = y_i^2$ and $\sum_{i=1}^{n} Y_i = 1$. Now, set $f(\lambda_1, \ldots, \lambda_n)$ to be the left hand side of A.17. The second derivative of $f$ with respect to each of the $\lambda_i$:

$$\frac{d^2 f(\lambda_1, \ldots, \lambda_n)}{d\lambda_k^2} = 2\frac{Y_k}{\lambda_k^3}\left(\sum_{i=1}^{n} Y_i \lambda_i - \lambda_k Y_k\right)$$

is non-negative and so $f$ is a convex function of each of the $\lambda_i$ separately.

This means that $f$ takes its maximum value for each $\lambda_i$ at one of the end-points of its range of values, i.e. at $\lambda_i = \lambda_{\min}$ or at $\lambda_i = \lambda_{\max}$.

Why?

Letting $Y = \sum_{i:\lambda_i=\lambda_{\min}} Y_i$, it follows that

$$f(\lambda_1, \dots, \lambda_n) \le (Y\lambda_{\min} + (1-Y)\lambda_{\max}) \left( \frac{Y}{\lambda_{\min}} + \frac{(1-Y)}{\lambda_{\max}} \right) = f^* \quad \text{say.} \tag{A.18}$$

Now, multiplying out the right hand side of A.18 and setting $z = Y(1-Y)$, we find

$$f^* = 1 + z \left( \frac{\lambda_{\max}}{\lambda_{\min}} + \frac{\lambda_{\min}}{\lambda_{\max}} - 2 \right). \tag{A.19}$$

As the coefficient of $z$ is positive (check) and using the simple property of quadratics that $Y(1-Y) \leq 1/4$, we have

$$f(\lambda_1, \ldots, \lambda_n) \leq 1 + \frac{1}{4}\left(\frac{\lambda_{\max}}{\lambda_{\min}} + \frac{\lambda_{\min}}{\lambda_{\max}} - 2\right) = \frac{(\lambda_{\min} + \lambda_{\max})^2}{4\lambda_{\min}\lambda_{\max}}, \tag{A.20}$$

which is the reciprocal of A.16. ■

## A.12 Solution to Question 8 on Slide 151

We can always assume that a quadratic function $f(x)$ has its minimum at $x^* = 0$ and that $f(x^*) = 0$ — as if not we can use the change of variables $y = x - x^*$ transforms $f(x)$ to $\bar{f}(y) = \frac{1}{2}x^T Qx$ apart from a constant which can be ignored.

So we have

$$f(x) = \frac{1}{2}x^T Qx, \quad \nabla f(x) = Qx, \quad \nabla^2 f(x) = Q.$$

The steepest descent update takes the form

$$x_{k+1} = x_k - \alpha_k \nabla fk = (I - \alpha_k Q)x_k.$$

It follows that

$$\|x_{k+1}\|^2 = {x_k}^T (I - \alpha_k Q)^2 x_k.$$

For any quadratic form $x^T Ax$, we have $x^T Ax \leq \lambda_{max} \|x\|^2$ where $\lambda_{max}$ is the largest eigenvalue of $A$; so

$$\|x_{k+1}\|^2 = {x_k}^T (I - \alpha_k Q)^2 x_k \leq \left(\text{largest eval of } (I - \alpha_k Q)^2\right) \|x_k\|^2.$$

Let the eigenvalues of $Q$ be $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$. Then the eigenvalues of $(I - \alpha_k Q)^2$ are just $(1 - \alpha_k \lambda_k)^2$ so $\Lambda$ the largest eigenvalue of $(I - \alpha_k Q)^2$, is just:

$$\Lambda = \max \left\{(1 - \alpha_k \lambda_1)^2, (1 - \alpha_k \lambda_n)^2\right\}.$$

We now have for $x_k \neq 0$ that

$$\frac{\|x_{k+1}\|}{\|x_k\|} \leq \max \left\{(1 - \alpha_k \lambda_1)^2, (1 - \alpha_k \lambda_n)^2\right\}.$$

Fig. 28 on the next Slide illustrates the convergence rate bounds as a function of the stepsize $\alpha$.

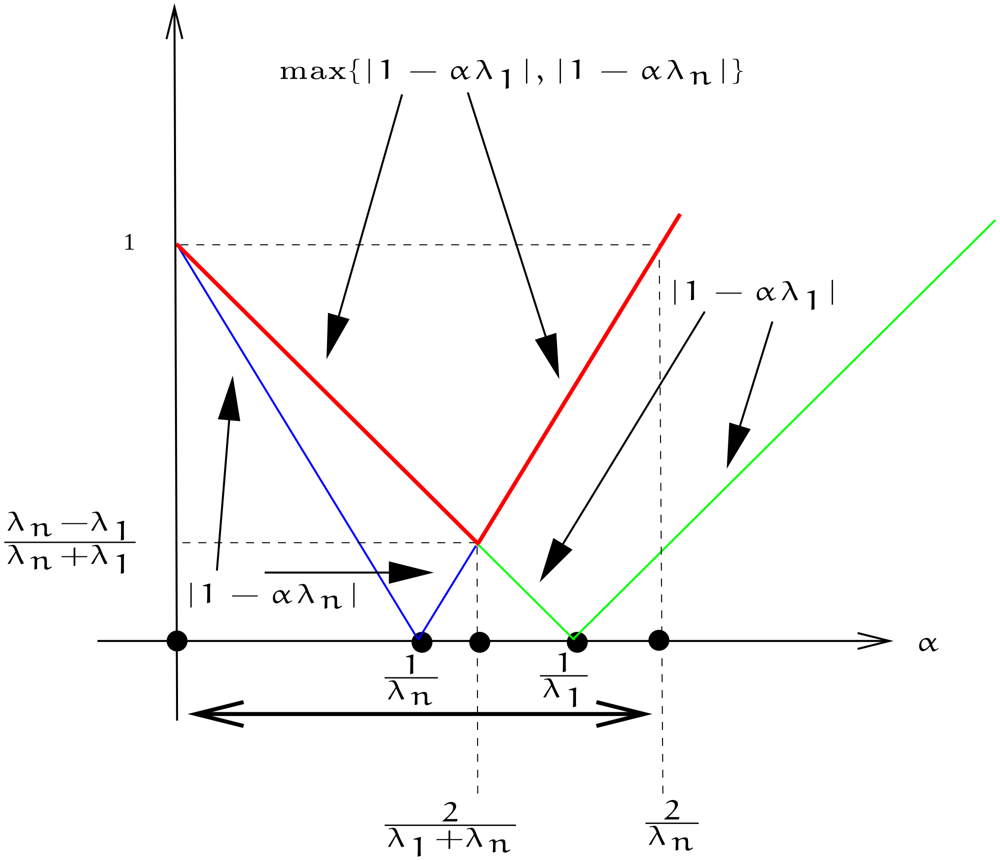

$\max\{|1-\alpha\lambda_1|,|1-\alpha\lambda_n|\}$
1
$|1-\alpha\lambda_1|$
$\frac{\lambda_n-\lambda_1}{\lambda_n+\lambda_1}$
$|1-\alpha\lambda_n|$
$\alpha$
$\frac{1}{\lambda_n}$
$\frac{1}{\lambda_1}$
$\frac{2}{\lambda_1+\lambda_n}$
$\frac{2}{\lambda_n}$

Figure 28: Convergence rate bound for steepest descent.

From the diagram (and after a little algebra), the value of $\alpha_k$ that minimises the bound is $\alpha_* = \frac{2}{\lambda_1+\lambda_n}$.

We therefore have the required result:

$$\frac{\|x_{k+1}\|}{\|x_k\|} \le \frac{\lambda_n - \lambda_1}{\lambda_1 + \lambda_n}.$$

We have recovered Eq. 4.30 as required. Note that the result holds for **any** quadratic function $\frac{1}{2}x^T Q x - b^T x$ using the change of variables mentioned above. Also we have used the conventional Euclidean norm rather than the weighted norm
$\|x\|_Q^2 \equiv x^T Q x$. ■

## A.13 Step-Length Selection Algorithms — an Introduction

We now consider practical algorithms for minimising the one-dimensional function 4.3 — repeated here for convenience.

$$\phi(\alpha) = f(x_k + \alpha p_k), \quad \alpha > 0, \tag{A.21}$$

or (typically) finding a step length $\alpha_k$ satisfying the Wolfe conditions . We assume that $p_k$ is a descent direction ($\phi'(0) < 0$) so the search can be limited to positive values of $\alpha$.

All line search algorithms need an initial estimate $\alpha_0$ and generate a sequence $\{\alpha_i\}$ that either terminates with a step length satisfying the Wolfe conditions or determines that such a step length does not exist. Usually (as mentioned in Section 4.2) we have a **bracketing phase** that finds an interval $[a, b]$ containing acceptable step lengths and a **refinement phase** that zooms in to find the final step length.

### A.13.1 Some Technical Results

In this subsection we develop some results which will be useful in proving convergence of the linesearch algorithm to be described in Sec A.14. This section may be skipped on first reading and referred to later.

First, we will define an auxiliary function $\Phi$ — a function that allows us to prove results about the function $\phi$. Properties of $\phi$ such as the Wolfe conditions can be expressed conveniently in terms of the signs of $\Phi$ and its derivative.

**Definition A.5** *Once the constant $c_1$ is specified, define the function $\Phi$ by*

$$\Phi(\alpha) = \phi(\alpha) - [\phi(0) + c_1 \alpha \phi'(0)] \tag{A.22}$$

It is easy to check that the following properties follow from the definition of $\Phi$. We state this as a Theorem for convenience.

**Theorem A.11** *Once $\Phi$ is defined as in Def A.5 the following properties hold:*

1. *$\Phi(\alpha) \leq 0$ is equivalent to the First Wolfe condition 4.9a holding at $\alpha$,*
2. *$\Phi'(\alpha) \geq 0$ implies the (weak form of the ) Second Wolfe condition 4.9b holds at $\alpha$,*
3. *$\Phi'(\alpha) = 0$ (i.e. $\alpha$ a local minimum of $\Phi$) implies that the* **Strong** *version of the Second Wolfe condition holds.*
4. *$(\alpha_1 < \alpha_2)$ and $\phi(\alpha_1) \leq \phi(\alpha_2)$ implies that $\Phi(\alpha_1) < \Phi(\alpha_2)$,*
5. *$\phi'(\alpha) \geq 0$ implies that $\Phi'(\alpha) > 0$,*
6. *If the* **Strong** *version of the Second Wolfe condition is false at $\alpha$ then $\operatorname{sign}(\Phi'(\alpha)) = \operatorname{sign}(\phi'(\alpha))$.*

**Proof:** We check the properties one-by-one.

1. By definition of $\Phi$ (A.22) and the first Wolfe condition (4.9a).
2. $\Phi'(\alpha) = \phi'(\alpha) - c_1\phi'(0)$. If $\Phi'(\alpha) \geq 0$ clearly $\phi'(\alpha) \geq c_1\phi'(0) \geq c_2\phi'(0)$ so the weak form of the Second Wolfe condition 4.9b holds at $\alpha$.
3. If $\Phi'(\alpha) = 0$ then $\phi'(\alpha) = c_1\phi'(0) > c_2\phi'(0)$ so the weak form of the Second Wolfe condition holds. Also $\phi'(\alpha) = c_1\phi'(0) < 0 < -c_2\phi'(0)$ so the Strong form of the Second Wolfe condition holds.
4. Let $\Delta\Phi = \Phi(\alpha_2) - \Phi(\alpha_1)$, then $\Delta\Phi = \Delta\phi - c_1\phi'(0)\Delta\alpha$. So as $\phi'(0) < 0$, $\Delta\phi \geq 0$ and $\Delta\alpha > 0$ then $\Delta\Phi \geq 0$ as required.

5. The second term in $\Phi'(\alpha)$ is strictly positive so provided the first term $\phi'(\alpha) \geq 0$ the result follows.

6. If the strong form of the Second Wolfe condition (4.10b) is false at $\alpha$ then $|\phi'(\alpha)| > -c_2\phi'(0) > -c_1\phi'(0)$ as $c_1 < c_2$. Again it is clear that if $\phi'(\alpha) > 0$ then so is $\Phi'(\alpha)$. And because the magnitude of $\phi'(\alpha)$ is bigger than the second term in $\Phi'(\alpha)$ it follows that if $\phi'(\alpha) < 0$ then so is $\Phi'(\alpha)$. Of course the case $\phi'(\alpha) = 0$ cannot arise as $|\phi'(\alpha)| > -c_2\phi'(0) > 0$.

   A little thought should convince you that the direction of the implications can be reversed as$(A \Rightarrow B) \equiv (\tilde{B} \Rightarrow \tilde{A})$. ■

We need some results about when the Wolfe conditions hold.

**Theorem A.12** *If*

$$\phi(\alpha_0) > \phi(0) + c_1 \alpha_0 \phi'(0), \tag{A.23}$$

*at some* $\alpha_0 > 0$, *the interval* $[0, \alpha_0]$ *contains* $\alpha$*-values that satisfy the strong Wolfe conditions.*

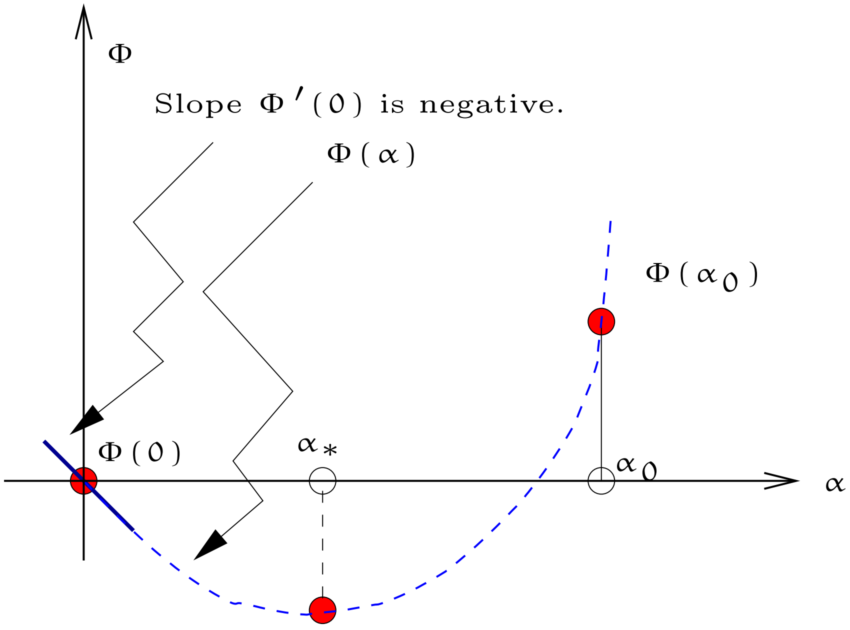


Figure 29: First Wolfe condition fails at $\alpha_0$

**Proof:** See the Figure on the previous Slide. $\Phi(0) = 0$ and $\Phi'(0) = (1 - c_1)\phi'_0 < 0$ so $\Phi$ decreases initially from zero, then increases from a negative to a positive value at $\alpha_0$. So (by Rolle's Theorem — why?) $\Phi$ must have a minimum, say $\alpha_*$ , between $0$ and $\alpha_0$ — where $\Phi' = 0$ and $\Phi$ is negative. Theorem A.11 tell us that the strong Wolfe conditions must hold at $\alpha_*$ . ■

We can extend this result to the following important Theorem:

**Theorem A.13** *If the sufficient decrease condition 4.9a* **holds** *at an $\alpha$-value, say $\alpha_{i-1}$ and* **fails** *at a larger $\alpha$-value, say $\alpha_i$, then there is an $\alpha$-value $\bar{\alpha}$ in the interval $[\alpha_{i-1}, \alpha_i]$ where the strong Wolfe conditions 4.10a and 4.10b hold.*

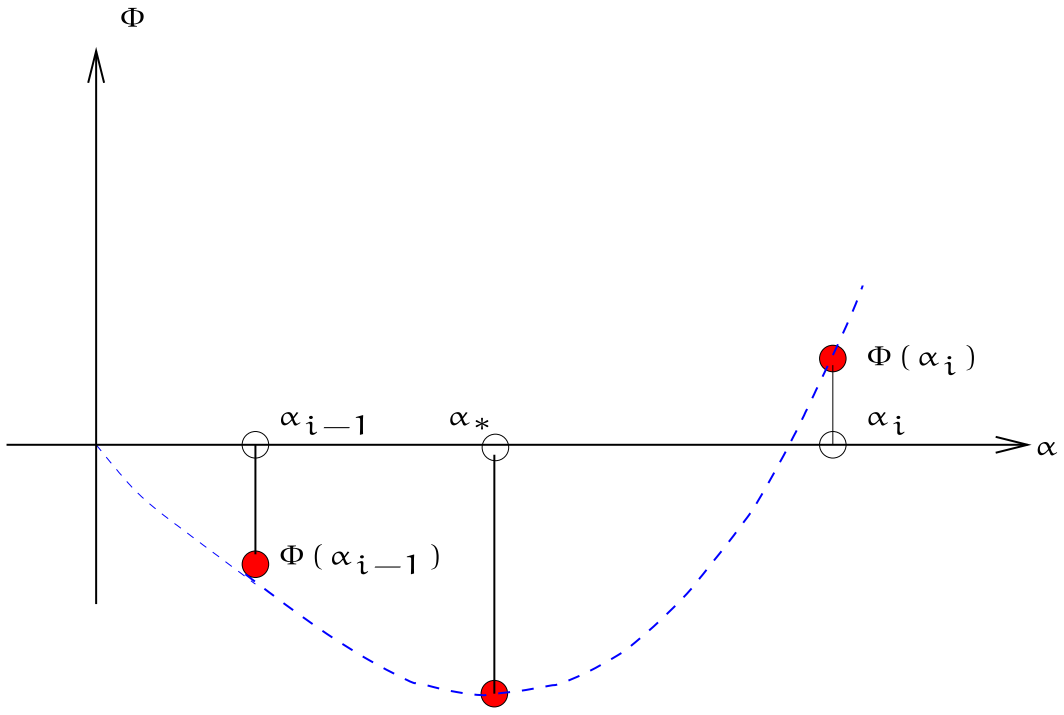


Figure 30: First Wolfe condition true at $\alpha_{i-1}$ and **fails** at $\alpha_i$

**Proof:** See the Figure on the previous Slide. This time $\Phi$ is negative at $\alpha_{i-1}$ and positive at $\alpha_i$ so must have a minimum, say $\alpha_*$ , between $\alpha_{i-1}$ and $\alpha_i$ — where $\Phi' = 0$ and $\Phi$ is negative. (Again, can you see how the result follows from Rolle's Theorem?) Theorem A.11 tells us that the strong Wolfe conditions must hold at $\alpha_*$ . ∎

### A.13.2 Interpolation

In the next few slides we give an informal description of the steps involved as a numbered sequence — the idea of interpolation (essentially using a polynomial fit to current data to produce a candidate for the next point) turns out to be crucial. We begin by describing a line search procedure based on interpolation of function & derivative values of $\phi$. It can be viewed as an improved version of Algorithm 4.1. The aim is to find a value of $\alpha$ that satisfies the sufficient decrease condition Eq. 4.9a without being "too small".

For the sake of clarity, we use $\alpha_k$ and $\alpha_{k-1}$ to denote the step lengths used at iterations $k$ and $k-1$ of the overall optimisation algorithm. We use the letters $i$ and $j$ to index the trial step lengths generated during the line search and refer to the initial guess as $\alpha_0$.

The procedures described here generate a decreasing sequence of values $\alpha_i$ such that each $\alpha_i$ is not much smaller than its predecessor $\alpha_{i-1}$.

Remember that the sufficient decrease condition can be written (in the notation of Eq. 4.3 as

$$\phi(\alpha_k) \leq \phi(0) + c_1 \alpha_k \phi'(0), \tag{A.24}$$

and since the constant $c_1$ is usually chosen small ($\approx 10^{-4}$, say), this condition really only asks for some small reduction in $\phi$. We design the procedure to be efficient in that it computes $\phi'$ as seldom as possible.

1. Suppose that the initial guess $\alpha_0$ is given. If we have
$$\phi(\alpha_0) \le \phi(0) + c_1 \alpha_0 \phi'(0), \tag{A.25}$$
then we can stop. If not — and this is the key to understanding the algorithm— the interval $[0, \alpha_0]$ **must** contain acceptable step lengths by Theorem A.12.

2. We form a quadratic approximation $\phi_q(\alpha)$ to $\phi$ by interpolating the three pieces of information available — $\phi(0)$, $\phi'(0)$ and $\phi(\alpha_0)$ — to obtain
$$\phi_q(\alpha) = \left(\frac{\phi(\alpha_0) - \phi(0) - \alpha_0\phi'(0)}{\alpha_0^2}\right)\alpha^2 + \phi'(0)\alpha + \phi(0). \tag{A.26}$$
(Check that $\phi_q(0) = \phi(0)$, $\phi_q'(0) = \phi'(0)$ and $\phi_q(\alpha_0) = \phi(\alpha_0)$.) Note also that the coefficient of $\alpha^2$ is positive as condition A.25 is false so the quadratic approximation $\phi_q$ has a unique minimum.

- The new trial value $\alpha_1$ is taken to be the minimiser of $\phi_q(\alpha)$, so

$$\alpha_1 = -\frac{\phi'(0)\alpha_0^2}{2\left[\phi(\alpha_0) - \phi(0) - \alpha_0\phi'(0)\right]}. \tag{A.27}$$

  It can be shown that

$$\alpha_1 \leq \frac{1}{2(1-c_1)}\alpha_0.$$

  This means that for $c_1 < \frac{1}{2}$, $\alpha_1 < \alpha_0$. For the small values of $c_1$ normally used, $\alpha_1 \approx \frac{1}{2}\alpha_0$.

3. If the sufficient decrease condition Eq. A.24 is satisfied at $\alpha_1$ we know by Thm. A.13 that there is a point in the interval $(\alpha_1, \alpha_0)$ where the strong Wolfe conditions hold.

4. Otherwise we can construct a **cubic** function that interpolates the four pieces of information $\phi(0)$, $\phi'(0)$ and $\phi(\alpha_0)$ and $\phi(\alpha_1)$, namely

$$\phi_c(\alpha) = a\alpha^3 + b\alpha^2 + \phi'(0)\alpha + \phi(0). \qquad \text{(A.28)}$$

- The values of $a$ and $b$ that ensure $\phi_c$ interpolates correctly are

$$\begin{bmatrix} a \\ b \end{bmatrix} = \frac{1}{\alpha_0^2 \alpha_1^2 (\alpha_1 - \alpha_0)} \begin{bmatrix} \alpha_0^2 & -\alpha_1^2 \\ -\alpha_0^3 & \alpha_1^3 \end{bmatrix} \begin{bmatrix} D_1 \\ D_0 \end{bmatrix}, \tag{A.29}$$

where

$$D_0 = \phi(\alpha_0) - \phi(0) - \alpha_0 \phi'(0) \quad \text{and} \tag{A.30}$$

$$D_1 = \phi(\alpha_1) - \phi(0) - \alpha_1 \phi'(0). \tag{A.31}$$

- By differentiating $\phi_c(\alpha)$, we find that the minimiser $\alpha_2$ of $\phi_c$ is given by

$$\alpha_2 = \frac{-b + \sqrt{b^2 - 3a\phi'(0)}}{3a}. \tag{A.32}$$

- An obvious question is whether $\alpha_2$ is well-defined, i.e is the argument of the square root in (A.32) always non-negative? Fortunately the answer is yes. See
- It can be shown that provided $c_1 < \frac{1}{4}$, $\alpha_2$ lies in the interval $[0, \dfrac{2}{3}\alpha_1]$. (The constant $c_1$ is usually taken to be much smaller than $\frac{1}{4}$.)

5. If necessary, this step is repeated, using a cubic interpolant of $\phi(0)$, $\phi'(0)$ and the two most recent values of $\phi$, until an $\alpha$ that satisfies the strong Wolfe conditions is found. If any $\alpha_i$ is either too close to its predecessor $\alpha_{i-1}$ or else too much smaller than $\alpha_{i-1}$, we reset $\alpha_i = \alpha_{i-1}/2$. This **safeguard** procedure ensures that we make reasonable progress on each iteration and that the final $\alpha$ is not too small.

**Alternative Interpolant** If directional derivatives ($\phi'$–values) are easily (cheaply) computed then an alternative approach is based on cubic interpolation of the values of $\phi$ and $\phi'$ at the two most recent values of $\alpha$.

Suppose that we have an interval $[a, b]$ known to contain "good" step lengths (as above) and two previous step length estimates $\alpha_{i-1}$ and $\alpha_i$ in this interval. Use a cubic $\Phi_c$ to interpolate $\phi(\alpha_{i-1})$, $\phi'(\alpha_{i-1})$, $\phi(\alpha_i)$ and $\phi'(\alpha_i)$. (This interpolant always exists and is unique.) The minimiser in $[a, b]$ is either at one of the endpoints or in the interior at

$$\alpha_{i+1} = \alpha_i + \Delta\alpha_i, \tag{A.33}$$

where

$$\Delta\alpha_i = -\Delta\alpha_{i-1}\left[\frac{\phi'(\alpha_i) + d_2 - d_1}{\phi'(\alpha_i) - \phi'(\alpha_{i-1}) + 2d_2}\right], \tag{A.34}$$

with

$$\begin{aligned} d_1 &= \phi'(\alpha_{i-1}) + \phi'(\alpha_i) - \frac{\Delta\phi_{i-1}}{\Delta\alpha_{i-1}}, \\ d_2 &= \left[d_1^2 - \phi'(\alpha_{i-1})\phi'(\alpha_i)\right]^{1/2}. \end{aligned}$$

In the above, $\Delta\phi_{i-1} \equiv \phi_i - \phi_{i-1}$ and $\Delta\alpha_{i-1} \equiv \alpha_i - \alpha_{i-1}$ . The interpolation process is repeated (if necessary) by discarding one or other of the two "old" step lengths $\alpha_{i-1}$ or $\alpha_i$. We can use the Wolfe conditions to guide us as to which should be discarded.

**Initial Step Length** For Newton and quasi-Newton methods, the initial step $\alpha_0 = 1$ should always be chosen. (Remember that as discussed in Section 4.4.3, this allows rapid convergence to take effect.)

For methods such as steepest descent and the nonlinear conjugate gradient method , a reasonable choice is to assume that the first-order change in $f$ at $x_k$ will be the same as that obtained at the previous step. In other words, we choose $\alpha_0$ so that $\alpha_0 {g_k}^T p_k = \alpha_{k-1} g_{k-1}^T p_{k-1}$ or

$$\alpha_0 = \alpha_{k-1} \frac{g_{k-1}^T p_{k-1}}{{g_k}^T p_k}$$

An alternative would be to use quadratic interpolation to fit $f(x_{k-1})$, $f(x_k)$ and $\phi'(0) = {g_k}^T p_k$.

## A.14 A Step-Length Selection Algorithm — the Details

We now describe in detail a one-dimensional search procedure that is **guaranteed** to find step length  satisfying the **strong** Wolfe conditions for any values of the parameters $c_1, c_2$ satisfying $0 < c_1 < c_2 < 1$. As before, we assume that $p$ is a descent direction and that $f$ is bounded below along $p$.

The algorithm  has two stages. The first stage **LineSearch** (Algorithm A.1) begins with a trial estimate $\alpha_1$ and keeps increasing it till it either finds an acceptable step length  or an interval that brackets the desired step lengths. In the latter case, the second stage is invoked by calling a function called **zoom** (Algorithm A.2 below), which successively decreases the size of the interval until an acceptable step length  is found.

The **Line Search** algorithm on the next slide either

- terminates with $\alpha_*$ set to a step length that satisfies the strong Wolfe conditions .
- or fails due to “maximum number of iterations exceeded”.

The parameter $\alpha_{max}$ is an upper bound on the step length and $i_{max}$ is the maximum number of steps allowed.

**Algorithm A.1 (Line Search)**

```
begin
  Set $\alpha_0 \leftarrow 0$, choose $i_{max}$, $\alpha_{max}$ and $\alpha_1 > 0$
  $i \leftarrow 1$
  while $i < i_{max}$ do
      Evaluate $\phi(\alpha_i)$
      if $\phi(\alpha_i) > \phi(0) + c_1 \alpha_i \phi'(0)$
        or $\left[\phi(\alpha_i) \geq \phi(\alpha_{i-1}) \quad \wedge \quad i > 1\right]$
        then
            set $\alpha_* \leftarrow$ zoom $(\alpha_{i-1}, \alpha_i)$; exit
      fi
      Evaluate $\phi'(\alpha_i)$
      if $|\phi'(\alpha_i)| \leq -c_2 \phi'(0)$
        then
            set $\alpha_* \leftarrow \alpha_i$; exit
      fi
      if $\phi'(\alpha_i) \geq 0$
        then
            set $\alpha_* \leftarrow$ zoom $(\alpha_i, \alpha_{i-1})$; exit
      fi
      FAILURE — choose $\alpha_{i+1} \in (\alpha_i, \alpha_{max})$  so $\alpha_{i-1} < \alpha_i$ for each $i$.
      set $i \leftarrow i + 1$
  end (while)
end
```

Note that the sequence of trial step lengths $\{\alpha_i\}$ is monotonically increasing, but that the order of the arguments supplied to the **zoom** function may vary.

Line number 20 performs extrapolation to find the next trial value $\alpha_{i+1}$ (as this line is only called if the interval $(\alpha_{i-1}, \alpha_i)$ does **not** contain step lengths satisfying the strong Wolfe conditions ). To implement this step we could simply set $\alpha_{i+1}$ to a fixed multiple (greater than 1) of $\alpha_i$ or else to a point midway between $\alpha_i$ and $\alpha_{max}$.

In the following Lemma we show that the intervals $(\alpha_{i-1}, \alpha_i)$ used in the calls to **zoom** at lines 9 and 18 always contain step lengths $\bar{\alpha}$ satisfying the strong Wolfe conditions — so it makes sense to "zoom" in on the interval.

**Lemma A.14** *In Alg. A.1, the interval $(\alpha_{i-1}, \alpha_i)$ contains step lengths $\bar{\alpha}$ satisfying the strong Wolfe conditions if one of the following three conditions is satisfied — i.e. at lines 6, 7 and 16.*

*(i)* $\alpha_i$ *violates the first Wolfe condition; (Line number 6)*

*(ii)* $\phi(\alpha_i) \geq \phi(\alpha_{i-1})$*; (Line number 7)*

*(iii)* $\phi'(\alpha_i) \geq 0$*. (Line number 16)*

**Proof:** We check that the conclusion holds for each condition in turn. In all cases **as the algorithm failed at the previous iteration** (see Def A.5 for the definition of $\Phi$ and Thm. A.11 for its properties)

$$\Phi(\alpha_{i-1}) \leq 0 \text{ and } \Phi'(\alpha_{i-1}) < 0. \tag{A.35}$$

(i) We have that $\alpha_i$ violates the first Wolfe condition; so the conditions of Theorem A.13 are met.

(ii) By Prop. 4 of $\Phi$ we have that $\Phi(\alpha_i) > \Phi(\alpha_{i-1})$. A simple modification of Theorem A.13 shows that there must be a $\alpha_*$ in $(\alpha_{i-1}, \alpha_i)$ where $\Phi(\alpha_*) \leq 0$ and $\Phi'(\alpha_*) = 0$ — sketch the situation.

(iii) As the Second Wolfe condition is false at line number 16, by Prop 6 of $\Phi$ we have that $\Phi'(\alpha_i) > 0$. Also we have $\Phi(\alpha_i) \leq 0$ and $\phi(\alpha_i) < \phi(\alpha_{i-1})$. (The latter inequality doesn't help us.) Also note (A.35). See the Figure on the next Slide.

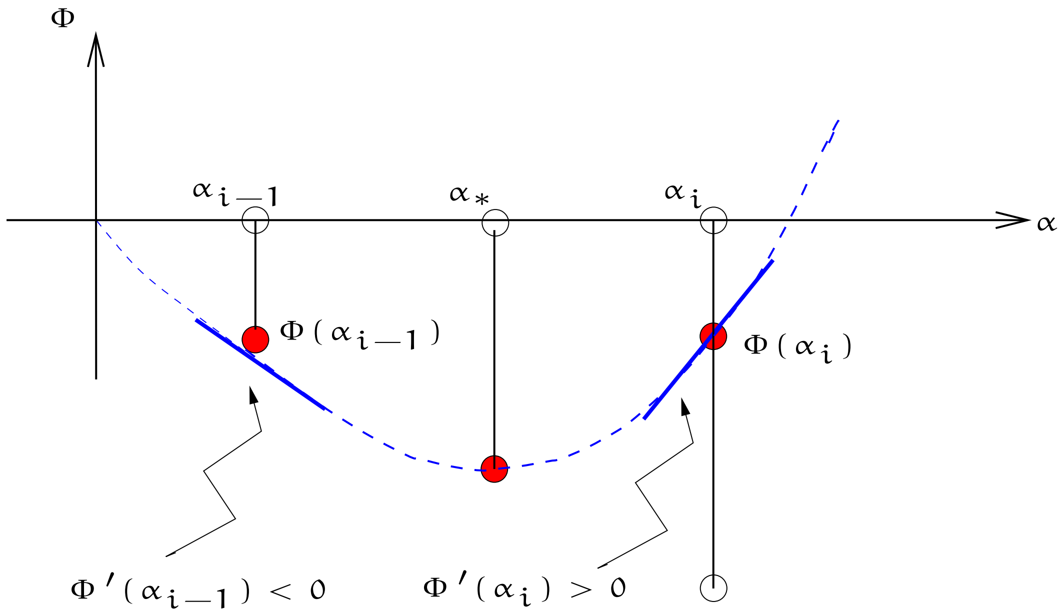


Figure 31: $\Phi'(\alpha_i) > 0$ and $\Phi \leq 0$ at $\alpha_{i-1}$ and $\alpha_i$

$\Phi$ is $\leq 0$ and strictly decreasing at $\alpha_{i-1}$ and is $\leq 0$ and strictly increasing at $\alpha_i$ so, as $\Phi'$ is continuous, $\Phi$ must have have a minimum in $(\alpha_{i-1}, \alpha_i)$ and is of course negative at the minimum. So there must be a $\alpha_*$ in $(\alpha_{i-1}, \alpha_i)$ where $\Phi(\alpha_*) \leq 0$ and $\Phi'(\alpha_*) = 0$. ■

Next, we analyse the **inputs** to the algorithm **zoom** . The order of its input arguments is such that each call (in **linesearch**) takes the form **zoom** $(\alpha_{lo}, \alpha_{hi})$,

**Lemma A.15** *where the following conditions hold:*

*(a) the interval bounded by* $\alpha_{lo}$ *and* $\alpha_{hi}$ *(use the notation* $I(\alpha_{lo}, \alpha_{hi})$*) contains step lengths that satisfy the strong Wolfe conditions ;*

*(b)* $\alpha_{lo}$ *is, of all the step lengths generated so far which satisfy the first Wolfe condition, the one with the least* $\phi$*-value — or equivalently*

- $\Phi(\alpha_{lo}) \le 0$ **and** *either* $\Phi(\alpha_{hi}) > 0$ *or* $\phi(\alpha_{hi}) \ge \phi(\alpha_{lo})$.

*(c)* $\alpha_{hi}$ *is chosen so that* $\Phi'(\alpha_{lo})(\alpha_{hi} - \alpha_{lo}) < 0$ *— sketch this and it is much easier to understand!*

**Proof:**

(a) **zoom** is invoked at lines 9 and 18 of **LineSearch** (Algorithm A.1). In both cases one of conditions (i) to (iii) of Lemma A.14 hold so $I(\alpha_{lo}, \alpha_{hi})$ contains $\alpha_*$ such that the strong Wolfe conditions hold at $\alpha_*$ .

(b)
- If **zoom** is invoked at line 9 then $\alpha_{lo}$ is $\alpha_{i-1}$. In this case $\Phi(\alpha_{lo}) \leq 0$ and either $\Phi(\alpha_{hi}) > 0$ or $\phi_{hi} \geq \phi_{lo}$ as claimed.
- If **zoom** is invoked at line 18 then $\alpha_{lo}$ is $\alpha_i$. We have $\Phi_{lo} \leq 0$ and $\phi(\alpha_i) < \phi(\alpha_{i-1})$ which translates into $\phi_{hi} \geq \phi_{lo}$ as claimed. (It is not true that $\Phi_{hi} > 0$ but we only need one of the two conditions $\phi_{hi} \geq \phi_{lo}$ and $\Phi_{hi} > 0$ to hold for $(\Phi_{hi} > 0) \vee (\phi_{hi} \geq \phi_{lo})$ to be true.)

(c)

- If **zoom** is invoked at line 9 we have $\alpha_{lo} = \alpha_{i-1}$ and $\alpha_{hi} = \alpha_i$ so $\alpha_{hi} - \alpha_{lo}$ is positive. The condition $\phi'(\alpha_{lo}) < 0$ holds as one of the exit conditions from the **previous** invocation of **LineSearch** $\phi'(\alpha_{i-1}) < 0$. But (because the second Wolfe condition failed at line 12 of **LineSearch** at the **previous** iteration) — by property 6 of $\Phi$, the functions $\phi'$ and $\Phi'$ have the same sign at $\alpha_{i-1}$ so $\Phi'(\alpha_{i-1}) \equiv {\Phi_{lo}}' < 0$ as claimed.
- If **zoom** is invoked at line 18 then $\alpha_{lo} = \alpha_i$ and $\alpha_{hi} = \alpha_{i-1}$ so $\alpha_{hi} - \alpha_{lo}$ is negative. But (because the second Wolfe condition failed at line 12 of **LineSearch** at the **current** iteration) — by property 6 of $\Phi$, the functions $\phi'$ and $\Phi'$ have the same sign at $\alpha_i$ so $\Phi'(\alpha_i) \equiv {\Phi_{lo}}' > 0$ as claimed. ■

**Algorithm A.2 (zoom)**

```
begin
  repeat
  Interpolate (using quadratic, cubic or bisection method)
    to find a trial step length $\alpha_j$ between $\alpha_{lo}$ and $\alpha_{hi}$.
  Evaluate $\phi(\alpha_j)$;
  if $\phi(\alpha_j) > \phi(0) + c_1 \alpha_j \phi'(0)$ or $\phi(\alpha_j) \geq \phi(\alpha_{lo})$
    then
        set $\alpha_{hi} \leftarrow \alpha_j$;
    else
        Evaluate $\phi'(\alpha_j)$;
        if $|\phi'(\alpha_j)| \leq -c_2 \phi'(0)$
          then
              set $\alpha_* \leftarrow \alpha_j$; exit
        fi
        if $\phi'(\alpha_j)(\alpha_{hi} - \alpha_{lo}) \geq 0$
          then
              set $\alpha_{hi} \leftarrow \alpha_{lo}$;
        fi
        set $\alpha_{lo} \leftarrow \alpha_j$;
  fi
  until (Too many iterations) or (Tolerance reached).
end
```

Finally, an analysis of the **output** of **zoom** . Only two cases can arise — we refer to them as the "Easy" and "Hard" cases respectively.

**Lemma A.16 (Easy Case)** *If* $\Phi_{\mathrm{lo}} \le 0$ **and** $\Phi_{\mathrm{hi}} \ge 0$ **and** $\Phi'_{\mathrm{lo}}(\alpha_{\mathrm{hi}} - \alpha_{\mathrm{lo}}) < 0$ *then the interval* $\mathrm{I}(\alpha_{\mathrm{lo}}, \alpha_{\mathrm{hi}})$ *contains a point* $\alpha^*$ *where* $\Phi(\alpha^*) \le 0$ *and* $\Phi'(\alpha^*) = 0$.

**Proof:** There are two possibilities;

(i) $\alpha_{\mathrm{lo}} < \alpha_{\mathrm{hi}}$: we have $\Phi_{\mathrm{lo}} \le 0 \le \Phi_{\mathrm{hi}}$ **and** $\Phi'_{\mathrm{lo}} < 0$. So, as $\Phi$ is decreasing at $\alpha_{\mathrm{lo}}$ and is $\le 0$ there, it follows that $\Phi$ has a minimum in $(\alpha_{\mathrm{lo}}, \alpha_{\mathrm{hi}})$ — the minimum cannot be at $\alpha_{\mathrm{hi}}$ as $\Phi$ is non-negative there. Therefore there is a point $\alpha^* \in (\alpha_{\mathrm{lo}}, \alpha_{\mathrm{hi}})$ where $\Phi(\alpha^*) < 0$ and $\Phi'(\alpha^*) = 0$.

(ii) $\alpha_{\mathrm{hi}} < \alpha_{\mathrm{lo}}$: exercise.

■

**Lemma A.17 (Hard Case)** *If* $\Phi_{\mathtt{lo}} \le 0$ **but** $\Phi_{\mathtt{hi}} < \Phi_{\mathtt{lo}}$ **and** $\phi(\alpha_{\mathtt{hi}}) \ge \phi(\alpha_{\mathtt{lo}})$ **and** $\Phi'_{\mathtt{lo}}(\alpha_{\mathtt{hi}} - \alpha_{\mathtt{lo}}) < 0$ *then the interval* $I(\alpha_{\mathtt{lo}}, \alpha_{\mathtt{hi}})$ *contains a point* $\alpha^*$ *where* $\Phi(\alpha^*) \le 0$ *and the Strong version of the second Wolfe condition holds.*

In this case we cannot prove that $\Phi'(\alpha^*) = 0$ but can show that nonetheless the Strong version of the Second Wolfe condition (4.10b) holds at $\alpha_0$. We will show first that this “hard case” cannot arise when $\alpha_{\mathtt{lo}} < \alpha_{\mathtt{hi}}$ then show that if $\alpha_{\mathtt{lo}} > \alpha_{\mathtt{hi}}$ the Wolfe conditions hold as required.

**Proof:** There are two possibilities;

- $\alpha_{lo} < \alpha_{hi}$: we have $\Phi_{lo} \leq 0$ **and** $\Phi_{hi} < \Phi_{lo}$ **and** $\Phi'_{lo} < 0$. By the Mean Value Theorem, there is an $\alpha_0$ in $(\alpha_{lo}, \alpha_{hi})$ such that:
$$\Phi'(\alpha_0) = \frac{\Phi_{hi} - \Phi_{lo}}{\alpha_{hi} - \alpha_{lo}}. \tag{A.36}$$
It follows that $\Phi'(\alpha_0)$ is **negative**.

  Now, by definition, $\Phi(\alpha) \equiv \phi(\alpha) - [\phi(0) + c_1 \alpha \phi'(0)]$.

  Substituting for $\Phi$ in (A.36) we find that $\Phi'(\alpha_0) = \frac{\phi_{hi} - \phi_{lo}}{\alpha_{hi} - \alpha_{lo}} - c_1 \phi'(0)$ and so, as the first term in the sum is $\geq \mathbf{0}$, we have that
$$0 > \Phi'(\alpha_0) \geq -c_1 \phi'(0) \tag{A.37}$$
which is a contradiction — so this case cannot arise.

- $\alpha_{lo} > \alpha_{hi}$: We have $\Phi_{lo} \le 0$ **and** $\Phi_{hi} < \Phi_{lo}$ **and** $\Phi'_{lo} > 0$. By the Mean Value Theorem, there is an $\alpha_0$ in $(\alpha_{lo}, \alpha_{hi})$ such that:
$$\Phi'(\alpha_0) = \frac{\Phi_{lo} - \Phi_{hi}}{\alpha_{lo} - \alpha_{hi}}. \tag{A.38}$$
It follows that $\Phi'(\alpha_0)$ is **positive**. Substituting for $\Phi$ in (A.38) we find that $\Phi'(\alpha_0) = \frac{\phi_{lo}-\phi_{hi}}{\alpha_{lo}-\alpha_{hi}} - c_1\phi'(0)$ and so, as the first term in the sum is $\le \mathbf{0}$, we have that
$$0 < \Phi'(\alpha_0) \le -c_1\phi'(0) \le -(c_1 + c_2)\phi'(0). \tag{A.39}$$
Substituting for $\Phi'$ in terms of $\phi'$,
$c_2\phi'_0 < c_1\phi'_0 \le \phi'(\alpha_0) \le -c_2\phi'_0$ so the Strong version of the Second Wolfe condition (4.10b) holds at $\alpha_0$.

Provided that $\Phi(\alpha_0) \leq 0$ we are finished. But if $\Phi(\alpha_0) > 0$ this is just case (ii) of the Easy case with $\alpha_0$ playing the role of $\alpha_{hi}$ — so there is an $\alpha_1 \in (\alpha_0, \alpha_{lo})$ such that $\Phi(\alpha_1) \leq 0$ and $\Phi'(\alpha_1) = 0$. ■

**Lemma A.18** *Each iteration of* **zoom** *generates an iterate* $\alpha_j$ *between* $\alpha_{lo}$ *and* $\alpha_{hi}$ *and then replaces one of these endpoints by* $\alpha_j$ *in such a way that the properties (a), (b) and (c) continue to hold.*

**Proof:** Consider the conditions that are true **after** each update in **zoom**. We examine each in turn, expressing the conditions in terms of $\Phi$ where possible.

(i) Line 7: $\Phi'_{lo}(\alpha_{hi} - \alpha_{lo}) < 0$ **and** $\Phi_{lo} \le 0$ **and either** $\Phi_{hi} > 0$ **or** $\phi(\alpha_{hi}) \ge \phi(\alpha_{lo})$.

- The first two conditions are "inherited" as $\alpha_{lo}$ is unchanged and the sign of $\alpha_{hi} - \alpha_{lo}$ is also unchanged.
- Properties (b) and (c) follow immediately.
- RTP (a). We have $\Phi_{lo} \le 0 < \Phi_{hi}$ **and** $\Phi'_{lo}(\alpha_{hi} - \alpha_{lo}) < 0$. This is the Easy case so by Lemma A.16 there is a point $\alpha^*$ where $\Phi(\alpha^*) \le 0$ and $\Phi'(\alpha^*) = 0$.

(ii) Line 12: EXIT with $\alpha^*$ satisfying the Wolfe conditions.

(iii) Line 16: $\Phi_{lo} \leq 0$ **and** $\phi(\alpha_{lo}) < \phi(\alpha_{hi})$ **and** $|\phi'_{lo}| > c_2|\phi'_0|$ **and** $\phi'_{lo}(\alpha_{hi} - \alpha_{lo}) < 0$. The last condition implies that $\Phi'_{lo}(\alpha_{hi} - \alpha_{lo}) < 0$ as $\phi'$ and $\Phi'$ have the same sign when $|\phi'_{lo}| > c_2|\phi'_0|$.

- Properties (b) and (c) follow immediately.
- RTP (a)
  - if $\Phi_{hi} \geq \Phi_{lo}$ we have the Easy case and by Lemma A.16 there is a point $\alpha^*$ where $\Phi(\alpha^*) \leq 0$ and $\Phi'(\alpha^*) = 0$.
  - If $\Phi_{hi} < \Phi_{lo}$ we have the Hard case and by Lemma A.17 there is a point $\alpha^*$ where $\Phi(\alpha^*) \leq 0$ and the Strong version of the Second Wolfe condition holds.

(iv) Line 18: $\Phi_{\mathrm{lo}} \leq 0$ **and** $|\phi'_{\mathrm{lo}}| > c_2|\phi'_0|$ **and** $\phi'_{\mathrm{lo}}(\alpha_{\mathrm{hi}} - \alpha_{\mathrm{lo}}) < 0$. Also (as $\alpha_{\mathrm{hi}}$ unchanged and $\phi_{\mathrm{lo}}$ decreases) we have $\Phi_{\mathrm{hi}} > 0$ **or** $\phi(\alpha_{\mathrm{hi}}) \geq \phi(\alpha_{\mathrm{lo}})$. These are the same conditions as those that hold at Line 16 and so properties (a), (b) and (c) hold after the update.

■

If the new estimate $\alpha_j$ happens to satisfy the strong Wolfe conditions , then **zoom** has done its job, so it terminates with $\alpha_* = \alpha_j$. Otherwise, if $\alpha_j$ satisfies the sufficient decrease condition and has a lower function value than $\alpha_{lo}$, we set $\alpha_{lo} \leftarrow \alpha_j$ to maintain condition (b). If this results in a violation of condition (c), we fix things by setting $\alpha_{hi}$ to the old value of $\alpha_{lo}$ .

As mentioned earlier, the interpolation step that determines $\alpha_j$ should be safeguarded to ensure that the new step length is not too close to the endpoints of the interval. Finally, as the algorithm approaches the solution, two consecutive function values may be indistinguishable in finite-precision arithmetic. A line search must include a stopping test if it cannot attain a lower function value after a certain number (say ten) of trial step lengths. Some procedures also stop if the relative change in $x$ is close to “machine epsilon”.

## A.15 Proof of Theorem 7.2 on Slide 249

We first need a Lemma

**Lemma A.19** *Let* $\mathrm{m}$ *be the quadratic function defined by*

$$\mathrm{m}(\mathrm{p}) = \mathrm{g}^\mathsf{T}\mathrm{p} + \frac{1}{2}\mathrm{p}^\mathsf{T}\mathrm{B}\mathrm{p}$$

*where* $\mathrm{B}$ *is any symmetric matrix. Then*

*(i)* $\mathrm{m}$ *attains a minimum if and only if* $\mathrm{B}$ *is positive semi-definite and* $\mathrm{g}$ *is in the range of* $\mathrm{B}$*.*

*(ii)* $\mathrm{m}$ *has a unique minimiser if and only if* $\mathrm{B}$ *is positive definite and* $\mathrm{g}$ *is in the range of* $\mathrm{B}$*.*

*(iii) if* $\mathrm{B}$ *is positive semi-definite, then every* $\mathrm{p}$ *satisfying* $\mathrm{B}\mathrm{p} = -\mathrm{g}$ *is a global minimiser of* $\mathrm{m}$*.*

**Proof:**

(i)

- $\Rightarrow$ Since $g$ is in the range of B, there is a $p$ s.t. $Bp = -g$. For any vector $w \in \mathbb{R}^n$,

$$\begin{aligned} m(p+w) &= g^T(p+w) + \frac{1}{2}(p+w)^T B(p+w) \\ &= (g^T p + \frac{1}{2}p^T Bp) + \cancel{g^T w + (Bp)^T w} + \frac{1}{2}w^T Bw \\ &= m(p) + \frac{1}{2}w^T Bw \\ &\geq m(p), \end{aligned}$$

as B is positive semidefinite.

- $\Leftarrow$ Let $p$ be a minimiser of $m$ so $\nabla m(p) = Bp + g = 0$. Obviously $g$ is in the range of B. Also from the second-order necessary conditions we have that B is positive semidefinite.

(ii)
- $\Rightarrow$ As in (i$\Rightarrow$) except that at the final step we have $w^{\mathsf{T}}Bw > 0$ whenever $w \neq 0$.
- $\Leftarrow$ Again as in (i$\Leftarrow$) . Once we have B positive semidefinite, assume that B is not positive definite, so there is a nonzero $w$ s.t. $Bw = 0$. But using the algebra from (i $\Rightarrow$) this gives $m(p + w) = m(p)$ which contradicts the assumption that the minimiser is unique.

(iii) Again, just use the algebra from (i $\Rightarrow$).

■

The proof of the Theorem follows on the next Slide.

## Proof of Theorem 7.2 on Slide 249

**Proof:** $\Leftarrow$ Assume that there is a $\lambda \geq 0$ s.t. the conditions (7.15a)–(7.15c) are satisfied. Then by Lemma A.19 (iii), — using conditions (7.15a) and (7.15c)— $p^*$ is a global minimum of the quadratic function (just $m(p)$ with $B$ replaced by $B + \lambda I$):

$$\hat{m}(p) = g^T p + \frac{1}{2} p^T (B + \lambda I) p = m(p) + \frac{\lambda}{2} p^T p. \tag{A.40}$$

As $\hat{m}(p) \geq \hat{m}(p^*)$ we have

$$m(p) \geq m(p^*) + \frac{\lambda}{2} \left( {p^*}^T p^* - p^T p \right). \tag{A.41}$$

Now, as $\lambda\left(\Delta - \|p^*\|\right) = 0$ by assumption and so $\lambda \left( \Delta^2 - {p^*}^T p^* \right) = 0$, it follows that

$$m(p) \geq m(p^*) + \frac{\lambda}{2} \left( \Delta^2 - p^T p \right). \tag{A.42}$$

So as $\lambda \geq 0$ we have that $m(p) \geq m(p^*)$ for all $p$ with $\|p\| \leq \Delta$. So $p^*$ is a minimiser of the trust region problem 7.9 as required.

$\square \leftarrow$

On the next Slide we check the “only if” case.

**Proof:** $\Rightarrow$ Assume that $p^*$ is a global solution to the trust region problem 7.9. RTP that there is a $\lambda \geq 0$ that satisfies (7.15a)–(7.15c).

- If $\|p^*\| < \Delta$ then $p^*$ is an unconstrained minimiser of $m(p)$ and therefore $\nabla m(p^*) = Bp^* + g = 0$ and $\nabla^2 m(p^*) = B$ is positive semidefinite, so (7.15a)–(7.15c) hold for $\lambda = 0$.
- Now assume that $\|p^*\| = \Delta$.
  - Obviously (7.15b) is satisfied.
  - Also $p^*$ satisfies the **equality constrained** problem

$$\min m(p) \quad \text{subject to } \|p\| = \Delta.$$

Now we have to cheat and assume the first-order “K.K.T.” optimality conditions (8.23a), (8.20b) and (8.20e) for **equality constrained** problems that we will discuss in Ch 8!

When we evaluate these conditions (the so-called KKT conditions) for the present problem they state that there is a $\lambda$ such that the "Lagrangian function" $\mathcal{L}$ where

$$\mathcal{L}(p, \lambda) = m(p) + \frac{\lambda}{2}(p^T p - \Delta^2)$$

has a stationary point at $p^*$. When we evaluate the gradient of $\mathcal{L}(p, \lambda)$ w.r.t. $p$ at $p^*$ we get

$$Bp^* + g + \lambda p^* = 0 \quad \text{or equivalently} (B + \lambda I)p^* = -g \quad \text{(A.43)}$$

so (7.15a) holds.

- It remains to check (7.15c). Appealing this time to the second-order necessary conditions for a constrained problem (8.35), it follows immediately that $B + \lambda I$ must be positive semidefinite.

- Finally, we need to check that $\lambda \geq 0$.

  As (7.15a) and (7.15c) are satisfied by $p^*$, Lemma A.19 tells us that $p^*$ minimises $\hat{m}(p)$, so (A.41) holds.

  RTP that

  $$\exists \lambda | \text{(7.15a)–(7.15c) are satisfied } \textbf{AND} \quad \lambda \geq 0$$

  Assume the contrary:

  $$\forall \lambda, \text{(7.15a)–(7.15c) are satisfied} \Rightarrow \lambda < 0$$

  or equivalently: suppose that $\lambda$ is always negative when (7.15a)–(7.15c) are satisfied.

Then (A.41) tells us that $m(p) \geq m(p^*)$ whenever $\|p\| \geq \|p^*\| = \Delta$. Since we already know that $p^*$ minimises $m(p)$ for $\|p\| \leq \Delta$ it follows that $p^*$ is a **unconstrained** minimiser of $m(p)$.

By the familiar necessary conditions for unconstrained problems, we have that $Bp = -g$ and $B$ is positive semidefinite. Therefore conditions (7.15a) and (7.15c) are satisfied by $\lambda = 0$ which contradicts our assumption that $\lambda$ must be negative. So there exists $\lambda \geq 0$ s.t. (7.15a)–(7.15c) are satisfied. ■

## A.16 Global Convergence for the Trust Region Algorithm

In the following we will prove a sequence of results which lead to the conclusion that the trust region algorithm  has the vital property of **global convergence**, i.e. that the sequence of gradients $\{g_k\}$ generated by Alg. 7.1 converges to zero (if $\eta > 0$).

We start by proving that the dogleg and two-dimensional subspace minimisation methods produce approximate solutions $p_k$ of the sub-problem that produce the following guaranteed minimum reduction in $m(p)$:

$$m_k(0) - m_k(p_k) \geq c_1 \|g_k\| \min\left(\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right), \tag{A.44}$$

for some constant $c_1 \in (0, 1]$. We will then apply this result to prove global convergence.

First we show that:

**Lemma A.20** *The Cauchy point* $p_k^c$ *(7.7) satisfies A.44 with* $c_1 = \frac{1}{2}$.

**Proof:**

**[Case 1]** Consider first the case $g_k{}^T B_k g_k \leq 0$. Here, we have

$$\begin{aligned}
m_k(p_k^c) - m_k(0) &= m_k(-\Delta_k g_k / \|g_k\|) - f_k \\
&= -\frac{\Delta_k}{\|g_k\|}\|g_k\|^2 + \frac{1}{2}\frac{\Delta_k^2}{\|g_k\|^2} g_k{}^T B_k g_k \\
&\leq -\Delta_k \|g_k\| \\
&\leq -\|g_k\| \min\left(\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right) \\
&\leq -\frac{1}{2}\|g_k\| \min\left(\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right),
\end{aligned}$$

as required (note that $-a, -b \leq -\min(a, b)$ when $a, b \geq 0$).

**[Case 2A]** Now consider the case $g_k{}^T B_k g_k > 0$ with

$$\frac{\|g_k\|^3}{\Delta_k g_k{}^T B_k g_k} \leq 1. \tag{A.45}$$

We have $\tau = \|g_k\|^3 / \Delta_k g_k{}^T B_k g_k$ and so ($p_k^c = \tau \Delta g / \|g\|$)

$$\begin{aligned}
m_k(p_k^c) - m_k(0) &= \frac{\|g_k\|^4}{g_k{}^T B_k g_k} + \frac{1}{2} g_k{}^T B_k g_k \, \frac{\|g_k\|^4}{(g_k{}^T B_k g_k\,)^2} \\
&= -\frac{1}{2} \frac{\|g_k\|^4}{g_k{}^T B_k g_k} \\
&\leq -\frac{1}{2} \frac{\|g_k\|^4}{\|B_k\| \|g_k\|^2} \quad \text{(careful with inequalities)} \\
&= -\frac{1}{2} \frac{\|g_k\|^2}{\|B_k\|} \\
&\leq -\frac{1}{2} \|g_k\| \min\left(\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right),
\end{aligned}$$

as required.

**[Case 2B]** Finally, A.45 does not hold, so

$$g_k{}^T B_k g_k \; < \frac{\|g_k\|^3}{\Delta_k}. \tag{A.46}$$

From the definition of $p_k^c$, we have $\tau = 1$, so using A.46 we have

$$\begin{aligned} m_k(p_k^c) - m_k(0) \quad &= \quad -\frac{\Delta_k}{\|g_k\|}\|g_k\|^2 + \frac{1}{2}\frac{\Delta_k{}^2}{\|g_k\|^2} g_k{}^T B_k g_k \\ &\leq \quad -\Delta_k\|g_k\| + \frac{1}{2}\frac{\Delta_k{}^2}{\|g_k\|^2}\frac{\|g_k\|^3}{\Delta_k} \\ &= \quad -\frac{1}{2}\Delta_k\|g_k\| \\ &\leq \quad -\frac{1}{2}\|g_k\| \min\left(\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right), \end{aligned}$$

again, as required.

■

To satisfy A.44, an approximate solution need only achieve a reduction that is at least some fixed fraction of the reduction achieved by the Cauchy point. We state this as a theorem:

**Theorem A.21** *Let* $p_k$ *be any vector such that* $\|p_k\| \leq \Delta_k$ *and* $m_k(0) - m_k(p_k) \geq c_2\left(m_k(0) - m_k(p_k^c)\right)$*. Then* $p_k$ *satisfies A.44 with* $c_1 = c_2/2$*.*

**Proof:** Since $\|p_k\| \leq \Delta_k$ we have by Lemma A.20 that

$$m_k(0) - m_k(p_k) \geq c_2\left(m_k(0) - m_k(p_k^c)\right) \geq \frac{1}{2} c_2 \|g_k\| \min\left(\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right)$$

as required. ■

Note that the dogleg & two-dimensional subspace minimisation algorithms both satisfy A.44 with $c_1 = \frac{1}{2}$ as they both produce approximate results for which $m_k(p_k) \leq m_k(p_k^c)$ (they find a better solution to the quadratic subproblem as they use a better approximation).

### A.16.1 Convergence to Stationary Points

Two separate results can be proved, depending on whether $\eta$ in Alg. 7.1 is zero or positive. For illustrative purposes we state the stronger result which holds for $\eta > 0$.

**Theorem A.22** *Let $\eta \in (0, \frac{1}{4})$ in Alg. 7.1. Suppose that $\|B_k\| \leq \beta$ for some constant $\beta$, that $f$ is and bounded below on the level set $S$*

$$S = \{x | f(x) \leq f(x_0)\} \tag{A.47}$$

*and Lipschitz continuously differentiable in an open neighbourhood $S(R_0) = \{x | \|x - y\| < R_0, \text{some}\, y \in S\}$ of the level set and that all approximate solutions $p_k$ of 7.2 satisfy the inequality A.44 and that $\|p_k\| \leq \Delta_k$, for some positive constants $c_1$. Then*

$$\lim_{k \to \infty} g_k = 0. \tag{A.48}$$

**Proof:** The proof is technical and is given in Appendix A.17.

## A.17 Proof of Theorem A.22 on Slide 570

Suppose that for some $M$, $\nabla f(x_M) \equiv g_M \neq 0$ (otherwise there is nothing to prove). Let $\beta_1$ be the Lipschitz constant for $g(x)$ on the level set A.47, then

$$\|g(x) - g_M\| \leq \beta_1 \|x - x_M\|$$

for any $x$ in the level set. Now define the scalars

$$\epsilon = \frac{1}{2}\|g_M\|, \quad R = \min\left(\frac{\|g_M\|}{2\beta_1}, R_0\right) = \min\left(\frac{\epsilon}{\beta_1}, R_0\right)$$

and the ball

$$\mathcal{B}(x_M, R) = \{x | \, \|x - x_M\| \leq R\}.$$

This ball is contained in $S(R_0)$ so Lipschitz continuity holds.

Then, if $x \in \mathcal{B}(x_M, R)$ , $\|g(x)\|$ "stays positive" inside the ball as:

$$\|g(x)\| \geq \|g_M\| - \|g(x) - g_M\| \geq \frac{1}{2}\|g_M\| = \epsilon.$$

If the entire sequence $\{x_k\}_{k\geq M}$ stays inside the ball $\mathcal{B}(x_M, R)$ then $\|g_k\| \geq \varepsilon > 0$ for all $k \geq M$.

**We need to show that this cannot happen.** (Jump to Slide 578 if you are impatient.)

- Assume that there is an $\varepsilon > 0$ and a positive index $M$ s.t.

$$\|g_k\| \geq \varepsilon, \forall k \geq M. \tag{A.49}$$

- It is easy to check that

$$\begin{aligned} |\rho_k - 1| &= \left| \frac{(f_k - f(x_k + p_k)) - (m_k(0) - m_k(p_k))}{m_k(0) - m_k(p_k)} \right| \\ &= \left| \frac{m_k(p_k) - f(x_k + p_k)}{m_k(0) - m_k(p_k)} \right| \end{aligned} \tag{A.50}$$

- We can rewrite Taylor's Theorem (3.6) in the form

$$f(x_k + p_k) = f(x_k) + g_k^T p_k + \int_0^1 [g(x_k + tp_k) - g_k]^T p_k dt$$

- So $(x_{k+1} \equiv x_k + p_k)$

$$|m_k(p_k) - f(x_{k+1}k)| = \left|\frac{1}{2}p_k^T B_k p_k - \int_0^1 [g(x_k + tp_k) - g_k]^T p_k dt\right| \leq (\beta/2)\|p_k|^2 + \beta_1\|p_k\|^2 \tag{A.51}$$

- From (A.44) we have that for $k \geq M$;

$$m_k(0) - m_k(p_k) \geq c_1\|g_k\| \min\left(\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right) \geq c_1 \varepsilon \min\left(\Delta_k, \frac{\varepsilon}{\beta}\right). \tag{A.52}$$

- Combining these results and using the fact that $\|p_k\| \leq \Delta_k$;

$$|\rho_k - 1| \leq \frac{\Delta_k^2(\beta/2 + \beta_1)}{c_1 \varepsilon \min(\Delta_k, \varepsilon/\beta)} \tag{A.53}$$

- We now find a bound on the RHS in (A.53) that holds whenever $\Delta_k$ is sufficiently small, i.e. for all $\Delta_k \leq \bar{\Delta}$, where

$$\bar{\Delta} = \min\left(\frac{1}{2}\frac{c_1 \varepsilon}{(\beta/2 + \beta_1)}, R_0\right).$$

- The $R_0$ term ensures that the bound (A.51) holds as $\|p_k\| \leq \Delta_k \leq \bar{\Delta} \leq R_0$.

- Since $c_1 \leq 1$ we have $\bar{\Delta} \leq \varepsilon/\beta$.

- Because of this, for all $\Delta_k \in [0, \bar{\Delta}]$ we have $\min(\Delta_k, \varepsilon/\beta) = \Delta_k$ so we have:

$$|\rho_k - 1| \leq \frac{\Delta_k^2(\beta/2 + \beta_1)}{c_1 \varepsilon \Delta_k} = \frac{\Delta_k(\beta/2 + \beta_1)}{c_1 \varepsilon} \leq \frac{\bar{\Delta}(\beta/2 + \beta_1)}{c_1 \varepsilon} \leq \frac{1}{2}. \tag{A.54}$$

- So $\rho_k \geq \frac{1}{2}$ and certainly $\rho_k \geq \frac{1}{4}$ which is what we need.

- By the operation of the TR Algorithm 7.1, we have $\Delta_{k+1} \geq \Delta_k$ whenver $\Delta_k$ falls below the threshold $\bar{\Delta}$.
- So reduction of $\Delta_k$ in the algorithm can only happen when $\Delta_k \geq \bar{\Delta}$.
- It follows that we must have

$$\Delta_k \geq \min(\Delta_M, \bar{\Delta}/4) \quad \text{for all } k \geq M. \tag{A.55}$$

- Now we need to eliminate the possibility that there is an infinite subsequence $\mathcal{K}$ of $k$ values such that $\rho_k \geq \frac{1}{4}$ for $k \in \mathcal{K}$.
- For elements of this subsequence greater than $M$ $(k \in \mathcal{K}, k \geq M)$ we have (as $\rho_k \geq \frac{1}{4}$ )

$$\begin{aligned} f_k - f(x_{k+1}) &= f_k - f(x_k + p_k) \\ &\geq \frac{1}{4}\left[m_k(0) - m_k(p_k)\right] \\ &\geq \frac{1}{4} c_1 \varepsilon \min(\Delta_k, \varepsilon/\beta). \end{aligned}$$

- As $f$ is bounded below, The LHS $\rightarrow 0$ so

$$\lim_{k \in \mathcal{K}, k \geq M} \Delta_k = 0.$$

- But this contradicts (A.55) which requires $\Delta_k$ to be bounded below by a positive number.

- So no such infinite subsequence $\mathcal{K}$ can exist.
- Therefore we must have $\rho_k \leq \frac{1}{4}$ for all $k$ sufficiently large.
- But this means that $\Delta_k$ is multiplied by $\frac{1}{4}$ at every iteration for all $k$ sufficiently large.
- So $\lim\limits_{k \to \infty} \Delta_k = 0$.
- Again we have a contradiction with (A.55).
- Our original assumption $\|g_k\| \geq \varepsilon$ must be false.
- We never (in this digression) specifie dthe value of $\varepsilon$.
- This means that $\liminf\limits_{k \to \infty} g(x_k) = 0$.
- This in fact is the result when the parameter $\eta$ is set to 0.
- For $\eta > 0$ we can do better — back to the main proof.

**Therefore, the sequence $\{x_k\}_{k\geq m}$ eventually leaves the ball.**

Let $L \geq M$ be the index such that $x_{L+1}$ is the first iterate after $x_M$ to leave the ball. Since $\|g_k\| \geq \varepsilon$ for $k = M, M+1, \ldots, L$, we can write ( the last inequality follows from (A.52) which holds when $\|g_k\| \geq \varepsilon$):

$$\begin{aligned} f(x_M) - f(x_{L+1}) &= \sum_{k=M}^{L} f(x_k) - f(x_{k+1}) \\ &\geq \sum_{k=M, x_k \neq x_{k+1}}^{L} \eta[m_k(0) - m_k(p_k)] \\ &\geq \sum_{k=M, x_k \neq x_{k+1}}^{L} \eta c_1 \varepsilon \min\left(\Delta_k, \frac{\varepsilon}{\beta}\right), \end{aligned}$$

where we have limited the sum to the iterations $k$ for which $x_k \neq x_{k+1}$, i.e. to the iterations where a step was taken (for which $\rho_k > \eta$).

- If $\Delta_k \leq \varepsilon/\beta$ for all $k = M, M+1, \ldots, L$, we have

$$\begin{aligned} f(x_M) - f(x_{L+1}) &\geq \eta c_1 \varepsilon \sum_{k=M, x_k \neq x_{k+1}} \Delta_k \geq \eta c_1 \varepsilon R \\ &= \eta c_1 \varepsilon \min\left(\frac{\varepsilon}{\beta_1}, R_0\right). \qquad \text{(A.56)} \end{aligned}$$

- In the other case, we have $\Delta_k > \varepsilon/\beta$ for some $k = M, M+1, \ldots, L$ and so

$$f(x_M) - f(x_{L+1}) \geq \eta c_1 \varepsilon \frac{\varepsilon}{\beta}. \qquad \text{(A.57)}$$

Since the sequence $\{f(x_k)\}_{k=0}^{\infty}$ is decreasing and bounded below, we know that

$$f(x_k) \to f^* \tag{A.58}$$

for some finite $f^*$. Therefore, using A.56 and A.57, we have that

$$\begin{aligned} f(x_M) - f^* &\geq f(x_M) - f(x_{L+1}) \\ &\geq \eta c_1 \varepsilon \min\left(\frac{\varepsilon}{\beta}, \frac{\varepsilon}{\beta_1}, R_0\right) \\ &= \frac{1}{2}\eta c_1 \|g_M\| \min\left(\frac{\|g_M\|}{2\beta}, \frac{\|g_M\|}{2\beta_1}, R_0\right) > 0. \end{aligned}$$

Finally, as $f(x_M) - f^* \downarrow 0$ we must have $g_M \to 0$. ■

## A.18 Advanced nonlinear conjugate gradient methods

### A.18.1 Dai-Yuan Method

In a paper published in 1999, [1] available on-line (within U.L.) at `http://epubs.siam.org/sam-bin/getfile/SIOPT/articles/31899.pdf` the authors proposed a new method. In this section a complete analysis of both convergence and numerical properties is presented — as in this case the analysis is relatively straightforward! The new formula for $\beta_{k+1}$ is

$$\beta_{k+1}^{DY} = \frac{\|g_{k+1}\|^2}{p_k^T y_k}, \tag{A.59}$$

where $y_k \equiv g_{k+1} - g_k$. Again, it is easy to check that the new method reduces to the the FR method in the case of quadratic objective functions with an exact line search.

**Dai-Yuan Method — Global Convergence** Despite the fact that the numerical properties of the new method are no better than those of the FR method, we include a proof of convergence here for two reasons:

1. The proof is similar to that for the F.R. method.
2. It is also a useful preliminary to the corresponding proof for the Hybrid method in Section A.18.2 below — which **does** have good numerical behaviour as well as convergence properties.

We will state the required result as a Theorem.

**Theorem A.23 (DY-convergence)** *Under the same assumptions as those of Zoutendijk's Theorem 4.3, (in particular that the weak Wolfe conditions hold and that $f$ is bounded below) — with the exception that we need* **not** *assume that the $p_k$ are descent directions, we have that the algorithm either stops at a stationary point ($\|g_k\| = 0$) or $\liminf \|g_k\| = 0$.*

**Proof:**

- First we show using induction that all the search directions $p_k$ are descent directions, i.e. $p_k^T g_k < 0$.
  - (Base Step — $k = 0$) Obviously $p_1 = -g_1$ so $p_1^T g_1 = -\|g_1\|^2 < 0$.
  - (Inductive Step — $k > 0$) Assume that $p_k^T g_k < 0$. RTP $p_{k+1}^T g_{k+1} < 0$. Using the update rule and the definition of $\beta_{k+1}^{DY}$ (A.59), we have

    $$p_{k+1}^T g_{k+1} = \frac{\|g_{k+1}\|^2}{p_k^T y_k} p_k^T g_k.$$

    But using the second Wolfe condition,

    $$p_k^T y_k \equiv p_k^T (g_{k+1} - g_k) \geq (c_2 - 1) p_k^T g_k > 0.$$

    Therefore $p_{k+1}^T g_{k+1} < 0$ as required.

- Now to prove the major result. Rewriting the update rule as $p_{k+1} + g_{k+1} = \beta_{k+1} p_k$ and taking the squared norm of each side we have;

  $$\|p_{k+1}\|^2 = (\beta_{k+1}^{DY})^2 \|p_k\|^2 - \|g_{k+1}\|^2 - 2{p_{k+1}}^T g_{k+1}. \quad \text{(A.60)}$$

  It is easy to check that:

  $$\begin{aligned} \frac{p_{k+1}^T g_{k+1}}{p_k^T g_k} &\equiv \frac{(-g_{k+1} + \beta_{k+1}^{DY} p_k)^T g_{k+1}}{p_k^T g_k} \\ &= \frac{(-g_{k+1} + \frac{\|g_{k+1}\|^2}{p_k^T y_k} p_k)^T g_{k+1}}{p_k^T g_k} \\ &= \frac{-\|g_{k+1}\|^2 (p_k^T y_k) + \|g_{k+1}\|^2 (p_k g_{k+1})}{(p_k^T g_k)(p_k^T y_k)} \\ &= \frac{\|g_{k+1}\|^2}{p_k^T y_k} = \beta_{k+1}^{DY}. \end{aligned}$$

So $\beta_{k+1}^{DY} = \dfrac{p_{k+1}^T g_{k+1}}{(p_k^T g_k)}$. (N.B. only when the DY formula for $\beta_{k+1}$ is used!)

Now, divide (A.60) across by $(p_{k+1}^T g_{k+1})^2$ and using the above formula for $\beta_{k+1}^{DY}$ we have

$$\frac{\|p_{k+1}\|^2}{(p_{k+1}^T g_{k+1})^2} = \frac{\|p_k\|^2}{(p_k^T g_k)^2} - \frac{2}{p_{k+1}^T g_{k+1}} - \frac{\|g_{k+1}\|^2}{(p_{k+1}^T g_{k+1})^2} \tag{A.61}$$

Completing the square we have

$$\frac{\|p_{k+1}\|^2}{(p_{k+1}^T g_{k+1})^2} = \frac{\|p_k\|^2}{(p_k^T g_k)^2} - \left(\frac{1}{\|g_{k+1}\|} + \frac{\|g_{k+1}\|}{p_{k+1}^T g_{k+1}}\right)^2 + \frac{1}{\|g_{k+1}\|^2}$$

and so

$$\frac{\|p_{k+1}\|^2}{(p_{k+1}^T g_{k+1})^2} \leq \frac{\|p_k\|^2}{(p_k^T g_k)^2} + \frac{1}{\|g_{k+1}\|^2}. \tag{A.62}$$

As $\frac{\|p_0\|^2}{(p_0^T g_0)^2} \equiv \frac{1}{\|g_0\|^2}$, by iterating (A.62)

$$\frac{\|p_k\|^2}{(p_k^T g_k)^2} \leq \sum_{i=0}^{k} \frac{1}{\|g_i\|^2} \quad \text{for } k \geq 0. \tag{A.63}$$

If the Theorem is false then for some $c > 0$, we have $\|g_k\| \geq \gamma > 0$ for all $k > 0$. Substituting in (A.63), it follows that

$$\frac{\|p_k\|^2}{(p_k^T g_k)^2} \leq \frac{k+1}{\gamma^2}$$

and so

$$\sum_{k\geq 0} \frac{(p_k^T g_k)^2}{\|p_k\|^2} \geq \sum_{k\geq 0} \frac{\gamma^2}{k+1}.$$

But this final sum diverges which contradicts Zoutendijk's Theorem 4.3. ■

**Dai-Yuan Method — Numerical Behaviour** We can "re-cycle" a lot of the algebra from the Theorem. Repeating (A.61);

$$\frac{\|p_{k+1}\|^2}{(p_{k+1}^T g_{k+1})^2} = \frac{\|p_k\|^2}{(p_k^T g_k)^2} - \frac{2}{p_{k+1}^T g_{k+1}} - \frac{\|g_{k+1}\|^2}{(p_{k+1}^T g_{k+1})^2}$$

Let $l_k = \frac{p_k^T g_{k+1}}{p_k^T g_k}$ and re-using the formula $p_{k+1}^T g_{k+1} = \frac{p_k^T g_k}{p_k^T y_k}\|g_{k+1}\|^2$;

we have:

$$\frac{\|p_{k+1}\|^2}{(p_{k+1}^T g_{k+1})^2} = \frac{\|p_k\|^2}{(p_k^T g_k)^2} + (1 - l_k^2)\frac{1}{\|g_{k+1}\|^2}.$$

Defining $\cos(\theta_{k+1})$ as previously, we have:

$$\frac{1}{\cos^2\theta_{k+1}} = \frac{\|g_{k+1}\|^2}{\|g_k\|^2}\frac{1}{\cos^2\theta_k} + (1 - l_k^2).$$

But the strong Wolfe conditions give $|l_k| \leq c_2$ and so $1 - l_k^2 \geq 1 - c_2^2 > 0$. So if as in the analysis of the FR algorithm, a "bad" choice of $p_{k+1}$ sets $\theta_{k+1} \approx \pi/2$ and $g_k \approx g_{k+1}$, then it follows that $\frac{1}{\cos^2\theta_{k+1}} >> 1$ and so the algorithm "sticks" — even when the strong Wolfe conditions hold.

### A.18.2 Hybrid Method

In a paper published in 2001, [2] Y.H. Dai & Y. Yuan proposed a new method that uses a "hybrid" version of $\beta$, namely

$$\beta^{H}_{k+1} = \max\{0, \min\{\beta^{HS}_{k+1}, \beta^{DY}_{k+1}\}\} \tag{A.64}$$

where the Hestenes & Stiefel version of $\beta$ is $\beta^{HS}_{k+1} = \frac{g^T_{k+1} y_k}{p^T_k y_k}$.

The method (like the DY method) is globally convergent requiring only weak Wolfe conditions. The proof is similar to that for the DY method and is given here for the sake of completeness. For clarity we state the result as a general Theorem — note that the hybrid method satisfies the condition (A.65).

## Hybrid Method — Global Convergence

**Theorem A.24** *Suppose that the formula* $\beta_{k+1}$ *satisfies*

$$r_{k+1} \equiv \frac{\beta_{k+1}}{\beta_{k+1}^{DY}} \in [-c, 1], \textit{ where } c = \frac{1-c_2}{1+c_2} > 0. \tag{A.65}$$

*Under the same assumptions as those of Zoutendijk's Theorem 4.3, (in particular that the weak Wolfe conditions hold and that* $f$ *is bounded below) — with the exception that we need* **not** *assume that the* $p_{k+1}$ *are descent directions, we have that the algorithm either stops at a stationary point (*$\|g_{k+1}\| = 0$*) or* $\liminf \|g_{k+1}\| = 0$.

**Proof:** The proof is structured as follows; first some results are assembled, then (as for the DY-convergence proof) we prove that the search directions $p_{k+1}$ are descent directions, finally we prove the substantive result.

**[Preliminaries]** We have as usual that

$$g_{k+1}^T p_{k+1} = -\|g_{k+1}\|^2 + \beta_{k+1} g_{k+1}^T p_k.$$

Using the substitution $\beta_{k+1} = r_{k+1} {\beta_{k+1}}^{DY}$, this can be re-written as

$$\begin{aligned} g_{k+1}^T p_{k+1} &= \frac{\|g_{k+1}\|^2}{p_k^T y_k} \left\{ p_k^T g_k + (r_{k+1} - 1) g_{k+1}^T p_k \right\} \\ &= {\beta_{k+1}}^{DY} \left\{ p_k^T g_k + (r_{k+1} - 1) g_{k+1}^T p_k \right\}. \end{aligned}$$

So, solving for ${\beta_{k+1}}^{DY}$ and using $\beta_{k+1} \equiv r_k {\beta_{k+1}}^{DY}$ we can write

$$\beta_{k+1} = \xi_{k+1} \frac{g_{k+1}^T p_{k+1}}{g_k^T p_k}, \tag{A.66}$$

where $l_k = \dfrac{g_{k+1}^T p_k}{g_k^T p_k}$ as before and "**xi**": $\xi_{k+1} = \dfrac{r_{k+1}}{1 + (r_{k+1} - 1) l_k}$.

Finally, defining "**eta**": $\eta_{k+1} = \frac{1+(r_{k+1}-1)l_k}{l_k-1}$, we can show that $g_{k+1}^T p_{k+1} = \eta_{k+1}\|g_{k+1}\|^2$ as follows:

$$\begin{aligned}
\|g_{k+1}\|^2\eta_{k+1} &= \|g_{k+1}\|^2 \frac{(1+(r_{k+1}-1)l_k)}{l_k-1} \\
&= \|g_{k+1}\|^2 \frac{(1+(r_{k+1}-1)\frac{g_{k+1}^T p_k}{g_k^T p_k})}{\frac{g_{k+1}^T p_k}{g_k^T p_k}-1} \\
&= \|g_{k+1}\|^2 \frac{(r_{k+1}-1)g_{k+1}^T p_k + g_k^T p_k}{g_{k+1}^T p_k - g_k^T p_k} \\
&= \|g_{k+1}\|^2 \frac{(r_{k+1} g_{k+1}^T p_k - p_k^T y_k)}{p_k^T y_k} \\
&= \|g_{k+1}\|^2 \left(-1 + r_{k+1}\frac{g_{k+1}^T p_k}{p_k^T y_k}\right) \\
&= g_{k+1}^T\left(-g_{k+1} + \beta_{k+1} p_k\right) \equiv g_{k+1}^T p_{k+1}.
\end{aligned}$$

**[Descent Direction]** As in the DY-convergence proof, we use induction.

- The Base Step is trivial as $p_0^T g_0 \equiv -\|g_0\|^2 < 0$.
- Assume $p_k^T g_k < 0$.
  - As we are assuming that the weak Wolfe conditions apply to the step size $\alpha_k$ ($p_k$ is a descent direction), we have (as in analysis of DY cgm) $l_k < c_2$.
  - So, as $r_{k+1} \in [-c, 1]$,

$$\begin{aligned} 1 + (r_{k+1} - 1) l_k &\geq 1 + c_2(-c - 1) \quad \textbf{Check!} \\ &= 1 + c_2 \left( -\frac{1 - c_2}{1 + c_2} - 1 \right) = \frac{1 - c_2}{1 + c_2} = c. \end{aligned} \tag{A.67}$$

- Remembering that $\eta_{k+1} \equiv \frac{1+(r_{k+1}-1)l_k}{l_k-1}$, we have just shown that $TL$ (the Top Line in $\eta_{k+1}$) satisfies $TL \geq c > 0$.
- We know that the Bottom Line $BL$ satisfies $BL < 0$ as $l_k < c_2 < 1$.
- It follows that $\eta_{k+1}$ is negative and so as $g_{k+1}^T p_{k+1} \equiv \eta_{k+1} \|g_{k+1}\|^2$ we have $g_{k+1}^T p_{k+1} < 0$ as required.

**[Global Convergence]** Now RTP that $\liminf \|g_{k+1}\| = 0$.

Assume not — so we have for some $\gamma > 0$ that $\|g_{k+1}\| > \gamma$ for all $k \geq 0$. As usual, we can write:

$$\|p_{k+1}\|^2 = {\beta_{k+1}}^2 \|p_k\|^2 - \|g_{k+1}\|^2 - 2{p_{k+1}}^T g_{k+1}.$$

Dividing across by $({p_{k+1}}^T g_{k+1})^2$ and using

$$\begin{aligned} \beta_{k+1} &= \xi_{k+1} \frac{{p_{k+1}}^T g_{k+1}}{p_k^T g_k} \\ {p_{k+1}}^T g_{k+1} &= \eta_{k+1} \|g_{k+1}\|^2, \end{aligned}$$

we have

$$\frac{\|p_{k+1}\|^2}{({p_{k+1}}^T g_{k+1})^2} = \xi_{k+1}^2 \frac{\|p_k\|^2}{(p_k^T g_k)^2} - \frac{1}{\eta_{k+1}^2} \frac{1}{\|g_{k+1}\|^2} - \frac{2}{\eta_{k+1}} \frac{1}{\|g_{k+1}\|^2}$$

Re-arranging the second and third terms in the RHS we can write

$$\frac{\|p_{k+1}\|^2}{(p_{k+1}{}^T g_{k+1})^2} = \xi_{k+1}^2 \frac{\|p_k\|^2}{(p_k^T g_k)^2} + \frac{1}{\|g_{k+1}\|^2}\left[1 - \left(1 + \frac{1}{\eta_{k+1}}\right)^2\right]$$

$$\leq \xi_{k+1}^2 \frac{\|p_k\|^2}{(p_k^T g_k)^2} + \frac{1}{\|g_{k+1}\|^2}\,. \tag{A.68}$$

We need to show that $|\xi_{k+1}| \leq 1$.

- Remember that $\xi_{k+1} = \frac{r_{k+1}}{1+(r_{k+1}-1)l_k}$.
- We have already seen (A.67) that (with $c \equiv (1-c_2)/(1+c_2)$;

$$1 + (r_{k+1} - 1)l_k \geq c.$$

- Also $r_{k+1} \geq -c$ so $c \geq -r_{k+1}$.
- Therefore

$$1 + (r_{k+1} - 1)l_k \geq -r_{k+1}.$$

- As $l_k \leq 1$ and $r_{k+1} \leq 1$, we have $(1 - r_{k+1})(1 - l_k) \geq 0$ and so

$$1 + (r_{k+1} - 1)l_k \geq r_{k+1}.$$

- Combining these two lower bounds for $1 + (r_{k+1} - 1)l_k$ we have

$$|1 + (r_{k+1} - 1)l_k| \geq |r_{k+1}|.$$

Therefore $|\xi_{k+1}| \leq 1$ so from (A.68) we can derive the inequality

$$\frac{\|p_{k+1}\|^2}{({p_{k+1}}^T g_{k+1})^2} \leq \frac{\|p_k\|^2}{(p_k^T g_k)^2} + \frac{1}{\|g_{k+1}\|^2}. \tag{A.69}$$

As in the proof of convergence for the DY-algorithm, we can iterate this inequality, giving

$$\frac{\|p_{k+1}\|^2}{({p_{k+1}}^T g_{k+1})^2} \leq \sum_{i=0}^{k+1} \frac{1}{\|g_i\|^2}.$$

and using the assumption $\|g_k\| > \gamma$ for all $k \geq 0$ we have

$$\frac{({p_{k+1}}^T g_{k+1})^2}{\|p_{k+1}\|^2} \geq \frac{\gamma^2}{k+2}.$$

Therefore the sum $\sum_{k=0}^{\infty} \frac{({p_k}^T g_k)^2}{\|p_k\|^2}$ diverges which contradicts Zoutendijk's Theorem 4.3. So $\liminf \|g_{k+1}\| = 0$ as required.

$\blacksquare$

**Hybrid Method — Numerical Behaviour** The method does not "stick": we show this using a similar argument to that for the DY method

- We have $\xi_{k+1} = \frac{r_{k+1}}{1+(r_{k+1}-1)l_k}$.
- For any $r_{k+1} \in [-c, 1]$, the denominator satisfies $1 + (r_{k+1} - 1)l_k \geq c$ — see (A.67).
- So $|\xi_{k+1}| \leq |r_{k+1}|/c$ ($r_{k+1}$ may be negative).
- As previously (A.68) we have

$$\frac{\|p_{k+1}\|^2}{(p_k^T g_k)^2} \leq \xi_{k+1}^2 \frac{\|p_k\|^2}{(p_k^T g_k)^2} + \frac{1}{\|g_{k+1}\|^2} \tag{A.70}$$

- Therefore, multiplying across by $\|g_{k+1}\|^2$:

$$\frac{1}{\cos^2\theta_{k+1}} \leq \xi_{k+1}^2 \frac{\|g_{k+1}\|^2}{\|g_k\|^2} \frac{1}{\cos^2\theta_k} + 1 \tag{A.71}$$

- Finally, using the upper bound above for $\xi_{k+1}$, we have (as $\xi_{k+1}^2 < r_{k+1}^2/c^2$)

$$\frac{1}{\cos^2\theta_{k+1}} \le \frac{1}{c^2} r_{k+1}^2 \frac{\|g_{k+1}\|^2}{\|g_k\|^2} \frac{1}{\cos^2\theta_k} + 1 \tag{A.72}$$

- Also, for the Hybrid methd, $r_{k+1} \equiv \frac{\beta_{k+1}^{H}}{\beta_{k+1}^{DY}}$ is just

$$r_{k+1} = \max\{0, \min\{\frac{g_{k+1}^T y_k}{\|g_{k+1}\|^2}, 1\}\} \tag{A.73}$$

- So, for the Hybrid method, $r_{k+1}$ is small precisely when $g_k \approx g_{k+1}$ or equivalently $y_{k+1} \approx 0$ and a “bad” $\theta_{k+1}$ gives $\cos\theta_{k+1} >> \cos\theta_k$ as $\xi_{k+1} \approx 0$ when $g_{k+1} \approx g_k$. So no “sticking”.

## A.19 Matrix Norms

**Definition A.6** *The norm of a matrix (not necessarily square) $A$ is defined in terms of a corresponding vector norm:*

$$\|A\| = \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|} \equiv \sup_{\|x\|=1} \|Ax\| \tag{A.74}$$

*as any scalar factor in $x$ can be cancelled above and below.*

For the familiar choice $\|x\| = \|x\|_2 \equiv \sqrt{\sum x_i^2}$, we need to show that

$$\|A\| = \max_{i=1,\dots,n} \sqrt{\Lambda_i} \tag{A.75}$$

where $\Lambda_i$ are the eigenvalues of the symmetric positive semidefinite matrix $A^T A$ as mentioned on Slide 123.

When the matrix $A$ is itself symmetric,

$$\|A\| = \max_{i=1,\dots,n} |\lambda_i| \tag{A.76}$$

where $\lambda_i$ are the (real) eigenvalues of the matrix $A$.

Let $R(x) = \dfrac{\|Ax\|^2}{\|x\|^2}$. Obviously

$$R(x) = \frac{x^T A^T Ax.}{x^T x}$$

We need to show that the largest value $R(x)$ can have is $\Lambda_1$ (provided that $\|x\| = 1$).

We use the notation $\{a_1, \dots, a_n\}$ for the eigenvectors of $A^T A$ corresponding to the eigenvalues $\Lambda_1 \geq \Lambda_2 \geq \cdots \geq \Lambda_n$. As $A^T A$ is symmetric, the $\Lambda_i$ are real and the corresponding eigenvectors $a_i$ are orthogonal and can be taken to be orthonormal. In fact the $\Lambda_i$ are non-negative (check).

Using the orthonormality of the $a_i$ we can write any vector $x = \sum x_i a_i$ and

$$R(x) = \frac{\sum_i \Lambda_i x_i^2}{\sum_j x_j^2}. \tag{A.77}$$

Now differentiate $R(x)$ wrt $x_k$ (say) and set the result to zero to find extremal values of $R(x)$:

$$2x_k \left( \|x\|^2 \Lambda_k - \sum_j \Lambda_j x_j^2 \right) = 0$$

Let $x^*$ be any $x \in \mathbb{R}^n$ that satisfies this equation. We want the supremum of $R(x)$ over non-zero $x$ so at least one of the $x_k^*$, say $x_p^* \neq 0$. So

$$R(x^*) = \Lambda_p$$

for any stationary value $x^*$. The choice of $x^*$ that maximises $R(x)$ is one that has $x_1^* \neq 0$ and in this case $R(x^*) = \Lambda_1$.

So

$$\|A\| \equiv \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|} \equiv \sup_{x \neq 0} \sqrt{R(x)} = \Lambda_1.$$

Finally, if $A$ is symmetric then $\Lambda_i = \lambda_i^2$, where $\lambda_i$ are the eigenvalues of $A$ so

$$\|A\| \equiv \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|} \equiv \sup_{x \neq 0} \sqrt{R(x)} = |\lambda_1|.$$

■

## A.20 Sherman-Morrison-Woodbury Formula

Just multiply the RHS of (6.20) by $(A + RST^T)$. In other words, RTP that

$$\left(A^{-1} - A^{-1}RU^{-1}T^TA^{-1}\right)\left(A + RST^T\right) = I.$$

Multiplying out this matrix product, it equates to $I + T_2$

where

$$T_2 = A^{-1}RST^T - A^{-1}RU^{-1}T^T - A^{-1}RU^{-1}T^TA^{-1}RST^T.$$

Now, RTP that $T_2 = 0$. But

$$T_2 = A^{-1}R\left[S - U^{-1} - U^{-1}T^TA^{-1}RS\right]T^T.$$

The factor is square brackets is zero iff $S = U^{-1}(I + T^TA^{-1}RS$ but this is true by definition of $U$.

## A.21 Derivation of Inverse DFP Formula

The DFP update formula (6.13)

$$H_{k+1} = (I - \gamma_k y_k s_k^T) H_k (I - \gamma_k s_k y_k^T) + \gamma_k y_k y_k^T,$$

may be written as $H_{k+1} = H_k + \Delta H_k$, where

$$\Delta H_k = R S R^T,$$

where

$$R = \begin{bmatrix} y_k & H s_k \end{bmatrix}, \quad S = \gamma_k \begin{bmatrix} 1 + \gamma_k s_k^T H_k s_k & -1 \\ -1 & 0 \end{bmatrix}.$$

A direct application of the SMW formula gives

$$(H_{k+1} + \Delta H_k)^{-1} = H_{k+1}{}^{-1} - H_k^{-1} R U^{-1} R^T H_k^{-1}$$

where $U = S^{-1} + R^T H_k^{-1} R$. Now,

$$\begin{aligned} S^{-1} &= \frac{-1}{\gamma_k^2} \begin{bmatrix} 0 & \gamma_k \\ \gamma_k & \gamma_k + \gamma_k^2 s_k^T H_k s_k \end{bmatrix} \\ &= - \begin{bmatrix} 0 & s_k^T y_k \\ s_k^T y_k & s_k^T y_k + s_k^T H_k s_k \end{bmatrix}. \end{aligned}$$

It follows by direct substitution that the $2 \times 2$ matrix $U$ is given by:

$$U = \begin{bmatrix} y_k^T H_k^{-1} y_k & 0 \\ 0 & -s_k^T y_k \end{bmatrix}.$$

Assembling the pieces we find that

$$(H_{k+1}+\Delta H_k)^{-1} = {H_{k+1}}^{-1} - \left\{ \frac{1}{y_k^T H_k^{-1} y_k} H_k^{-1} y_k y_k^T H_k^{-1} - \frac{1}{s_k^T y_k} s_k s_k^T \right\}.$$

Writing $J_k \equiv H_k^{-1}$, we have generated (6.21) as required.

## A.22 Robustness of BFGS

In Subsection 6.2.1 we stated that BFGS tends to correct itself quickly if it encounters numerical problems while DFP will be slow to recover.

Our strategy in proving "robustness" will be to prove that when the eigenvalues of $H_k$ become large, these large eigenvalues will be reduced at the next iteration of BFGS. Similarly when the eigenvalues of $H_k$ become excessively small, these small eigenvalues will be increased at the next iteration of BFGS. This behaviour was not designed into the algorithm — an unexpected bonus.

The tools we will use to examine the eigenvalues of $H_k$ are the trace and determinant of the matrix. As part of the proof of convergence for BFGS we proved that the trace and determinant of $H_k$ are modified in a particular way after an update (Eq. 6.30 and Eq. 6.31):

$$\operatorname{trace} H_{k+1} = \operatorname{trace} H_k - \frac{\|H_k s_k\|^2}{s_k^T H_k s_k} + \frac{\|y_k\|^2}{y_K^T s_k} \tag{A.78}$$

and

$$\det H_{k+1} = \det H_k \left( \frac{y_k^T s_k}{s_k^T H_k s_k} \right). \tag{A.79}$$

We will make the same assumptions as in the statement of Thm 6.26 (convergence Theorem for BFGS), and will re-cycle some of the algebra from the proof.

First we need to show that the second term ($T_2 = -\dfrac{\|H_k s_k\|^2}{s_k^T H_k s_k}$, say) on the RHS of (A.78) has the desired "eigenvalue reduction" property.

Taylor's Theorem can be written as

$$f_{k+1} - f_k = g_k^T s_k + \frac{1}{2} s_k^T \nabla^2 f(\xi_k) s_k,$$

where $\xi_k$ is on the line segment from $x_k$ to $x_{k+1}$. From the first Wolfe condition (4.9a) we have that

$$f_{k+1} \le f_k + c_1 \alpha_k g_k^T p_k = f_k + c_1 g_k^T s_k.$$

Combining the latter two equations gives us:

$$(c_1 - 1) g_k^T s_k \ge \frac{1}{2} s_k^T \nabla^2 f(\xi_k) s_k \ge m \|s_k^2\|$$

using the lower bound on $z^T \nabla^2 f(x) z$ from Thm 6.26.

Now using (as usual)

$$\cos\theta_k = \frac{-g_k^T p_k}{\|g_k\|\|p_k\|} = \frac{-g_k^T s_k}{\|g_k\|\|s_k\|},$$

we can simplify the previous inequality to

$$\|s_k\| \leq d\|g_k\|\cos\theta_k, \quad d \equiv \frac{2(1-c_1)}{m}. \tag{A.80}$$

Now to use these results to simplify $T_2$. As $H_k p_k = -g_k$ it follows that $H_k s_k = -\alpha_k g_k$ so

$$\begin{aligned} \frac{\|H_k s_k\|^2}{s_k^T H_k s_k} &= \frac{{\alpha_k}^2 \|g_k\|^2}{\alpha_k \|s_k\| \|g_k\| \cos\theta_k} \\ &= \frac{\alpha_k \|g_k\|}{\|s_k\| \cos\theta_k} \qquad \text{(A.81)} \\ &\geq \frac{\alpha_k \|g_k\|}{d \|g_k\| \cos^2\theta_k} \qquad \text{using (A.80)} \\ &= \frac{\alpha_k}{d \cos^2\theta_k}. \end{aligned}$$

So $T_2$ in the trace inequality (A.78) above satisfies $T_2 \leq -\frac{\alpha_k}{d \cos^2\theta_k}$.

We showed previously (6.29) in the convergence analysis for BFGS that

$$\frac{y_k^T y_k}{y_k^T s_k} \le M,$$

where $M$ is the positive constant referred to in (6.26) of Thm. 6.2. So the third term $T_3 = \frac{y_k^T y_k}{y_k^T s_k}$ in the trace inequality (A.78) is bounded above by $M$ and we can write

$$\operatorname{trace} H_{k+1} \le \operatorname{trace} H_k - \frac{\alpha_k}{d \cos^2 \theta_k} + M.$$

Now to use this result:

(i) Suppose that at the current iteration, we do **not** have $\cos\theta_k$ close to zero, then as we saw in Ch. 4 (in the discussion following Zoutendijk's Thm. 4.3) good progress can be made towards the minimum. However in this case the trace is likely to increase so some eigenvalues **may** become large as $T_2$ is likely to be much smaller in magnitude than $M$.

(ii) Large eigenvalues in $H_k$ correspond to $|\cos\theta_k| \ll 1$ as in (A.81) we saw that

$$\frac{\|H_k s_k\|^2}{s_k^T H_k s_k} = \frac{\alpha_k \|g_k\|}{\|s_k\| \cos\theta_k}$$

In this “problem” case progress will be slow as the gradient is almost perpendicular to the update direction $s$, (again, refer to the discussion following Zoutendijk’s Thm. 4.3). But $T_2 \leq -\frac{\alpha_k}{d \cos^2\theta_k}$ so $T_2$ is large and negative (and the “worse” $\cos\theta_k$ is, the more negative $T_2$ is) so the trace of $H_{k+1}$ will be much reduced from that of $H_k$ — so some or all of the large eigenvalues will be reduced to moderate values.

(iii) What if $\alpha_k \to 0$ — as will happen if $H_k$ is "nearly singular", resulting in $\|p_k\|$ becoming very large as $H_k p_k = -g_k$? ("Nearly singular" means one or more eigevalues are very small in magnitude.)

But this difficulty is also "fixed" by BFGS — as follows. We have (Eq. A.79 above)

$$\det H_{k+1} = \det H_k \left( \frac{y_k^T s_k}{s_k^T H_k s_k} \right).$$

If $s_k^T H_k s_k$ is "small" compared to $y_k^T s_k \equiv s_k^T \overline{H}_k s_k$ then $H_k$ has a small eigenvalue compared to those of the average Hessian $\overline{H}_k$— the "nearly singular" case. But this means that the factor multiplying $\det H_k$ is large, so the determinant of $H_k$ and therefore the eigenvalues are increased (in magnitude). So the small eigenvalue case is self-correcting just as the large eigenvalue case was.

It is an interesting Exercise to show (using a similar analysis) that, for the DFP algorithm (6.13), the $\frac{-1}{\cos^2\theta}$ term is missing from the trace update formula so excessively large eigenvalues are not reduced. (On the other hand, excessively small eigenvalues **are** increased.)

(See App. A.25 for some of the details.)

## A.23 Second-Order Necessary Conditions for Inequality-Constrained Problems

We begin with a definition:

**Definition A.7 (Tangent Cone)** *A vector* $\mathrm{d}$ *is a tangent vector to the feasible region* $\mathcal{F}$ *at a point* $x$ *if these is a sequence* $z_k$ *of feasible points (a "feasible sequence") approaching* $x$ *and a sequence of positive scalars* $\{t_k\}$ *with* $t_k \to 0$ *such that*

$$\lim_{k\to\infty} \frac{z_k - x}{t_k} = \mathrm{d}. \tag{A.82}$$

*The set of all tangents to* $\mathcal{F}$ *at* $x^*$ *is called the* **tangent cone** *and is written* $T_{\mathcal{F}}(x^*)$.

It is easy to see that the tangent cone is a cone in the technical sense that $v \in T_{\mathcal{F}}(x^*) \Rightarrow \alpha v \in T_{\mathcal{F}}(x^*)$ for all $\alpha > 0$. Setting the sequence $z_k \equiv x^*$ shows that $0 \in T_{\mathcal{F}}(x^*)$.

The definition of the tangent cone is not affected by the algebraic specification of the feasible region $\mathcal{F}$, only on its geometry. The feasible direction set $\mathcal{F}_1(x^*)$ however is a linearised description of the (algebraic) constraints at $x^*$ and so **is** dependent on the algebraic form of the constraints.

The tangent cone description is very useful when analysing the geometry of $\mathcal{F}$ near $x^*$ but the feasible direction set $\mathcal{F}_1(x^*)$ is easier to work with.

We will now prove that if the LICQ Def. 8.1 holds then the tangent cone at $x^*$ is equal to the set $\mathcal{F}(x^*, \lambda^*)$ of feasible directions defined in Def. 8.6. This result will allow us to prove Thm. 8.4.

**Lemma A.25** *Let $x^*$ be a feasible point. Then the following statements are true:*

1. $T_{\mathcal{F}}(x^*) \subseteq \mathcal{F}(x^*)$
2. *If the LICQ is satisfied at $x^*$ then $T_{\mathcal{F}}(x^*) = \mathcal{F}(x^*)$.*

**Proof:** Without loss of generality, assume that all the constraints are active at $x^*$ by simply dropping the inactive ones – they are irrelevant sufficiently close to $x^*$.

1. RTP that $\mathsf{T}_{\mathcal{F}}(\mathsf{x}^*) \subseteq \mathcal{F}(\mathsf{x}^*)$ so let $\mathbf{d} \in \mathsf{T}_{\mathcal{F}}(\mathsf{x}^*)$, we need to show that $\mathbf{d} \in \mathcal{F}(\mathsf{x}^*)$. As $\mathbf{d} \in \mathsf{T}_{\mathcal{F}}(\mathsf{x}^*)$ we have that $\lim\limits_{k \to \infty} \dfrac{z_k - x}{t_k} = \mathbf{d}$. or equivalently that

$$z_k = x^* + t_k d + o(t_k). \tag{A.83}$$

where the "little-oh" symbol is defined in Def. 3.11. For any $i \in \mathcal{E}$ we have that

$$\begin{aligned} 0 &= \frac{1}{t_k} c_i(z_k) \\ &= \frac{1}{t_k}\left(c_i(x^*) + t_k \nabla c_i(x^*)^T d + o(t_k)\right) \\ &= \nabla c_i(x^*)^T d + \frac{o(t_k)}{t_k}. \end{aligned}$$

Taking the limit as $k \to \infty$ the last term goes to zero so $\nabla c_i(x^*)^T d$ for $i \in \mathcal{E}$.

Now consider the active inequality constraints. Let $i \in \mathcal{I} \cap \mathcal{A}(x^*)$ then:

$$\begin{aligned} 0 &\leq \frac{1}{t_k} c_i(z_k) \\ &= \frac{1}{t_k}\left(c_i(x^*) + t_k \nabla c_i(x^*)^T d + o(t_k)\right) \\ &= \nabla c_i(x^*)^T d + \frac{o(t_k)}{t_k}. \end{aligned}$$

Again taking the limit as $k \to \infty$ the last term goes to zero so $\nabla c_i(x^*)^T d \geq 0$ for $i \in \mathcal{I} \cap \mathcal{A}(x^*)$ as required. So $d \in \mathcal{F}(x^*)$ and therefore $T_{\mathcal{F}}(x^*) \subseteq \mathcal{F}(x^*)$ as required.

2. Now RTP that if the LICQ holds then $\mathcal{F}(x^*) \subseteq T_{\mathcal{F}}(x^*)$ and so $\mathcal{F}(x^*) = T_{\mathcal{F}}(x^*)$. So we assume that $d \in \mathcal{F}(x^*)$ and seek to show that $d \in T_{\mathcal{F}}(x^*)$.

   As the LICQ holds, the $m \times n$ matrix $A(x^*)$ of active constraint gradients has full row rank $m= |\mathcal{A}(x^*)|$. Let $Z$ (as usual) be a matrix whose columns form a basis for the null space of $A(x^*)$ so that $Z \in \mathbb{R}^{n\times(n-m)}$ and $A(x^*)Z = 0$.

   Let $d \in \mathcal{F}(x^*)$ and suppose that $\{t_k\}_{k=0}^{\infty}$ is any sequence of positive reals such that $\lim_{k\to\infty} = 0$. Define a system of $n$ equations in $n$ unknowns $R(z, t) =$ (where $t$ is a scalar parameter) such that $R : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ by:

$$R(z, t) = \begin{bmatrix} c(z) - tA(x^*)d \\ Z^T(z - x^* - td) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \qquad \text{(A.84)}$$

We want to show that the solutions $z = z_k$ of this system of equations corresponding to (small) $t = t_k$ constitute a feasible sequence that approaches $x^*$ and satisfies the definition Def. A.82.

At $t = 0$, $z = x^*$ satisfies $R(z, t) = 0$ and the Jacobian of $R$ at this value of $(z, t)$ is

$$\nabla_x R(x^*, 0) = \begin{bmatrix} A(x^*) \\ Z^T \end{bmatrix}. \qquad \text{(A.85)}$$

This matrix is non-singular due to the definition of $Z$. So (by the Implicit Function Theorem) we can "solve for $z$" when $t$ is sufficiently close to zero. The I.F. Thm. guarantees that the solution exists and is unique when $t$ is sufficiently close to zero.

So $R(z, t_k) = 0$ has a unique solution $z_k$ for all sufficiently small values of $t_k$.

It follows from the definition of $R(z, t)$ (A.84) that we have $c(z_k) = t_k A(x^*)d$ for all $k$ sufficiently large and so as $d \in \mathcal{F}(x^*)$:

$$i \in \mathcal{E} \Rightarrow c_i(z_k) = t_k \nabla c_i(x^*)d = 0 \tag{A.86}$$

$$i \in \mathcal{I} \cap \mathcal{A}(x^*) \Rightarrow c_i(z_k) = t_k \nabla c_i(x^*)d \geq 0 \tag{A.87}$$

so certainly $z_k$ is feasible for all $k$ sufficiently large.

We need to check that (A.82) holds at $x^*$ for the feasible sequence $\{z_k\}$. Using the fact that $R(z_k, t_k) = 0$ for all $k$ sufficiently large together with Taylor's Theorem we have that:

$$
\begin{aligned}
0 = R(z_k, t_k) &= \begin{bmatrix} c(z_k) - t_k A(x^*)d \\ Z^T(z_k - x^* - t_k d) \end{bmatrix} \\
&= \begin{bmatrix} A(x^*)(z_k - x^*) + o(\|z_k - x^*\|) - t_k A(x^*)d \\ Z^T(z_k - x^* - t_k d) \end{bmatrix} \\
&= \begin{bmatrix} A(x^*) \\ Z^T \end{bmatrix} (z_k - x^* - t_k d) + o(\|z_k - x^*\|).
\end{aligned}
$$

If we divide the last equation across by $t_k$ and use the fact that the Jacobian (A.85) is non-singular we have that

$$\frac{z_k - x^*}{t_k} = d + o\left(\frac{\|z_k - x^*\|}{t_k}\right)$$

and so that (A.82) holds at $x^*$ for the feasible sequence $\{z_k\}$. Therefore $d \in T_{\mathcal{F}}(x^*)$ and so $T_{\mathcal{F}}(x^*) = \mathcal{F}(x^*)$ as required.

$\blacksquare$

We can now prove Thm. 8.4— the full Second-order Necessary conditions.

**Proof:** (of Thm. 8.4) Since $x^*$ is a local minimum, all feasible sequences $\{z_k\}$ approaching $x^*$ must have $f(z_k) \geq f(x^*)$ for all $k$ sufficiently large. We will construct a feasible sequence whose limiting direction is $w$ and show that the property $f(z_k) \geq f(x^*)$ implies that $w^T \nabla xx \mathcal{L}(x^*, \lambda^*) w \geq 0$.

Choose an arbitrary $w \in \mathcal{C}(x^*)$. Since $w \in \mathcal{C}(x^*) \subseteq \mathcal{F}(x^*) = T_{\mathcal{F}}(x^*)$ by Lemma A.25 we can use the technique in the Lemma to choose a sequence of positive reals $\{t_k\}$ and a feasible sequence $\{z_k\}$ converging to $x^*$ such that

$$\lim_{k \to \infty} \frac{z_k - x^*}{t_k} = w \tag{A.88}$$

or just

$$z_k - x^* = t_k w + o(t_k). \tag{A.89}$$

By the construction technique for the feasible sequence $\{z_k\}$, we have by (A.86) and (A.87) that

$$c_i(z_k) = t_k \nabla c_i(x^*) w, \quad \text{for all } i \in \mathcal{A}(x^*). \tag{A.90}$$

We can write:

$$\begin{aligned} \mathcal{L}(z_k, \lambda^*) &= f(z_k) - \sum_{i \in \mathcal{E} \cup \mathcal{I}} \lambda_i^* c_i(z_k) \\ &= f(z_k) - t_k \sum_{i \in \mathcal{A}(x^*)} \lambda_i^* \nabla c_i(x^*)^T w \\ &= f(z_k) \end{aligned} \tag{A.91}$$

We can also perform a Taylor series expansion to estimate $\mathcal{L}(z_k, \lambda^*)$ near $x^*$.

$$\begin{aligned}\mathcal{L}(z_k, \lambda^*) = \mathcal{L}(x^*, \lambda^*) + (z_k - x^*)^T \nabla \mathcal{L}(x^*, \lambda^*) \\ + \frac{1}{2}(z_k - x^*)^T \nabla^2 \mathcal{L}(x^*, \lambda)(z_k - x^*) \\ + o(\|z_k - x^*\|^2). \qquad (A.92)\end{aligned}$$

By complementarity (8.20e) we have $\mathcal{L}(x^*, \lambda^*) = f(x^*)$. By (8.23a), the gradient term in (A.92) is zero.

So using (A.91) we have

$$f(z_k) = f(x^*) + \frac{1}{2}(z_k - x^*)^T \nabla^2 \mathcal{L}(x^*, \lambda)(z_k - x^*) + o(t_k^2). \qquad (A.93)$$

Finally, if $w^T \nabla xx \mathcal{L}(x^*, \lambda^*) w < 0.$ then (A.93) implies that $f(z_k) < f(x^*)$ for all $k$ sufficiently large — contradicting the fact that $x^*$ is a local minimum. Therefore (8.35) must hold. ■

(Back to Thm. 8.4)

## A.24 Second-Order Sufficient Conditions for Inequality-Constrained Problems

Proof of Thm. 8.7. We repeat the statement of the Theorem for convenience.

**Theorem A.26 (Second-Order Sufficient Conditions)** *Suppose that for some feasible point* $x^* \in \mathbb{R}^n$ *there is a Lagrange multiplier vector* $\lambda^*$ *such that the KKT conditions (8.23a–8.20e) are satisfied. Suppose also that*

$$d^T \nabla_{xx} \mathcal{L}(x^*, \lambda^*) d > 0, \qquad \text{for all stationary directions } d \in \mathcal{C}(\lambda^*), d \neq 0. \tag{A.94}$$

*Then* $x^*$ *is a strict local solution for (8.1).*

**Proof:**

- First define $\bar{\mathcal{C}} = \{d \in \mathcal{C}(x^*, \lambda^*) | \|d\| = 1\}$.
- This is a closed and bounded subset of $\mathcal{C}(x^*, \lambda^*)$ so by (A.94) the minimiser of $d^T \nabla_{xx} \mathcal{L}(x^*, \lambda^*) d$ over this set is a strictly positive number, $\sigma$.
  - A continuous function of a closed and bounded subset of $\mathbb{R}^n$is bounded above and below and attains its upper and lower bounds.
- It is easy to check that if $d \in \mathcal{C}(x^*, \lambda^*)$ then so is $\alpha d$ for any $\alpha > 0$ (this is the defining property of a **cone**).
- It follows that $(d/\|d\|) \in \bar{\mathcal{C}}$ if and only if $d \in \mathcal{C}(x^*, \lambda^*)|$ for non-zero $d$. So (A.94) implies that

$$d^T \nabla_{xx} \mathcal{L}(x^*, \lambda^*) d \geq \sigma \|d\|^2, \quad \text{for all } d \in \mathcal{C}(x^*, \lambda^*). \qquad \text{(A.95)}$$

- We prove the Theorem by showing that every feasible sequence $\{z_k\}$ approaching $x^*$ has $f(z_k) \geq f(x^*) + (\sigma/2)\|z_k - x^*\|^2$, for all $k$ sufficiently large.
- Assume the opposite; that there is a feasible sequence $\{z_k\}$ approaching $x^*$ such that:

  $$f(z_k) < f(x^*) + (\sigma/2)\|z_k - x^*\|^2 \quad \text{for all } k \text{ sufficiently large} \tag{A.96}$$

  — we will show that this leads to a contradiction.
- The sequence $d_k = \dfrac{z_k - x^*}{\|z_k - x^*\|}$ is a bounded sequence in $\mathbb{R}^n$.
- As mentioned in Sec. 8.2.1, the Bolzano-Weirstrass Thm states that every sequence on a closed and bounded set has a convergent subsequence.
  - The set of vectors with unit norm is closed as well as bounded so the Bolzano-Weirstrass Thm applies.

- So $\{\mathrm{d}_k\}$ has a convergent subsequence converging to some vector $\mathbf{d}$ (say). If necessary re-numbering, we can write

$$\lim_{k\to\infty} \frac{z_k - x^*}{\|z_k - x^*\|} = \mathbf{d}. \tag{A.97}$$

- By Lemma A.25 we know that $T_{\mathcal{F}}(x^*) \subseteq \mathcal{F}(x^*, \lambda^*)$ and so $\mathbf{d} \in \mathcal{F}(x^*, \lambda^*)$.

- By definition of the Lagrangian and as the optimal multipliers are non-negative for $i \in \mathcal{I}$ and also as $c_i(z_k) \geq 0, i \in \mathcal{I}$ and $c_i(z_k) = 0, i \in \mathcal{E}$ we have

$$\mathcal{L}(z_k, \lambda^*) = f(z_k) - \sum_{i \in \mathcal{A}(x^*)} \lambda_i^* c_i(z_k) \leq f(z_k). \tag{A.98}$$

- Note that when (as in the proof for the necessary conditions) the LICQ holds, we have $\mathcal{L}(z_k, \lambda^*) = f(z_k)$.

- The Taylor series result from the proof for the necessary conditions still holds in the form:

$$\mathcal{L}(z_k, \lambda^*) = f(x^*) + \frac{1}{2}(z_k - x^*)^T \nabla^2 \mathcal{L}(x^*, \lambda^*)(z_k - x^*) + o(\|z_k - x^*\|^2). \tag{A.99}$$

- Now if $d \notin \mathcal{C}(x^*, \lambda^*)$ then for at least one $j \in \mathcal{I} \cap \mathcal{A}(x^*)$, we have

$$\lambda_j^* d^T \nabla c_j(x^*) > 0 \tag{A.100}$$

and $\lambda_i^* d^T \nabla c_i(x^*) \geq 0$ for all other $i \in \mathcal{I} \cap \mathcal{A}(x^*)$.

- We can expand $\lambda_j^* c_j(z_k)$ in a Taylor series:

$$\begin{aligned} \lambda_j^* c_j(z_k) &= \lambda_j^* c_j(x^*) + \lambda_j^*(z_k - x^*)^T \nabla c_j(x^*) + o(\|(z_k - x^*\|) \\ &= \|z_k - x^*\| \lambda_j^* d^T \nabla c_j(x^*) + o(\|(z_k - x^*\|). \end{aligned}$$

- This allows us to get an upper bound on the Lagrangian at $z_k$:

$$\begin{aligned} \mathcal{L}(z_k, \lambda^*) &\equiv f(z_k) - \sum_{i \in \mathcal{A}(x^*)} \lambda_i^* c_i(z_k) \\ &\leq f(z_k) - \lambda_j^* c_j(z_k) \\ &= f(z_k) - \|z_k - x^*\| \lambda_j^* d^T \nabla c_j(x^*) + o(\|(z_k - x^*\|). \end{aligned} \tag{A.101}$$

- Using (A.99) we can write a TS expansion for the Lagrangian at $z_k$:

  $$\mathcal{L}(z_k, \lambda^*) = f(x^*) + O(\|z_k - x^*\|^2)$$

  and combining this with (A.101) (flipping the inequality around):

  $$f(z_k) \geq f(x^*) + \|z_k - x^*\| \lambda_j^* d^T \nabla c_j(x^*) + o(\|(z_k - x^*\|). \quad (A.102)$$

- But in (A.96) we assumed that $f(z_k) < f(x^*) + (\sigma/2)\|z_k - x^*\|^2$ for all k sufficiently large.

- We can show these two inqualities are incompatible as follows.
  - Subtract (A.102) from (A.96): $0 < \|z_k - x^*\| \left((\sigma/2)\|z_k - x^*\| - \lambda_j^* d^T \nabla c_j(x^*)\right)$.
  - But for $k$ sufficiently large $(\sigma/2)\|z_k - x^*\| - \lambda_j^* d^T \nabla c_j(x^*) < 0$ — giving a contradiction.
  - We conclude that our assumption $d \notin \mathcal{C}(x^*, \lambda^*)$ is false, so $d \in \mathcal{C}(x^*, \lambda^*)$ and so $d^T \nabla_{xx} \mathcal{L}(x^*, \lambda^*) d \geq \sigma > 0$ from the discussion at the start of this proof.

- Finally we show that the assumption (A.96) yields a contradiction, even with $d \in \mathcal{C}(x^*, \lambda^*)$.
  - We will use the result $d^T \nabla_{xx} \mathcal{L}(x^*, \lambda^*) d \geq \sigma > 0$.

$$\begin{aligned}
f(z_k) &\geq \mathcal{L}(z_k, \lambda^*) \\
&= f(x^*) + \frac{1}{2}(z_k - x^*)^T \nabla^2 \mathcal{L}(x^*, \lambda^*)(z_k - x^*) + o(\|z_k - x^*\|^2) \\
&= f(x^*) + \frac{1}{2} d^T \nabla^2 \mathcal{L}(x^*, \lambda^*) d \|z_k - x^*\|^2 + o(\|z_k - x^*\|^2) \\
&\geq f(x^*) + (\sigma/2)\|z_k - x^*\|^2 + o(\|z_k - x^*\|^2).
\end{aligned}$$

- This latter inequality contradicts (A.96).

- So **every** feasible sequence $\{z_k\}$ approaching $x^*$ must satisfy $f(z_k) \geq f(x^*) + (\sigma/2)\|z_k - x^*\|^2 + o(\|z_k - x^*\|^2)$ for all $k$ sufficiently large.
- We conclude that $x^*$ is a strict local solution in the sense that $f(x) > f(x^*)$ for all feasible $x$ in some neighbourhood of $x^*$.

■

## A.25 Outline of Analysis for Robustness of DFP

Begin by showing that for for the DFP algorithm (6.13), we have

$$\mathrm{Tr} H_{k+1} = \mathrm{Tr} H_k - 2\gamma_k y_k H_k s_k + \gamma_k^2 s_k^T H_k s_k + \gamma_k \|y_k\|^2.$$

Referring to the terms on the RHS as $T_1, \cdots, T_4$ respectively we can find an upper bound for each (in magnitude).

It is easy to check that

$$\frac{\|y_k\|^2}{y_k^T s_k} \leq M$$

and that

$$\begin{aligned}\frac{s_k H_k s_k}{y_k^T s_k} &\le \frac{s_k H_k s_k}{(1-c_2)(-g_k^T s_k)} \quad (2nd.W.cond.)\\ &= \frac{(-\alpha_k) g_k^T s_k}{(1-c_2)(-g_k^T s_k)} \quad (\text{as } H_k s_k = -\alpha_k g_k.)\\ &= \frac{\alpha_k}{1-c_2}.\end{aligned}$$

Using these inequalities $T_4 \le M$ and $T_3 \le \frac{M\alpha_k}{1-c_2}$.

Finding an upper bound on the magnitude of $T_2$ is a bit trickier. First note that

- $m\|z\|^2 \leq z^T \nabla^2 f(x) z^{\leq} M\|z\|^2$ so taking $z = s_k$ and setting $x = x_k + \tau s_k$ and integrating from $\tau = 0$ to $1$

  $$m\|s_k\|^2 \leq s_k^T \bar{H}_k s_k \equiv s_k^T y_k \leq M\|s_k\|^2$$

  so $s_k^T y_k \geq m\|s_k\|^2$.

- $\|g_k\|\|s_k\| \cos\theta_k \equiv -g_k^T s_k \leq \frac{1}{1-c_2} y_k^T s_k$ (2nd. W. cond.) But we saw above that $s_k^T y_k \leq M\|s_k\|^2$ so $\|s_k\| \geq \mathcal{K}\|g_k\| \cos\theta_k$ where $\mathcal{K} = \frac{1-c_2}{M}$.

Now examine the magnitude of $T_2$:

$$\begin{aligned} |T_2| &= \frac{|y_k H_k s_k|}{y_k^T s_k} \quad (\text{as } y_k^T s_k > 0) \\ &\leq \frac{|y_k H_k s_k|}{m\|s_k\|^2} \quad (\text{using } s_k^T y_k \geq m\|s_k\|^2). \end{aligned}$$

Now consider the top line; $|y_k H_k s_k|$. Using $H_k s_k \equiv -\alpha_k g_k$ and the Cauchy-Schwartz inequality,

$$\begin{aligned} |T_2| &\leq \frac{|y_k H_k s_k|}{m\|s_k\|^2} \\ &\leq \frac{\alpha_k \|y_k\| \|g_k\|}{m\|s_k\|^2}. \end{aligned}$$

But $\|y_k\| \leq M\|s_k\|$ follows directly from $s_k^T y_k \leq M\|s_k\|^2$ above (why?) so

$$\begin{aligned} |T_2| &\leq \frac{\alpha_k \|y_k\| \|g_k\|}{m\|s_k\|^2} \\ &\leq \frac{\alpha_k M \|g_k\|}{m\|s_k\|} \\ &\leq \frac{\alpha_k M}{m\mathcal{K}\cos\theta_k} \quad \text{using } \|s_k\| \geq \mathcal{K}\|g_k\|\cos\theta_k \text{ above.} \end{aligned}$$

So, finally;

$$\mathrm{Tr}H_{k+1} \leq \mathrm{Tr}H_k + \frac{\alpha_k M}{m\mathcal{K}\cos\theta_k} + \frac{M\alpha_k}{1-c_2} + M$$

so that the trace of $H_k$ (and therefore the eigenvalues of $H_k$) may increase by up to $M$. The term $\frac{\alpha_k M}{m\mathcal{K}\cos\theta_k}$ that we might hope would **decrease** the trace of $H_k$ when $\cos\theta_k$ is small, is a bound on the **magnitude** of $T_2$. So $|T_2|$ may be large when $\cos\theta_k$ is small but as $T_2$ itself is of uncertain sign we cannot rely on $T_2$ to reduce large eigenvalues in $H_k$ when they occur. This is the major failing of the DFP method. (Compare with the discussion of BFGS on Slide 616.)

## A.26 The L-BFGS Algorithm

The method is unchanged from the standard BFGS method for the first $m$ iterations — except that we store

$$V_i \equiv (I - \gamma_i y_i s_i^T), \quad \gamma_i \quad \text{and} \quad s_i$$

at each iteration. The BFGS update rule can be written:

$$J_{k+1} = V_k^T J_k V_k + \gamma_k s_k s_k^T.$$

We need to see the effect of this succession of iterations. Given a starting value $J_0$, we can explicitly calculate the successive $J_i$:

$$\begin{aligned}
J_1 &= V_0^T J_0 V_0 + \gamma_0 s_0 s_0^T \\
J_2 &= V_1^T J_1 V_1 + \gamma_1 s_1 s_1^T \\
&= V_1^T V_0^T J_0 V_0 V_1 + \gamma_0 V_1^T s_0 s_0^T V_1 + \gamma_1 s_1 s_1^T \\
J_3 &= V_2^T J_2 V_2 + \gamma_2 s_2 s_2^T \\
&= V_2^T V_1^T V_0^T J_0 V_0 V_1 V_2 + \gamma_0 V_2^T V_1^T s_0 s_0^T V_1 V_2 + \gamma_1 V_2^T s_1 s_1^T V_2 + \gamma_2 s_2 s_2^T \\
&\vdots
\end{aligned}$$

and so

$$
\begin{aligned}
J_k =& V_{k-1}^T V_{k-2}^T \dots V_0^T J_0 V_0 V_1 \dots V_{k-2} V_{k-1} \\
& + \gamma_0 V_{k-1}^T V_{k-2}^T \dots V_1^T s_0 s_0^T V_1 \dots V_{k-2} V_{k-1} \\
& + \gamma_1 V_{k-1}^T V_{k-2}^T \dots V_2^T s_1 s_1^T V_2 \dots V_{k-2} V_{k-1} \\
& \vdots \\
& + \gamma_{k-3} V_{k-1}^T V_{k-2}^T s_{k-3} s_{k-3}^T V_{k-2} V_{k-1} \\
& + \gamma_{k-2} V_{k-1}^T s_{k-2} s_{k-2}^T V_{k-1} \\
& + \gamma_{k-1} s_{k-1} s_{k-1}^T. \qquad (A.103)
\end{aligned}
$$

After $m$ iterations we have:

$$\begin{aligned}
J_m =& V_{m-1}^T V_{m-2}^T \dots V_0^T J_0 V_0 V_1 \dots V_{m-2} V_{m-1} \\
&+ \gamma_0 V_{m-1}^T V_{m-2}^T \dots V_1^T s_0 s_0^T V_1 \dots V_{m-2} V_{m-1} \\
&+ \gamma_1 V_{m-1}^T V_{m-2}^T \dots V_2^T s_1 s_1^T V_2 \dots V_{m-2} V_{m-1} \\
&\vdots \\
&+ \gamma_{m-3} V_{m-1}^T V_{m-2}^T s_{m-3} s_{m-3}^T V_{m-2} V_{m-1} \\
&+ \gamma_{m-2} V_{m-1}^T s_{m-2} s_{m-2}^T V_{m-1} \\
&+ \gamma_{m-1} s_{m-1} s_{m-1}^T. \qquad \text{(A.104)}
\end{aligned}$$

After $m$ iterations we discard the oldest $V_k$, $s_k$ and $\gamma_k$ $(k = 0)$ and from now on we continue to only use the $m$ most recent $s$ and $y$ values. (Effectively we set $V_0 = I$ and $\gamma_0 = 0$ for the purposes of amending (A.104).)

So

$$\begin{aligned}
J_{m+1} =& V_m^T V_{m-1}^T \dots V_1^T J_0 V_1 V_2 \dots V_{m-1} V_m \\
& + \gamma_1 V_m^T V_{m-1}^T \dots V_2^T s_1 s_1^T V_2 \dots V_{m-1} V_m \\
& + \gamma_2 V_m^T V_{m-1}^T \dots V_3^T s_2 s_2^T V_3 \dots V_{m-1} V_m \\
& \vdots \\
& + \gamma_{m-2} V_m^T V_{m-1}^T s_{m-2} s_{m-2}^T V_{m-1} V_m \\
& + \gamma_{m-1} V_m^T s_{m-1} s_{m-1}^T V_m \\
& + \gamma_m s_m s_m^T.
\end{aligned}$$

and for any $k > m$ we have:

$$
\begin{aligned}
J_k =& V_{k-1}^T V_{k-2}^T \dots V_{k-m}^T J_0 V_{k-m} V_{k-m+1} \dots V_{k-2} V_{k-1} \\
&+ \gamma_{k-m} V_{k-1}^T V_{k-2}^T \dots V_{k-m+1}^T s_{k-m}^T s_{k-m} V_{k-m+1} \dots V_{k-2} V_{k-1} \\
&+ \gamma_{k-m+1} V_{k-1}^T V_{k-2}^T \dots V_{k-m+2}^T s_{k-m+1}^T s_{k-m+1} V_{k-m+2} \dots V_{k-2} V_{k-1} \\
&\vdots \\
&+ \gamma_{k-3} V_{k-1}^T V_{k-2}^T s_{k-3}^T s_{k-3} V_{k-2} V_{k-1} \\
&+ \gamma_{k-2} V_{k-1}^T s_{k-2}^T s_{k-2} V_{k-1} \\
&+ \gamma_{k-1} s_{k-1}^T s_{k-1}.
\end{aligned}
$$

or if you prefer

$$\begin{aligned}
J_{k+1} =& V_k^T V_{k-1}^T \dots V_{k-m+1}^T J_0 V_{k-m+1} V_{k-m+2} \dots V_{k-1} V_k \\
&+ \gamma_{k-m+1} V_k^T V_{k-1}^T \dots V_{k-m+2}^T s_{k-m+1}^T s_{k-m+1} V_{k-m+2} \dots V_{k-1} V_k \\
&+ \gamma_{k-m+2} V_k^T V_{k-1}^T \dots V_{k-m+3}^T s_{k-m+2}^T s_{k-m+2} V_{k-m+3} \dots V_{k-1} V_k \\
&\vdots \\
&+ \gamma_{k-2} V_k^T V_{k-1}^T s_{k-2}^T s_{k-2} V_{k-1} V_k \\
&+ \gamma_{k-1} V_k^T s_{k-1}^T s_{k-1} V_k \\
&+ \gamma_k s_k^T s_k. \qquad \text{(A.105)}
\end{aligned}$$

Despite its messy appearance, the above formula for $J_{k+1}$ is not that hard to program. It only requires that we accumulate successive values of $V_k$, $s_k$ and $\gamma_k$ at each iteration and that we discard the oldest values of $V_k$, $s_k$ for each iteration after $k = m$.

In practice we store $y_k$, $s_k$ as $V_k$ and $\gamma_k$ are easy to calculate from them. This greatly reduces the storage requirements to $2nm$ for $s$ and $y$. Also we only require $-J_{k+1}g$ so we only need to calculate a product of $V_i$'s, $V_i^T$'s and $s_i$'s with $g$ for each line in (A.105) and add them together. No $n \times n$ matrices needed — except $J_0$ which is usually a multiple of the identity matrix and so does not need to be stored.

**Exercise A.1** *Can you write down a formula for one of the lines of* (A.105) *times* $g$*?*

There is an ingenious variation on (A.105) that is more efficient in terms of storage and much simpler to program:

**Algorithm A.3 (L-BFGS (Inner Loop) )**

```
begin
    Given $J_0$ then at the $k^{th}$ iteration of the main algorithm
    if $k <= m$
        $\text{Bound} = k; \quad Y = [Y \quad y]; \quad S = [S \quad s]; \quad \Gamma = [\Gamma \quad \gamma];$
else
        $\text{Bound} = m;$
        $Y = [Y(:, 2 : m) \quad y];\ S = [S(:, 2 : m) s]; \quad \Gamma = [\Gamma(2 : m) \quad \gamma];$
    end;
    $q = g; \quad A = [\quad ];$
    We have augmented the $(n \times k - 1)$ matrices $S$, $Y$
    with an extra column $s_k$, $y_k$
     and the $k - 1$-dim vector $\Gamma$
     with an extra entry $\gamma_k$.
    When $k > m$ we discard the oldest columns of $s$ and $y$
    and the oldest entry of $\gamma$.
```

```
We now use the accumulated $s$, $y$ and $\gamma$-values
to accumulate a $k$-dim vector $A$ of $\alpha$-values
and calculate a $n$-dim vector $\mathsf{q}$
for   $i = \text{Bound} : -1 : 1$
begin
    $s = S(:, i)$;
    $y = Y(:, i)$;
    $\gamma = \Gamma(i)$;
    $\alpha = \gamma s^T \mathsf{q}$;
    $A = [\alpha \quad A]$;
    $\mathsf{q} = \mathsf{q} - \alpha y$;
end;
$r = J_0 \mathsf{q}$;
Now we use the accumulated $\alpha$-values in $A$
to calculate the final value of $r \equiv J_{k+1} g$.
```

```
    for $i = 1 : \text{Bound}$
    begin
        $s = S(:, i);$
        $y = Y(:, i);$
        $\alpha = A(i);$
        $\gamma = \Gamma(i);$
        $\beta = \gamma y^T r;$
        $r = r + (\alpha - \beta)s;$
    end;
    $p = -r;$
end
```

See App. A.27 for a justification of this algorithm and App. A.28 for a proof that it converges to a stationary point.

## A.27 Justification for the L-BFGS Algorithm

It is not at all obvious that the above L-BFGS algorithm is equivalent to (A.105).

This Section demonstrates the required equivalence by stepping through the pseudo-code of Alg. A.3. It is convenient to index the "stored" numbers $\gamma_{k-m} \dots \gamma_{k-1}$, vectors $s_{k-m} \dots s_{k-1}$ and matrices $V_{k-m} \dots V_{k-1}$ needed to compute $J_k g$ by the indices $1$ to $m$. So we will show that Alg. A.3 computes $J_{m+1} g$.

- Start by setting $Q = g$, the gradient $g_k$. For arbitrary $m \geq 1$:
- Step through Lines 21 to 26 of the Algorithm for $i$ from $i = m$ down to $i = 1$. ("Forward Pass")

$$\begin{aligned}
\alpha_m &= \gamma_m s_m^T Q \equiv \gamma_m s_m^T g \\
Q &= Q - \alpha_m y_m \equiv g - \gamma_m s_m^T g y_m = \left(I - \gamma_m y_m s_m^T\right) g \equiv V_m g \\
\alpha_{m-1} &= \gamma_{m-1} s_{m-1}^T Q \equiv \gamma_{m-1} s_{m-1}^T V_m g \\
Q &= Q - \alpha_{m-1} y_{m-1} \equiv Q - \gamma_{m-1} s_{m-1}^T V_m g \equiv V_m V_{m-1} g \\
&\vdots \\
Q &= V_2 \dots V_m g \\
\alpha_1 &= \gamma_1 s_1^T V_2 \dots V_m g \\
r_0 &= J_0 Q \equiv J_0 V_1 \dots V_m g.
\end{aligned}$$

- Step through Lines 33 to 38 of the Algorithm for $i$ from $i = 1$ up to $i = m$. ("BackwardPass") (We subscript successive values of $r$ for clarity — in fact each $r$ overwrites its predecessor.)

$$
\begin{aligned}
\beta &= \gamma_1 y_1^T r_0 \\
r_1 &= r_0 + (\alpha_1 - \beta) s_1 = r_0 + \left(\gamma_1 s_1^T V_2 \dots V_m g - \gamma_1 y_1^T r_0\right) s_1 \\
&= V_1^T r_0 + \gamma_1 s_1 s_1^T V_2 \dots V_m g \\
\beta &= \gamma_1 y_2^T r_1 \\
r_2 &= r_1 + (\alpha_2 - \beta) s_2 = r_1 + \left(\gamma_2 s_2^T V_3 \dots V_m g - \gamma_2 y_2^T r_1\right) s_2 \\
&= V_2^T r_1 + \gamma_2 s_2 s_2^T V_3 \dots V_m g \\
&\vdots \\
r_i &= V_i^T r_{i-1} + \gamma_i s_i s_i^T V_{i+1} \dots V_m g \\
&\vdots \\
r_{m-1} &= V_{m-1}^T r_{m-2} + \gamma_{m-1} s_{m-1} s_{m-1}^T V_m g \\
r_m &= V_m^T r_{m-1} + \gamma_m s_m s_m^T g
\end{aligned}
$$

- Finally, "back-substitute" for the $r_i$ to get an explicit expression for $r \equiv r_m$.

$$\begin{aligned} r \equiv r_m &= V_m^T r_{m-1} + \boldsymbol{\gamma_m s_m s_m^T g} \quad = V_m^T r_{m-1} + \mathbf{T_m} \\ &= \mathbf{V_m^T} \left(V_{m-1}^T r_{m-2} + \boldsymbol{\gamma_{m-1} s_{m-1} s_{m-1}^T V_m g}\right) + T_m \\ &= V_m^T V_{m-1}^T r_{m-2} + \mathbf{T_{m-1}} + T_m \\ &= V_m^T V_{m-1}^T \left(V_{m-2}^T r_{m-3} + \gamma_{m-2} s_{m-2} s_{m-2}^T V_{m-1} V_m g\right) \\ &\qquad + T_{m-1} + T_m \\ &\vdots \end{aligned}$$

So, by induction,

$$\begin{aligned} r \equiv r_m = V_m^T \dots V_1^T r_0 + \gamma_m s_m s_m^T g + \gamma_{m-1} V_m^T s_{m-1} s_{m-1}^T V_m g \\ + \gamma_{m-2} V_m^T V_{m-1}^T s_{m-2} s_{m-2}^T V_{m-1} V_m g + \\ + \dots + \gamma_1 V_m^T \dots V_2^T s_1 s_1^T V_2 \dots V_m \end{aligned}$$

which is (A.105) as claimed.

## A.28 Convergence of the L-BFGS Algorithm

Finally, we know a lot about the convergence properties of the BFGS method. can we say anything about those of L-BFGS?

We will be able to prove that L-BFGS has R-linear convergence — a slightly weaker version of linear convergence that we now define.

**Definition A.8 (R-linear)** *Let* $\{x_k\}$ *be a sequence in* $\mathbb{R}^n$ *that converges to* $x^*$ *. We say that the convergence is* **R-linear** *if there is a constant* $r \in (0,1)$ *such that*

$$\|x_k - x^*\| \leq \nu_k, \quad \textit{for all } k \textit{ sufficiently large, where } \nu_k \to 0 \textit{ linearly.} \tag{A.106}$$

This is a weaker property than conventional Q-linear convergence (as defined in (3.12)), i.e. linear convergence implies R-linear convergence but not vice versa.

**Example A.1** *The sequence*

$$x_k = \begin{cases} 1 + 2^{-k} & k \quad \textit{even} \\ 1 & k \quad \textit{odd.} \end{cases}$$

*converges* $\mathsf{R}$*-linearly to* $1$*. Note that this sequence does not have (*$\mathsf{Q}$*-)linear convergence as the "error" increases from* $0$ *at every second iteration.*

*Similarly the sequence*

$$x_k = 1 + 2^{-k}$$

converges linearly to $1$.

We can now state and prove a Theorem on the convergence properties of L-BFGS, subject to strong (restrictive) assumptions on $f$.

**Theorem A.27 (L-BFGS Convergence)** *Let $x_0$ be a starting point and let $f$ satisfy:*

- $f$ *is* $C^2$ *on* $\mathbb{R}^n$
- *The set* $D = \{x \in \mathbb{R}^n : f(x) \le f(x_0)\}$ *is convex*
- *There exist* $M_1, M_2$ *positive s.t.*

$$M_1 \|z\|^2 \le z^T \nabla^2 f(x) z \le M_2 \|z\|^2, \quad \textit{for all } z \in \mathbb{R}^n \textit{ and all } x \in D. \tag{A.107}$$

*Assume that the "starting estimates" of the Hessian and inverse Hessian at each value of* $k$ *before updating using the Alg. A.3 are bounded in norm. Then for any positive definite matrix* $B_0$*, the algorithm generates a sequence* $\{x_k\}$ *that converges to* $x^*$*.*

*Also there is a constant* $0 \le r < 1$ *such that*

$$f_k - f_* \le r^k (f_0 - f_*)$$

*so* $\{x_k\}$ *converges* R*-linearly (as defined above) to* $x^*$*.*

**Proof:** From the proof of Zoutendijk's Theorem 4.3 we have

$$f_{k+1} \le f_k - c\cos^2\theta_k \|\nabla fk\|^2,$$

for some $c > 0$ which may be taken as small as we wish wlog. So

$$(f_{k+1} - f_*) \le (f_k - f_*) - c\cos^2\theta_k \|\nabla fk\|^2. \tag{A.108}$$

Now RTP that

$$\|\nabla fk\|^2 \ge d\,(f_k - f_*) \tag{A.109}$$

for some positive constant $d$ — as this will imply that

$$(f_{k+1} - f_*) \le \left(1 - c'\cos^2\theta_k\right)(f_k - f_*) \tag{A.110}$$

for some positive constant $c'$.

We need to combine some results:

- Using Taylor's Thm.:

$$g_k \equiv g_k - g_* = \nabla^2 f(x_k + \tau(x_k - x^*))(x_k - x^*), \quad 0 \le \tau \le 1$$

  So $\|g_k\|^2 = (x_k - x^*)^T \nabla^2 f(x_k + \tau(x_k - x^*))(x_k - x^*)$. Using the lower bound on $z^T \nabla^2 f(x) z$, we have

$$\|g_k\|^2 \ge M_1 \|x_k - x^*\|^2.$$

- Again using Taylor's Thm.:

$$\begin{aligned} f_k - f_* &= (x_k - x^*)^T g_* + \frac{1}{2}(x_k - x^*)^T \nabla^2 f(x_k + \eta(x_k - x^*))(x_k - x^*) \\ &\le \frac{M_2}{2}\|x_k - x^*\|^2. \end{aligned}$$

- So $\|\nabla fk\|^2 \ge M_1 \|x_k - x^*\|^2 \ge 2\frac{M_1}{M_2^2}(f_k - f_*)$ which is (A.109) with $d = 2\frac{M_1}{M_2^2}$.

Substituting (A.109) in (A.108) gives (A.110) as required.

Now RTP that $\cos\theta_k > C > 0$ for all $k$ as this combined with Zoutendijk's Theorem implies that $x_k \to x^*$.

We need to examine L-BFGS again. Alg. A.3 computes $J_{k+1}$ in terms of $(s_k, y_k), (s_{k-1}, y_{k-1}), \ldots, (s_{k-m+1}, y_{k-m+1})$ and $J_0$. This is equivalent to a succession of $m$ iterations of the conventional BFGS algorithm using these $(s, y)$ pairs — starting with the same $J_0$.

As the algebra is easier for the inverse BFGS algorithm, we will work with this update rule. So we can represent the sequence of updates by:

$$H_k^{(l+1)} = H_K^{(l)} - \frac{H_K^{(l)} s_k^{(l)} s_k^{(l)T} H_K^{(l)}}{s_k^{(l)T} H_K^{(l)} s_k^{(l)}} + \frac{y_k^{(l)} y_k^{(l)T}}{s_k^{(l)T} s_k^{(l)}} \tag{A.111}$$

where $H_{k+1} \equiv H_k^{(m)}$ and $s_k^{(l)}$ and $y_k^{(l)}$ for $l = 1 \ldots m$ are the $l^{th}$ values of $s$ and $y$ stored at the $k^{th}$ iteration of the outer loop of the algorithm.

The results previously derived for the determinants and traces of $H_{k+1}$ may be recycled based on (A.111) above.

Certainly (neglecting the negative term in (6.30))

$$\mathrm{Tr}(H_k^{(l)}) \le \mathrm{Tr}((H_k^{(0)}) + \sum_{l=1}^{m} \frac{\|y_k^{(l)}\|^2}{y_k^{(l)T} s_k^{(l)}}. \qquad (A.112)$$

Also $\dfrac{\|y_k^{(l)}\|^2}{y_k^{(l)T} s_k^{(l)}} \le M$ for some positive constant $M$ by (6.29)so that

$$\mathrm{Tr}(H_{k+1} \le \mathrm{Tr}((H_k^{(0)}) + mM \le M_3, \quad \text{for some } M_3 > 0.$$

Also (relying on the result for $\mathrm{Det}(H_k)$ (6.31));

$$\mathrm{Det}(H_k^{(l+1)}) = \mathrm{Det}(H_k^{(l)}) \frac{y_k^{(l)T} s_k^{(l)}}{s_k^{(l)T} H_k^{(l)} s_k^{(l)}}$$

so

$$\begin{aligned}\mathrm{Det}(H_{k+1}) \equiv \mathrm{Det}(H_k^{(m)}) &= \mathrm{Det}(H_k^{(0)}) \prod_{l=1}^{m} \frac{y_k^{(l)T} s_k^{(l)}}{s_k^{(l)T} s_k^{(l)}} \prod_{l=1}^{m} \frac{s_k^{(l)T} s_k^{(l)}}{s_k^{(l)T} H_k^{(l)} s_k^{(l)}} \\ &\geq \mathrm{Det}(H_k^{(0)}) M_1^m M_3^m \equiv M_4, \text{ say,}\end{aligned}$$

where $M_1$ was defined in (A.107).

We have $\mathsf{Tr}(H_{k+1}) \le M_3$ and $\mathsf{Det}(H_{k+1}) \ge M_4$, so the least eigenvalue of $H_{k+1}$ is uniformly bounded below (by $m_0$, say) given that we know that the largest eigenvalue is bounded above (by $M_0$ say). (We justify this claim in Lemma A.28 below.)

So

$$\cos(\theta_k) \equiv \frac{s_k^T H_k s_k}{\|s_k\| \|H_k s_k\|} \ge \frac{m_0 \|s_k\|^2}{\|s_k\| \|H_k s_k\|}.$$

But $\||H_k s_k\| \le \|H_k\| \|s_k\| \le M_0 \|s_k\|$ which means that $\cos(\theta_k) \ge \frac{m_0}{M_0} = \mu > 0$ and therefore by Zoutendijk's Theorem $x_k \to x^*$.

Now finally, we have $\cos(\theta_k) \ge \mu > 0$ with $\mu < 1$ and also

$$f_{k+1} - f_* \le \left(1 - C\cos(\theta_k)^2\right)(f_k - f_*) \tag{A.113}$$

it follows that

$$1 - C\cos(\theta_k)^2 \le (1 - c\mu^2)(f_k - f_*).$$

Setting $r = 1 - C\mu^2 < 1$ and iterating (A.113) gives us

$$f_k - f_* \leq r^k(f_0 - f_*)$$

as required. $\blacksquare$

**Lemma A.28** *Given a sequence of positive definite matrices* $A_k$*, if* $\mathrm{Tr}A_k \le M_3$ *and* $\mathrm{Det}A_k \ge M_4$ *for some positive constants* $M_3$ *and* $M_4$ *for all* $k$ *then for all* $k$ *we have that the smalles eigenvalue* $\lambda_1^{(k)}$ *satisfies* $\lambda_1^{(k)} > m_0$ *for some positive constant* $m_0$*.*

**Proof:** Assume that there is a subsequence $\mathcal{K} \subseteq \mathbb{N}$ such that $\lambda_1^{(j)} < 1/j$ for all $j \in \mathcal{K}$ and seek a contradiction.( This is just the negation of $\lambda_1^{(k)} > m_0$ — check.) It follows that

$$\prod_{i=1}^{n} \lambda_i^{(j)} < \frac{1}{j} \prod_{i=2}^{n} \lambda_i^{(j)} \quad \text{for all } j \in \mathcal{K}.$$

Now using $\mathrm{Det}A_k \ge M_4$, we have

$$\prod_{i=2}^{n} \lambda_i^{(j)} \ge j M_4 \quad \text{for all } k \in \mathcal{K}.$$

But $\mathrm{Tr}A_k \le M_3 \Rightarrow \lambda_i^{(j)} \le M_3$ for all $k \in \mathcal{K}$.

So

$$\prod_{i=2}^{n} \lambda_i^{(j)} \leq M_3^{n-1} \equiv M_5 \quad \text{a positive constant, for all } j \in \mathcal{K}.$$

Therefore $M_5 \geq j M_4$ for all $j \in \mathcal{K}$ — which is false. ■

## A.29 Example With Two Inequality Constraints

Back to Slide 289

**Example A.2 (Two Inequality Constraints)** *Suppose we add an extra constraint to the problem* (8.11) *to obtain*

$$\min x_1 + x_2 \quad s.t. \quad 2 - x_1^2 - x_2^2 \geq 0, \quad x_2 \geq 0, \tag{A.114}$$

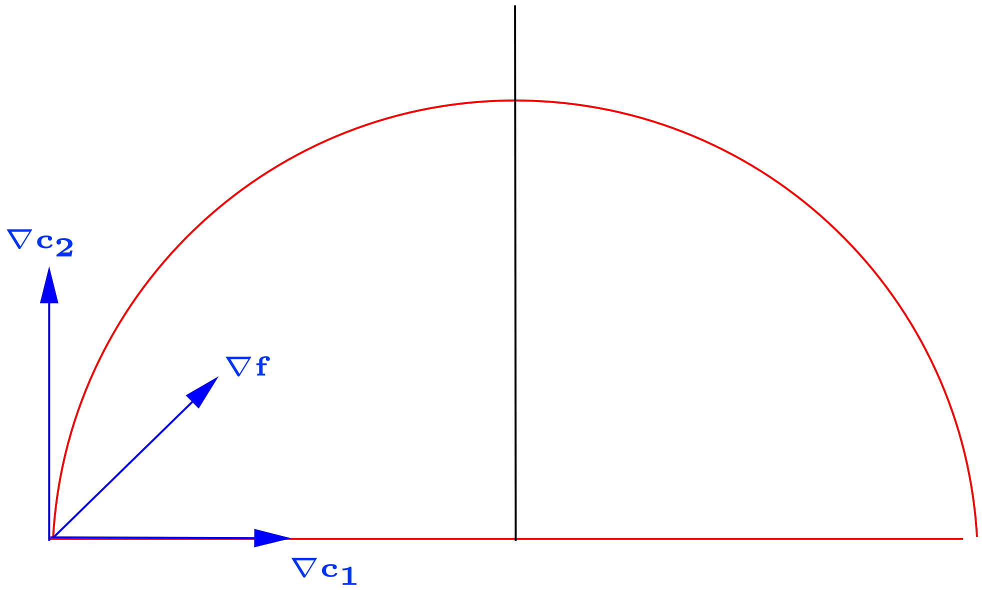


*Figure 32: Illustration of Example A.2*

*The feasible region is the half-disk illustrated in the Figure.*

*It is easy to see that the solution lies at* $(-\sqrt{2}, 0)^T$*, a point at which both constraints are active.*

- By repeating the arguments for the previous examples, we conclude that a direction $\mathbf{d}$ is a feasible descent direction, to first order, if it satisfies the following conditions:

$$\nabla c_i(x)^T d \geq 0, \quad i \in \mathcal{I} = \{1, 2\}, \quad \nabla f(x)^T d < 0. \tag{A.115}$$

- It is clear from Figure 32 that **no such direction exists** when $x = (-\sqrt{2}, 0)^T$.
- Or in other words, there are **no feasible descent directions** at this point — confirming that this point is a solution.
- The conditions $\nabla c_i(x)^T d \geq 0, i = 1, 2$, are both satisfied only if $\mathbf{d}$ lies in the quadrant defined by $\nabla c_1(x)$ and $\nabla c_2(x)$ but it is clear by inspection that all vectors $\mathbf{d}$ in this quadrant satisfy $\nabla f(x)^T d \geq 0$.

- How do the Lagrangian and its derivatives behave for this problem and the solution point $(-\sqrt{2}, 0)^T$?
- First, we include an additional term $\lambda_i c_i(x)$ in the Lagrangian for each additional constraint, so we have

$$\mathcal{L}(x, \lambda) = f(x) - \lambda_1 c_1(x) - \lambda_2 c_2(x), \tag{A.116}$$

  where $\lambda = (\lambda_1, \lambda_2)^T$ is the vector of Lagrange multipliers.

- The extension of condition (8.16) to this case is

$$\nabla_x \mathcal{L}(x^*, \lambda^*) = 0, \quad \text{for some} \quad \lambda^* \geq 0, \tag{A.117}$$

  where the inequality $\lambda^* \geq 0$ means that all components of $\lambda^*$ are required to be nonnegative.

- By applying the complementarity condition (8.17) to both inequality constraints, we obtain

$$\lambda_1^* c_1(x^*) = 0, \quad \lambda_2^* c_2(x^*) = 0. \tag{A.118}$$

- When $x^* = (-\sqrt{2}, 0)^T$, we have

$$\nabla f(x^*) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad \nabla c_1(x^*) = \begin{bmatrix} 2\sqrt{2} \\ 0 \end{bmatrix}, \quad \nabla c_2(x^*) = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

so that it is easy to verify that $\nabla_x \mathcal{L}(x^*, \lambda^*) = 0$ when we select $\lambda^*$ as follows:

$$\lambda^* = \begin{bmatrix} 1/(2\sqrt{2}) \\ 1 \end{bmatrix}. \tag{A.119}$$

- Note that both components of $\lambda^*$ are positive.
- We consider now some other feasible points that are **not** solutions of (A.114) and examine the properties of the Lagrangian and its gradient at these points.
- For the point $x = (\sqrt{2}, 0)^T$, we again have that both constraints are active.
- However, the objective gradient $\nabla f(x)$ no longer lies in the quadrant defined by the conditions $\nabla c_i(x)^T d \geq 0, i = 1, 2$ (see Figure 33).

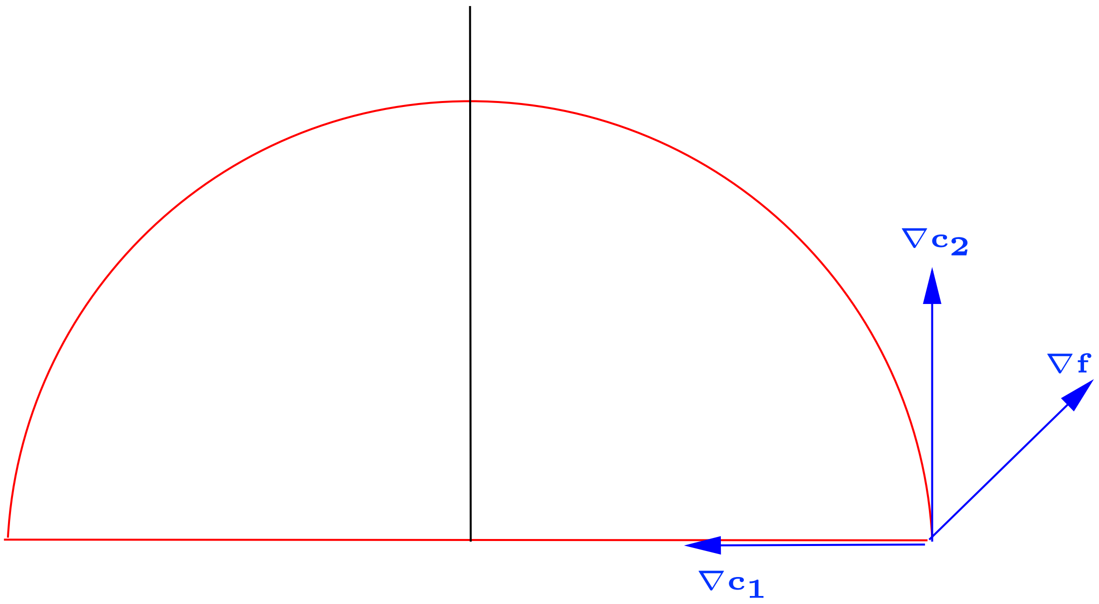


Figure 33: Illustration of Example A.2

- One first-order feasible descent direction from this point—a vector $d$ that satisfies (A.115) — is $d = (-1, 0)^T$; there are many others (see the exercises).
- For this value of $x$ it is easy to verify that the condition $\nabla_x \mathcal{L}(x, \lambda) = 0$ is satisfied only when $\lambda = (-1/(2\sqrt{2}), 1)^T$.

- Note that the first component $\lambda_1$ is negative, so that the conditions (A.117) are not satisfied at this point.
- Finally, consider the point $x = (1, 0)^T$, at which only the second constraint $c_2$ is active.
- At this point, linearisation of $f$ and $c$ as in Example 8.2 gives the following conditions, which must be satisfied for $d$ to be a feasible descent direction, to first order:

$$1 + \nabla c_1(x)^T d \geq 0, \quad \nabla c_2(x)^T d \geq 0, \quad \nabla f(x)^T d < 0. \quad \text{(A.120)}$$

- In fact, we need worry only about satisfying the second and third conditions, since we can always satisfy the first condition by making $d$ small enough in magnitude.

- By noting that

$$\nabla f(x) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad \nabla c_2(x) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

  it is easy to verify that the vector $d = (-\frac{1}{2}, \frac{1}{4})^T$ satisfies (A.120) and is therefore a descent direction.

- To show that optimality conditions (A.117) and (A.118) fail, we note first from (A.118) that since $c_1(x) > 0$, we must have $\lambda_1 = 0$.

- Therefore, in trying to satisfy $\nabla_x \mathcal{L}(x, \lambda) = 0$, we are left to search for a value $\lambda_2$ such that $\nabla f(x) - \lambda_2 \nabla c_2(x) = 0$.

- No such $\lambda_2$ exists — so this point fails to satisfy the optimality conditions.

Back to Slide 289

## A.30 Sensitivity Analysis for Inequality Constrained problems

- We are interested in the solutions of

  $$\min f(x) | c_i(x) = u_i, i \in \mathcal{E}$$

  which we can write as $\min f(x)$ such that $c(x) = u$, where $c(x)$ is the vector of equality constraints and $u$ the vector of perturbations.

- For $\|u\|$ sufficiently small it is reasonable to expect (and can be shown using the Implicit Function Theorem) that the solution $(x(u), \lambda(u))$ of the perturbed problem depends in a $C^1$ manner on $u$ and that the KKT equation continues to hold:

  $$\nabla_x f(x(u)) + \nabla_x c\,(x(u))\,\lambda(u) = 0. \tag{A.121}$$

  together with the perturbed constraints $c\,(x(u)) = u$.

- Taking the gradient w.r.t. $u$ of the perturbed constraints $c\left(x(u)\right) = u$ we have by the Chain Rule that:

$$\nabla_u c\left(x(u)\right) \equiv \nabla_x c\left(x(u)\right) \nabla_u x(u) = \nabla_u u \equiv I$$

- Now multiply the perturbed KKT eq. (A.121) across by $\nabla_u x(u)$:

$$\nabla_u x(u) \nabla_x f(x(u)) + \nabla_u x(u) \nabla_x c\left(x(u)\right) \lambda(u) = 0.$$

- Using $\nabla_x c\left(x(u)\right) \nabla_u x(u) = I$ we can write:

$$\nabla_u x(u) \nabla_x f(x(u)) + \lambda(u) = 0.$$

- So finally

$$\lambda(u) = -\nabla_u x(u) \nabla_x f(x(u))$$

and (again by the Chain Rule) (8.30) follows.

## A.31 The Bord Gáis Uplift Problem—Introduction

I briefly introduced this problem in the Introduction to these Notes. I'll repeat the preamble here for completeness before going into the details.

- Each day the Electricity Regulator assigns to each of the 100+ electricity generators large and small a production schedule for the next day based on projected demand.
- The is a (hard) mixed Integer Linear program.
  - Fortunately we don't need to solve it.
- The Regulator does this on the basis of (honestly) stated costs fixed and marginal costs provided by the generators.
- After each day (more or less) the Regulator is required to apply a correction to the half-hourly electricity price for the day to

ensure that all generators at least have their fixed and marginal costs met — as the actual amounts provided may not equal those specified in the schedule.

- The method for doing this is unusual!
- The Regulator solves a “Quadratic program” (QP):

$$\min_{\mathsf{UP}_h, h=1\ldots48} \mathsf{F}(\mathsf{UP}_h) \equiv \alpha \sum_h \left[ (\mathsf{SP}_h + \mathsf{UP}_h) \sum_g \mathsf{Q}_{gh} \right] + \beta \sum_h \mathsf{UP}_h^2 \tag{A.122}$$

subject to

$$\sum_h [(\mathsf{SP}_h + \mathsf{UP}_h)\mathsf{Q}_{gh}] \geq \mathsf{CR}_g, \text{for } g = 1\ldots\mathsf{G} \tag{A.123}$$

$$\mathsf{UP}_h \geq 0, \text{for } h = 1\ldots48 \tag{A.124}$$

where $\mathsf{G}$ is the number of generators, approximately $130$.

- The symbols:

- $\mathsf{SP}_h$ stands for the "Shadow prices", the marginal price of electricity per MWhr in each (half-)hour based on the information provided by the generators.
- $\mathsf{UP}_h$ stands for the "uplift", the correction factor that is applied retrospectively to the shadow prices.
- $\mathsf{Q}_{gh}$ stands for the volume (quantity) of electricity provided by each generator in each (half-)hour.

- The constraints ensure that:
  - The constraints ensure that each generator "covers their costs".
  - and that the uplifts are non-negative.
- The above can be written in a Matlab-like notation as

$$\min_{u \in \mathbb{R}^{48}} F(u) \equiv \alpha e^T Q(s+u) + \beta u^T u \tag{A.125}$$

where $e$ is a column vector of size $G$ of ones. and subject to

$$Q(s+u) \geq c \tag{A.126}$$

$$u \geq 0 \tag{A.127}$$

### A.31.1 Solving the QP

- The Regulator solves the QP — easily done in a fraction of a second — for example with Matlab.
- But Bord Gáis want to understand the solutions.

### A.31.2 KKT equations

- The first-order necessary condition for $x$ to be a local minimum of the unconstrained minimisation problem $\min\limits_{x \in \mathbb{R}} f(x)$ is that $f'(x) = 0$.

- The corresponding condition on $\mathbb{R}^n$ for $x$ to be a local minimum of the unconstrained minimisation problem $\min\limits_{x \in \mathbb{R}^n} f(x)$ is that the gradient (the vector of first partial derivatives) of $f$ is zero — $\nabla f(x) = 0$.
- The condition on $\mathbb{R}^n$ for $x$ to be a local minimum of the constrained minimisation problem $\min\limits_{x \in \mathbb{R}^n} f(x)$ is that the gradient of the Lagrangian $\mathcal{L}$ (a combination of $f$ and the constraints) is zero — $\nabla \mathcal{L}(x) = 0$.

### A.31.3 The KKT Equations for the QP

- Calculating the gradients wrt $u$ gives:

$$\alpha Q^T e + 2\beta u - Q^T \lambda - \mu = 0 \tag{A.128}$$

where $\lambda_g \geq 0$ for $g = 1 \dots G$ and $\mu_h \geq 0$ for $h = 1 \dots 48$ and also $\lambda_g$ and $\mu_h$ are zero when the corresponding constraints are inactive.

- It can be shown that the multipliers $\mu_h$ must all be zero.
- For a convex QP (which the present problem is) the KKT conditions are both necessary and sufficient for optimality — and the multipliers are unique.

### A.31.4 Analysis

- We can write (A.128) as

$$2\beta u = -\alpha Q^T e + Q^T \lambda. \tag{A.129}$$

- If (as the Regulator has decreed) $\alpha = 0$ and $\beta = 1$ if several generation constraints (say $g \in \mathcal{A}_{g_0}$) to bind then we have

$$u = \frac{1}{2} \sum_{g \in \mathcal{A}_{g_0}} \lambda_g \left(q_g\right). \tag{A.130}$$

- The $\lambda_g$ must (as part of the necessary conditions) all be non-negative.
- The binding constraints now take the form $\left(Qu\right)_g = c_g - \left(Qs\right)_g$ for $g \in \mathcal{A}_{g_0}$.

### A.31.5 Significance of the KKT necessary/sufficient condition

The optimal uplifts in any day must satisfy:

$$u = \frac{1}{2} \sum_{g \in \mathcal{A}_{g_0}} \lambda_g \left(q_g\right). \tag{A.131}$$

- So the optimal set of uplifts $u$ found by the Regulator **must** be a non-negative linear combination of the columns of $Q^T$ (rows of $Q$) corresponding to the binding generation constraints.
- Of course we don't know until the problem is solved **which** generation constraints are binding.
- So of what use is the result?
- We should check (over the year for which the system has been operating) **which** generation constraints have been binding in each 24-hour period.

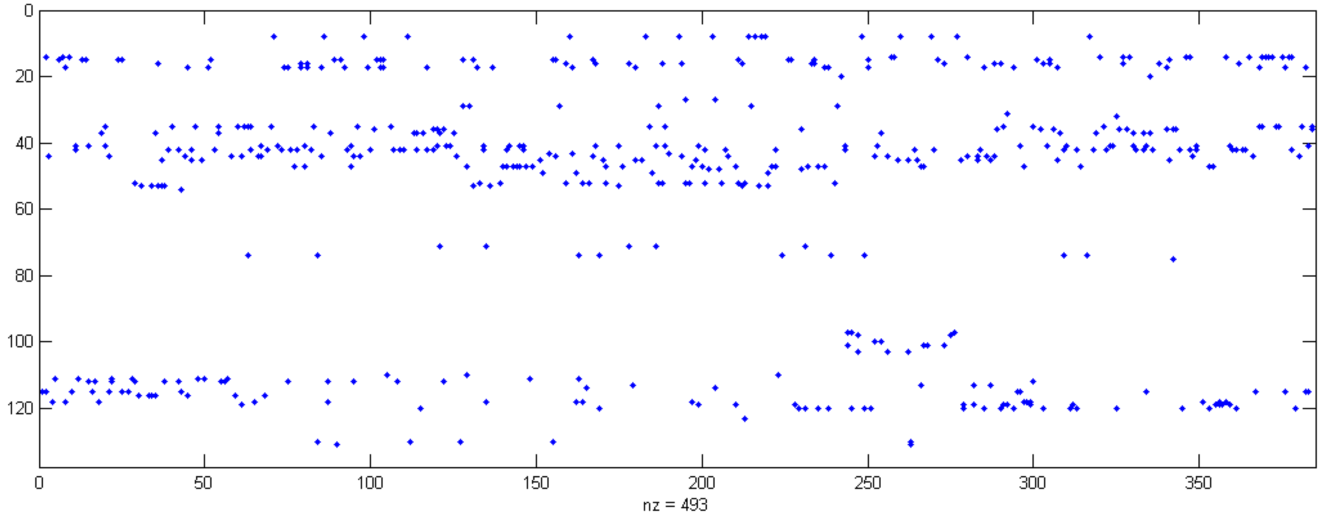
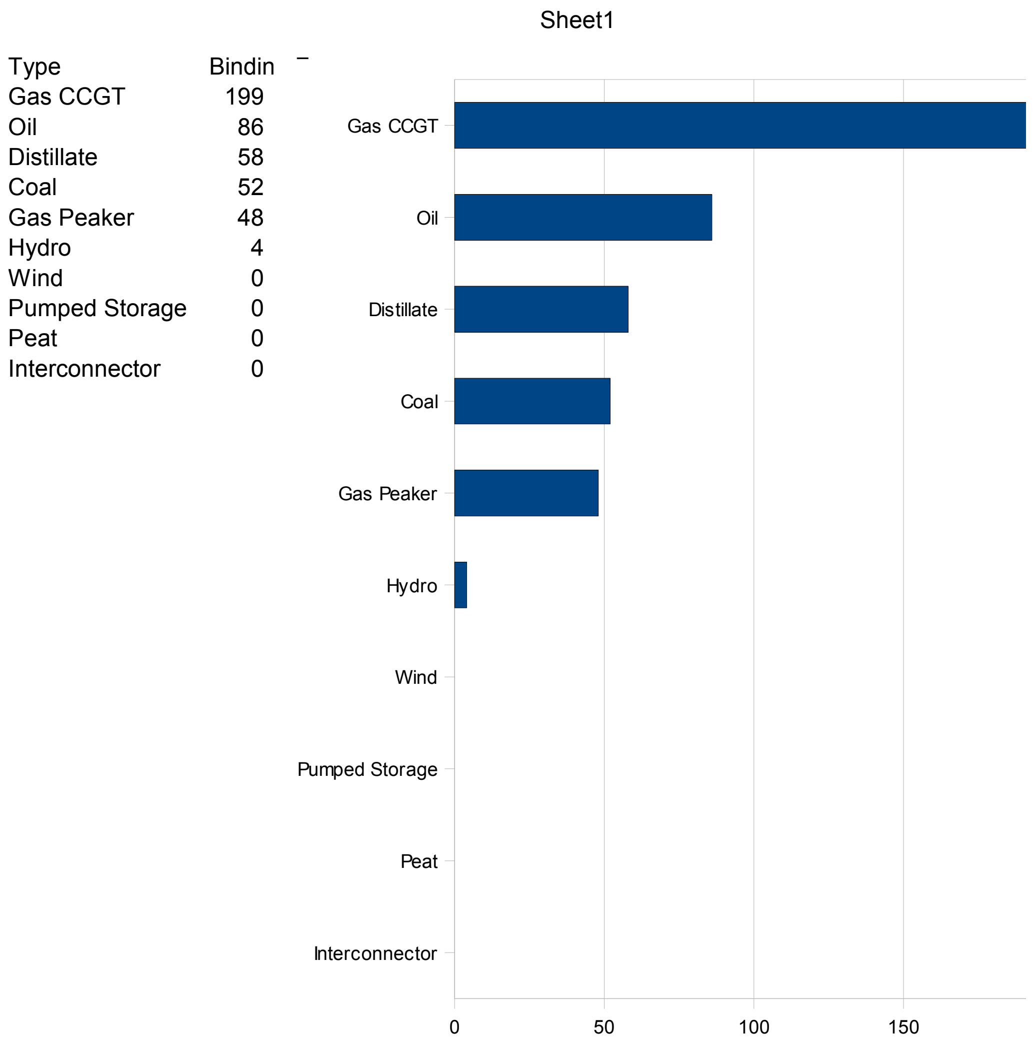


Figure 34: Recurrence of Binding Generators

- There is a very clear band structure in the Figure.

- A small subset of “usual suspect” generators are clearly binding from day to day.
- Which?

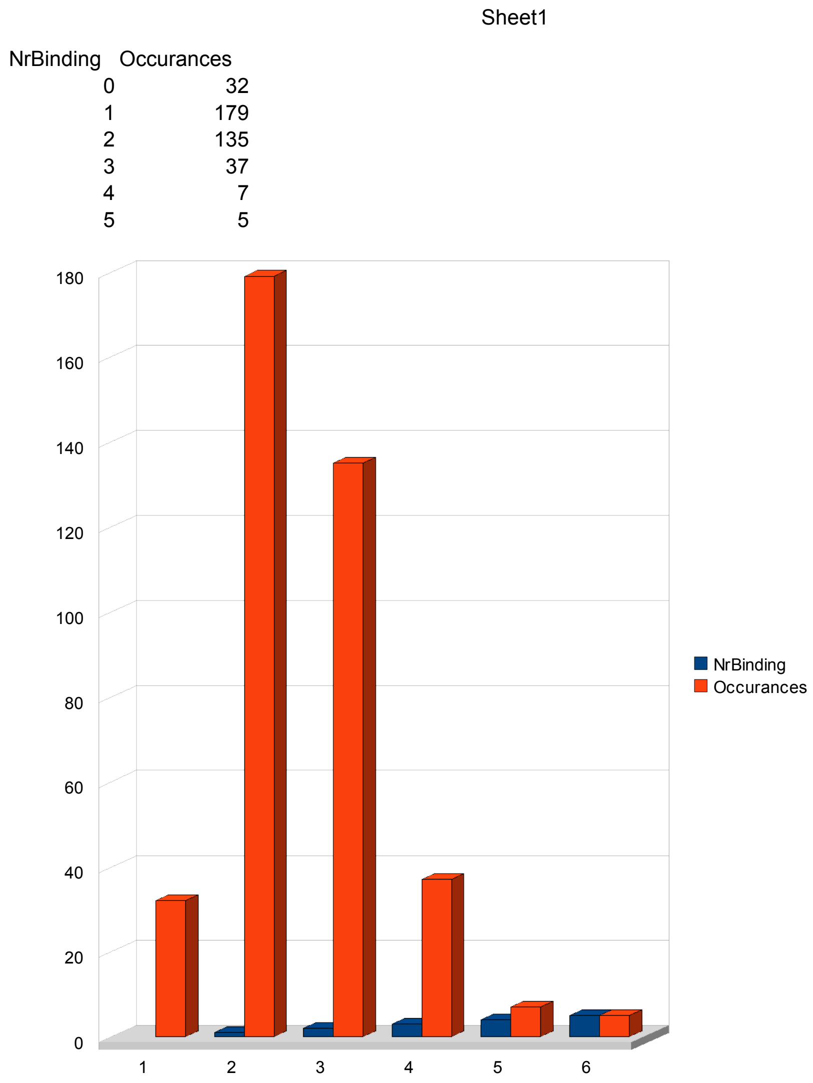


Sheet1

| Type | Bindin |
|---|---|
| Gas CCGT | 199 |
| Oil | 86 |
| Distillate | 58 |
| Coal | 52 |
| Gas Peaker | 48 |
| Hydro | 4 |
| Wind | 0 |
| Pumped Storage | 0 |
| Peat | 0 |
| Interconnector | 0 |

It appears that it is almost always the large thermal plants that have binding generation constraints.

- Of the small subset of generators that are clearly binding from day to day —
- How many generators are typically binding?



Sheet1

| NrBinding | Occurances |
|---|---|
| 0 | 32 |
| 1 | 179 |
| 2 | 135 |
| 3 | 37 |
| 4 | 7 |
| 5 | 5 |

### A.31.6 Preliminary Conclusions From KKT Analysis

Bord Gáis state:

- It (study) was a perfect example of how to apply a sophisticated mathematical technique to get a clear and clean explanation of something that was previously an opaque black box to us and to the market as a whole
- It was a real insight to learn that uplift is just a linear combination of the output of one or two generators on most days.
- And to classify which generators those were.