AN INTRODUCTION TO NON-LINEAR OPTIMIZATION

NOTES FOR MATH/CS 435/535, WWU

Edition: Winter, 2015

Steve McDowall

Version: Monday 23rd February, 2015

Contents

Chapter 1

Introduction

1.1 Examples

Constrained Optimization

1. Find the shortest ladder which can be leaned against a wall and have a box which is a m wide and b m tall fit underneath it. Note that the constraint can be considered an *inequality* constraint, though we can quickly argue that it could be changed to be an *equality* constraint.

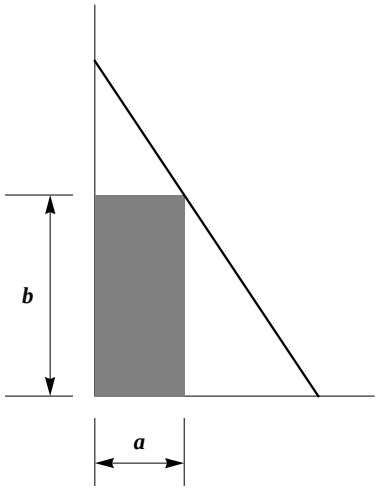


Figure 1.1: Shortest ladder problem.

2. A system of two scaffolds suspended by ropes is constructed, as in Figure 1.2 below. Loads x_1 and x_2 are to be positioned according to x_3 and x_4. The ropes A and B can

bear no more than 300 kg each, C and D can bear 200 kg, and E and F can bear 100 kg. Find the maximum load $x_1 + x_2$, and the corresponding positions x_3, x_4 that the system can bear. Physically, the sum of the vertical forces on each scaffold must be zero; if T_A denotes the tension (force) in rope A (etc.) then, for example, for the middle scaffold it must hold that

$$T_C + T_D = x_1 + T_E + T_F.$$

But, further, the sum of the *moments* must also be zero. A moment is defined to be the product of the force times the displacement (from any convenient choice of zero). Again, for the middle scaffold, this would mean (taking the zero to be at the left hand end of the scaffold)

$$10T_D = x_3 x_1 + 2T_E + 10T_F.$$

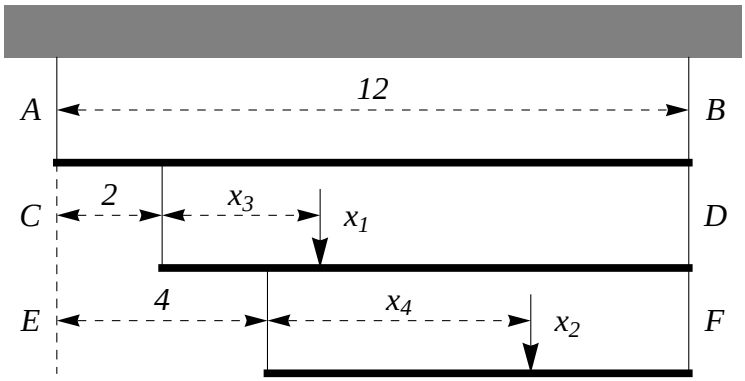


Figure 1.2: Scaffold problem.

3. A consumer who consumes two goods x_1 and x_2 has a utility function ("happiness" function) $u(x_1, x_2) = x_1 x_2$; if she has an income of I and the goods cost p_1 and p_2 then she would like to maximize u subject to $p_1 x_1 + p_2 x_2 \leq I$.

4. How should one divide their savings between three mutual funds with expected returns of 8%, 10% and 15% so as to minimize the *risk* while achieving an expected return of 11%. The *risk* is determined by considering the variance of the returns of each fund; if the funds are x, y and z, then the risk is found to be

$$r(x, y, z) = 400x^2 + 800y^2 + 200xy + 1600z^2 + 400yz.$$

Approximate Optimization

More often than not, optimization problems are so big, or so complex, that applying a theoretically sound procedure to find the optimum is practically infeasible. In other situations, the

quantity we wish to optimize has no closed-form expression; it can only be evaluated point by point. For these reasons we are interested in iterative algorithms which yield a sequence of approximate solutions, which hopefully converges to the true solution.

5. **What ellipsoid best approximates a box?**

   (a) Given a rectangle of dimensions $X \times Y$, what is the ellipse which best approximates the rectangle? This is not a well-defined question as it stands, so to be more precise, we wish to find the ellipse which minimizes the difference between the diameters of the ellipse and the dimensions of the rectangle, and simultaneously minimizes the difference between the areas of the ellipse and the rectangle. We can formulate this as: find (x^*, y^*) which minimizes

   $$F(x,y) = \big(x - \frac{1}{2}X\big)^2 + \big(y - \frac{1}{2}Y\big)^2 + \big(\pi xy - XY\big)^2.$$

   You will quickly discover that seeking the solution analytically is not tenable.

   In particular, the "most pleasing" rectangle is the rectangle whose dimensions have the ratio equal to the Golden ratio $\varphi = (1+\sqrt{5})/2$. So let us find the "most pleasing" ellipse by solving the above problem with $X = \varphi$ and $Y = 1$.

   (b) The three dimensional generalization of (a) is the following: Given a rectangular box of dimensions $X \times Y \times Z$, find the ellipsoid with radii x, y and z which simultaneously minimizes the differences between the spatial dimensions *and* the volumes. More specifically, find (x^*, y^*, z^*) which minimizes

   $$F(x,y,z) = \big(x - \frac{1}{2}X\big)^2 + \big(y - \frac{1}{2}Y\big)^2 + \big(z - \frac{1}{2}Z\big)^2 + \big(\frac{4}{3}\pi xyz - XYZ\big)^2.$$

6. In certain electronic devices, such as liquid crystal displays for example, the long axes of molecules are aligned, to a certain extent, with a preferred direction, $\vec{k}$ say. The process of alignment, however, is stochastic. Accordingly, there is a probability density function (pdf) describing the distribution of θ, the angle between the axis of the molecule and $\vec{k}$. One would like to be able to determine what this pdf is, and thus determine how well the molecules are aligned. Experimentally, one can measure either

   (a) S_2, the average (over many molecules) of $P_2(\cos\theta) = \dfrac{3}{2}\cos^2\theta - \dfrac{1}{2}$, or

   (b) S_2 *and* S_4, the average of $P_4(\cos\theta) = \dfrac{35}{8}\cos^4\theta - \dfrac{15}{4}\cos^2\theta + \dfrac{3}{8}$.

   The functions P_2 and P_4 are the 2^{nd} and 4^{th} degree *Legendre* polynomials. It must hold that $-0.5 \le S_2 \le 1.0$, and $\dfrac{35}{18}S_2^2 - \dfrac{5}{9}S_2 - \dfrac{7}{18} \le S_4 \le \dfrac{5}{12}S_2 + \dfrac{7}{12}$. An S_2 value of 1 (or close to 1) corresponds to very strong alignment with $\vec{k}$; $S_2 = 0$ indicates total randomness; and $S_2 = -0.5$ corresponds to strong alignment within the plane perpendicular to

$\vec{k}$. Given these measurements, there is a principle (called *maximum entropy*) which determines the "most likely" pdf that the data came from. In case (a), this requires finding c_2 so that

$$S_2 = \frac{\displaystyle\int_0^\pi P_2(\cos\theta)\exp\big[c_2P_2(\cos\theta)\big]\sin\theta\,d\theta}{\displaystyle\int_0^\pi \exp\big[c_2P_2(\cos\theta)\big]\sin\theta\,d\theta}, \tag{1.1}$$

in which case the pdf is

$$\theta \sim \frac{\exp\big[c_2P_2(\cos\theta)\big]}{\displaystyle\int_0^\pi \exp\big[c_2P_2(\cos\theta')\big]\sin\theta'\,d\theta'}.$$

In case (b) this requires finding c_2 and c_4 so that

$$S_2 = \frac{\displaystyle\int_0^\pi P_2(\cos\theta)\exp\big[c_2P_2(\cos\theta)+c_4P_4(\cos\theta)\big]\sin\theta\,d\theta}{\displaystyle\int_0^\pi \exp\big[c_2P_2(\cos\theta)+c_4P_4(\cos\theta)\big]\sin\theta\,d\theta}, \qquad \textit{and} \tag{1.2}$$

$$S_4 = \frac{\displaystyle\int_0^\pi P_4(\cos\theta)\exp\big[c_2P_2(\cos\theta)+c_4P_4(\cos\theta)\big]\sin\theta\,d\theta}{\displaystyle\int_0^\pi \exp\big[c_2P_2(\cos\theta)+c_4P_4(\cos\theta)\big]\sin\theta\,d\theta}, \tag{1.3}$$

and then

$$\theta \sim \frac{\exp\big[c_2P_2(\cos\theta)+c_4P_4(\cos\theta)\big]}{\displaystyle\int_0^\pi \exp\big[c_2P_2(\cos\theta')+c_4P_4(\cos\theta')\big]\sin\theta'\,d\theta'}.$$

Variational Problems

Here, instead of looking for an optimal vector $\vec{x}^* \in \mathbb{R}^n$ we are looking for an optimal *function* within some space/set of allowable functions.

7. **The Brachistochrone Problem:** a ball is to slide along a frictionless track from the point $(0, Y)$ to $(X, 0)$ affected only by gravity. What shape should the track be so as to minimize the time it takes? Here, the set of "allowable functions" will be ones for which their graph goes through the specified points.

8. **Design of a channel:** We wish to design the shape of a channel which has prescribed cross-sectional area A in such a way as to minimize the friction between the fluid flowing in the channel and the walls of the channel. Assuming that the total friction is proportional to the extent to which the fluid touches the walls, we wish to find the profile which minimizes the arc-length P of the cross-section, as indicated in Figure 1.3

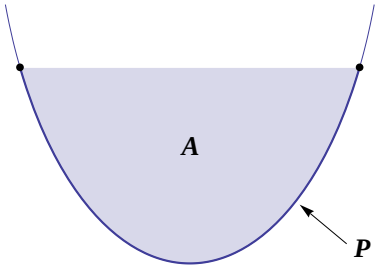


Figure 1.3: Channel

Dynamical Programming

These are discrete, sequential problems. For example, a traveler wants to get from city A to city B and has a variety of routes which go through various other cities; she knows the "cost" of traversing any given road between two joined cities and wants to find the "cheapest" route from A to B.

Optimal Control

You may think of these problems as continuous versions of dynamical programming problems. The continuous version of the example above might be the traveler can travel at a speed which is a function of their position, and so we seek the curve they should follow so as to minimize the time it takes to get to their destination. This is the principal that determines the path of light.

Example: Suppose we have a motorized vehicle whose movement we can control via its accelerator ($u > 0$) and brake ($u < 0$), and we are constrained by $-a \leq u \leq b$. What is the optimal (fastest) acceleration/deceleration procedure to get the object from point A to point B, starting and finishing at rest?

1.2 Terminology

- **Objective function:** the function $f(\vec{x})$, $f : \mathbb{R}^n \to \mathbb{R}$ which we want to optimize.
- **Constraints:** limitations/constraints on the independent variable $\vec{x}$.
  - **Constraint function(s):** the function $h : \mathbb{R}^n \to \mathbb{R}^m$ consisting of m functions $h_j : \mathbb{R}^n \to \mathbb{R}$ which describe the constraints.
  - **Feasible points:** vectors $\vec{x} \in \mathbb{R}^n$ which satisfy any constraints present.
- **Local minimizer:** a (feasible) vector $\vec{x}^* \in \mathbb{R}^n$ for which there is $\varepsilon > 0$ such that $f(\vec{x}^*) \leq f(\vec{x})$ for all *feasible* $\vec{x}$ with $|\vec{x} - \vec{x}^*| < \varepsilon$.

- **Global minimizer:** a (feasible) vector $\vec{x}^* \in \mathbb{R}^n$ for which $f(\vec{x}^*) \leq f(\vec{x})$ for all feasible $\vec{x}$.

1.3 Notation

A lot of notation will simply be introduced as we need it. For starters:

- Vectors $\vec{x} \in \mathbb{R}^n$ are considered to be *columns*. We write $\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$, or more efficiently $\vec{x} = [x_1, x_2, \cdots, x_n]^T$. In $\mathbb{R}^3$ we will often use $\vec{x} = [x, y, z]^T$.
- Vector inequalities: While it is arguably an abuse of notation, we will write $\vec{u} \leq \vec{v}$ to mean that $u_i \leq v_i$ for each component i of the vectors $\vec{u}$ and $\vec{v}$.
- The gradient vector for a function $f : \mathbb{R}^n \to \mathbb{R}$ at the point $\vec{x}$ is $\nabla f(\vec{x})$ (and is itself a column vector): $\nabla f(\vec{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$.
- (See B.7.) One of the most important objects we will use throughout the term is the **differential** of a vector valued function. If $h : \mathbb{R}^n \to \mathbb{R}^m$, then $Dh(\vec{x})$ is an $m \times n$ matrix (most easily remembered since also $Dh : \mathbb{R}^n \to \mathbb{R}^m$) consisting of all the first order derivatives of the component functions of h. Its *rows* are the transposes of the gradients:

$$Dh(\vec{x}) = \begin{bmatrix} \nabla h_1(\vec{x})^T \\ \nabla h_2(\vec{x})^T \\ \vdots \\ \nabla h_m(\vec{x}) \end{bmatrix} = \begin{bmatrix} \frac{\partial h_1}{\partial x_1} & \frac{\partial h_1}{\partial x_2} & \cdots & \frac{\partial h_1}{\partial x_n} \\ \frac{\partial h_2}{\partial x_1} & \frac{\partial h_2}{\partial x_2} & \cdots & \frac{\partial h_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial h_m}{\partial x_1} & \frac{\partial h_m}{\partial x_2} & \cdots & \frac{\partial h_m}{\partial x_n} \end{bmatrix}.$$

  Note: for a scalar valued function $f : \mathbb{R}^n \to \mathbb{R}$, the differential Df still makes sense; it is simply $Df(\vec{x}) = \nabla f(\vec{x})^T$. This is important to remember.
- An open ball of radius ε centered at $\vec{x}$ is denoted $B_\varepsilon(\vec{x})$. Precisely,

$$B_\varepsilon(\vec{x}) = \{\vec{y} \, : \, |\vec{x} - \vec{y}| < \varepsilon\}.$$

- If S is a (hyper) surface in $\mathbb{R}^n$ and $\vec{x} \in S$, then the tangent space to S at $\vec{x}$ is denoted $T_{\vec{x}}S$.
- If $A : V \to W$ is a linear operator between vector spaces V and W, then

1. by $\mathcal{N}(A)$ we mean the null-space of A, $\mathcal{N}(A) \subset V$;
2. by $\mathcal{R}(A)$ we mean the range of A, $\mathcal{R}(A) \subset W$.

1.4 Mathematical Background

Appendix B contains a collection of mathematical concepts we will encounter through the course. It's expected that you know all the items referenced to either (204) or (224). However, we will "remind" ourselves of topics along the way – be sure to speak up if there are things we are using which you do not properly understand! Other items will be addressed as we need them.

At the beginning of each section there will be a list of items from the appendix which you should review.

Chapter 2

Constrained Optimization

2.1 Constrained Optimization with equality constraints

Review topics:

- matrices (B.2.1, B.2.4, B.2.6, B.2.9);
- dot product and norm (B.3.1, B.3.3, B.3.4);
- geometry in $\mathbb{R}^n$ (B.6.2);
- calculus in $\mathbb{R}^n$ (B.7.1, B.7.3, B.7.4, B.7.5, B.7.7, B.7.8, B.7.9, B.7.12)

Consider as a reference example:
EXAMPLE 2.1. Optimize $f(x_1, x_2, x_3) = 3x_1x_3 + 4x_2x_3$ subject to $x_2^2 + x_3^2 = 4$ and $x_1x_3 = 3$.

Note: it is somewhat an abuse of notation to write $f(\vec{x})$ as $f(x_1, x_2, x_3)$; it should really be written as $f\left(\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}\right)$ (which is clearly cumbersome), or $f([x_1, x_2, x_3]^T)$. However, since there is no likelihood of confusion, we will allow ourselves to write $f(x_1, x_2, x_3)$.

2.1.1 Formulation

Formulation of the problem:

Minimize $f(\vec{x}) : \mathbb{R}^n \to \mathbb{R}$ subject to $h(\vec{x}) = \vec{0}$, where $h : \mathbb{R}^n \to \mathbb{R}^m$.

The function f is the *objective function*. The *constraint* $h(\vec{x}) = \vec{0}$ is really m constraints $h_j(\vec{x}) = 0$ which must be satisfied simultaneously.

In the reference Example 2.1, $f : \mathbb{R}^3 \to \mathbb{R}$, $h : \mathbb{R}^3 \to \mathbb{R}^2$,

$$h(\vec{x}) = \begin{bmatrix} h_1(\vec{x}) \\ h_2(\vec{x}) \end{bmatrix} = \begin{bmatrix} x_2^2 + x_3^2 - 4 \\ x_1 x_3 - 3 \end{bmatrix}$$

The above formulation includes the problem of maximization also: we can minimize $-f$ to find the maximum of f.

2.1.2 A brief review of introductory multi-variable calculus and Lagrange multipliers

- We consider an objective function $f : \mathbb{R}^n \to \mathbb{R}$; we wish to optimize f subject to *a collection of* constraints. Constraints are zero-sets of functions. Depending on dimension, these are level curves, level surfaces, or level hyper-surfaces.

- In Math 224, $n = 2$. It really only makes sense in this case to consider *one* constraint curve. If there was more than one, then either they don't intersect (then there is no feasible set); or they intersect at a point (or finitely many points) in which case we can just check these points; or they coincide over a curve, in which case there is really only one constraint.

  However, when $n > 2$, two surfaces can intersect in a curve, and two hyper-surfaces can intersect in a surface (etc.).

Perpendicular to a (hyper-) surface. Recall (at least in dimensions $n = 2, 3$) that (non-zero) gradient vectors are *perpendicular* to level curves, surfaces (hyper-surfaces). What does this mean? For a level curve, this means it is perpendicular to the *tangent line* to the curve at the point in question. For a level surface, the gradient vector is perpendicular to the *tangent plane* to the surface at the point.

More precisely, let $\vec{x}^*$ be a point where $h(\vec{x}^*) = 0$. To say that $\nabla h(\vec{x}^*)$ is perpendicular to the level set $h(\vec{x}) = 0$ at $\vec{x}^*$ means that if we take *any smooth curve* lying in the surface $h(\vec{x}) = 0$, going through $\vec{x}^*$, then $\nabla h(\vec{x}^*)$ is perpendicular to the *tangent vector* to the curve at $\vec{x}^*$.

- *Proof that gradient vectors are perpendicular to level sets.* We must start with an arbitrary smooth curve in the level set $h(\vec{x}) = 0$ going through $\vec{x}^*$: suppose $\vec{x}(t) : (-\varepsilon, \varepsilon) \to \mathbb{R}^n$ is a curve in $h(\vec{x}) = 0$ with $\vec{x}(0) = \vec{x}^*$ (it goes through $\vec{x}^*$ at $t = 0$), and with $\vec{x}\,'(0) \neq \vec{0}$. (Note: $\vec{x}\,'(0)$ is the tangent vector of the curve at $\vec{x}(0)$, i.e. at $\vec{x}^*$.) Then

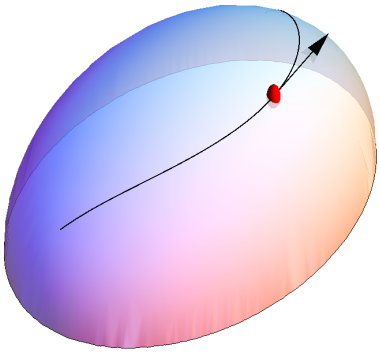


Figure 2.1: A tangent vector to a surface is a tangent vector to a curve which lies in the surface.

$h(\vec{x}(t)) = 0$ for all t, so by the chain rule (B.7.8),

$$0 = \frac{d}{dt} h(\vec{x}(t)) = \nabla h(\vec{x}(t))^T \vec{x}\,'(t).$$

In particular, $\nabla h(\vec{x}^*)$ is perpendicular to $\vec{x}\,'(0)$. □

EXAMPLE 2.2. Let $h(x, y) = x^2 + y^2 - 1$, so $S = \{h(\vec{x}) = 0\}$ is the unit circle in $\mathbb{R}^2$; let $\vec{x}^* = [\sqrt{3}/2, 1/2]^T \in S$. A family of parameterized curves $\vec{x}(t)$ lying in S is

$$\vec{x}(t) = \big[\cos(\alpha t + \pi/6), \sin(\alpha t + \pi/6)\big]^T.$$

Indeed, $\|\vec{x}(t)\| = 1$ and $\vec{x}(0) = \vec{x}^*$. Thus $\vec{x}\,'(0)$ is a tangent vector to S at $\vec{x}^*$:

$$\vec{x}\,'(0) = \alpha[-\sin(\pi/6), \cos(\pi/6)]^T = \alpha[-1/2, \sqrt{3}/2]^T.$$

(Varying α results in longer/shorter/reversed tangent vectors; all of them together give the whole *line* tangent to S at $\vec{x}^*$ (translated to the origin).)

Now $\nabla h(\vec{x}^*) = [\sqrt{3}, 1]^T$ and we see that $\nabla h(\vec{x}^*) \cdot \vec{x}\,'(0) = 0$ as claimed. (Note that this isn't a demonstration of the above proof, since we've taken *some* curves through the point, lying in S, but not *all* curves!) □

Keeping with the case of a single constraint, now suppose that $\vec{x}^*$ is a **minimizer of the constrained problem**. Again let $\vec{x}(t)$ be a parameterized curve lying in $S = \{h(\vec{x}) = 0\}$ as we used in the proof above. Then $f(\vec{x}(t))$ must have a minimum at $t = 0$ (do you see why?), so by single variable calculus (using the chain rule of multi-variable calculus),

$$0 = \frac{d}{dt} f(\vec{x}(t))\Big|_{t=0} = \nabla f(\vec{x}(0))^T \vec{x}\,'(0) = \nabla f(\vec{x}^*)^T \vec{x}\,'(0)$$

and we see that $\vec{x}\,'(0)$ is also perpendicular to $\nabla f(\vec{x}^*)$. (Recall from above that such an $\vec{x}\,'(0)$ is perpendicular to $\nabla h(\vec{x}^*)$.)

But now we have $\nabla f(\vec{x}^*)$ and $\nabla h(\vec{x}^*)$ orthogonal to the same vector – when the set S has co-dimensions *one* (e.g. a surface in three dimensions) this means $\nabla f(\vec{x}^*)$ has to be a multiple of $\nabla h(\vec{x}^*)$; if the co-dimension of S is greater than one (e.g. a curve in three dimensions) then this conclusion cannot be made.

This is **Lagrange's Theorem** for $m = 1$: If $\vec{x}^*$ is a minimizer then there exists $\lambda \in \mathbb{R}$ such that

$$\nabla f(\vec{x}^*) + \lambda \nabla h(\vec{x}^*) = \vec{0}.$$

- The constant λ is called the Lagrange multiplier. Its existence is a *necessary* condition, but not a sufficient one.

EXAMPLE 2.3. Minimize $f(x, y) = x - y$ subject to $x^2 + y^2 - 1 = 0$. We need

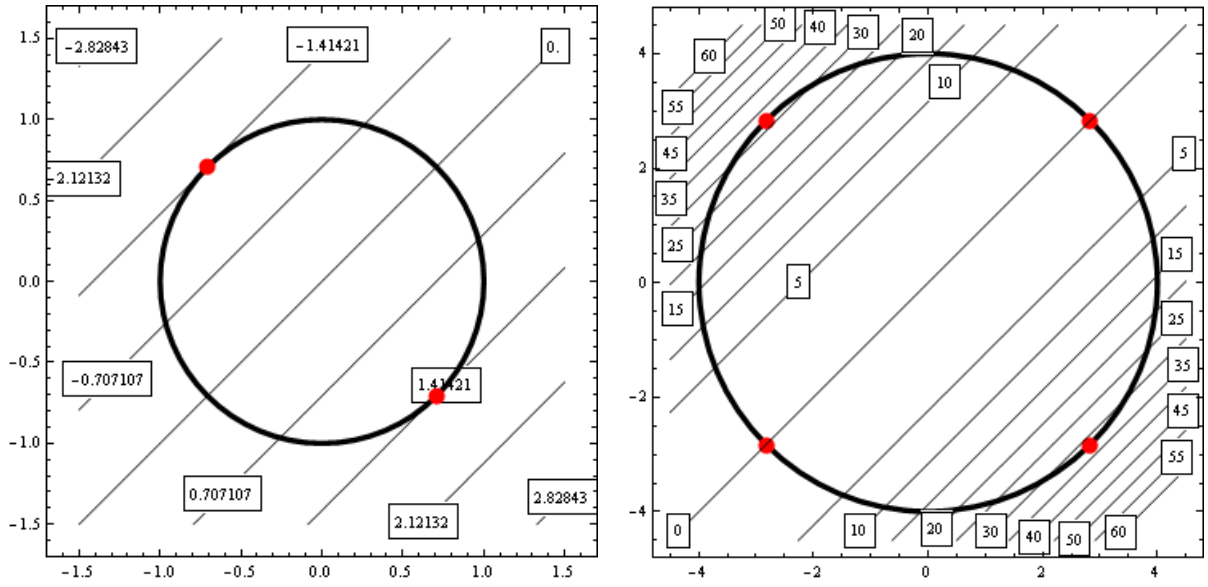
$$\begin{bmatrix} 1 \\ -1 \end{bmatrix} + \lambda \begin{bmatrix} 2x \\ 2y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \Rightarrow \quad 2\lambda(x + y) = 0.$$

If $\lambda = 0$ we get a contradiction; thus $x = -y$ and then $2x^2 = 1$, so $x = \pm 1/\sqrt{2}$ and the candidates are $[1/\sqrt{2}, -1/\sqrt{2}]^T$ and $[-1/\sqrt{2}, 1/\sqrt{2}]^T$. Evaluating, $f(1/\sqrt{2}, -1/\sqrt{2}) = 2/\sqrt{2}$ and $f(-1/\sqrt{2}, 1/\sqrt{2}) = -2/\sqrt{2}$. See Figure 2.3. □

EXAMPLE 2.4. Min/Maximize $f(x, y) = x^2 - 2xy + y^2$ subject to $x^2 + y^2 = 16$. One finds: $f(-2\sqrt{2}, -2\sqrt{2}) = 0$, $\lambda = 0$; $f(2\sqrt{2}, 2\sqrt{2}) = 0$, $\lambda = 0$; $f(-2\sqrt{2}, 2\sqrt{2}) = 32$, $\lambda = -2$; $f(2\sqrt{2}, -2\sqrt{2}) = 32$, $\lambda = -2$. Understand why we got $\lambda = 0$ in the two first cases by looking at Figure 2.4. □

The meaning of λ when $m = 1$. As we will now see, the multiplier λ measures the instantaneous rate of change of the critical values of the objective function with respect to changes in the *constraint level*. So, to see this, we re-cast the formulation slightly by writing the constraint as $h(\vec{x}) = c$, so $c = 0$ is the constraint we have just considered above. Let $\vec{x}^* = \vec{x}^*(0)$ be the minimizer of the constrained problem (if there is more than one, just pick one), and let λ be the corresponding Lagrange multiplier. Now consider the *family* of constrained problems as we let c vary (for values close to zero) and let $\vec{x}^*(c)$ be the corresponding minimizers; we now view $\vec{x}^*$ as a *function* of c, that is, $\vec{x}^* : \mathbb{R} \to \mathbb{R}^n$. Let $M(c) = f(\vec{x}^*(c))$ be the actual minimum achieved there, the value of the objective function at the optimal point. We ask the question: how is M changing as a function of c, near $c = 0$; i.e. by how much (approximately) will the minimum increase or decrease as a result of increasing/decreasing the constraint h? To answer this question we compute the derivative of M with respect to c, at $c = 0$:

$$\frac{dM}{dc}(0) = \underbrace{Df(\vec{x}^*(0))}_{1\times n}\underbrace{D\vec{x}^*(0)}_{n\times 1} = -\lambda \underbrace{Dh(\vec{x}^*(0))}_{1\times n}\underbrace{D\vec{x}^*(0)}_{n\times 1}.$$



(a) Example 2.3. (b) Example 2.4.

Figure 2.2: Examples of single equality constrained optimization problems. Shown are the constraint, level curves of the objective function, and the optimal points.

One must understand, in the above, that $Df(\vec{x}^*(0))$ is the differential with respect to the variable $\vec{x}$ (at $\vec{x}^*(0)$) and $D\vec{x}^*(0)$ is the differential with respect to the variable c (at $c = 0$). Now for every c, $h(\vec{x}^*(c)) = c$, so differentiating this, at $c = 0$, we obtain $Dh(\vec{x}^*(0))D\vec{x}^*(0) = \dfrac{d}{dc}c = 1$, so $\dfrac{d}{dc}M(0) = -\lambda$.

You should return to the previous examples and Figures 2.2 to understand this result in the context of those examples. In particular, notice that for the points where $\lambda = 0$ in Example 2.4, changes in the constraint will, indeed, not change the minimal value of f.

2.1.3 The Lagrange Multiplier Theorem

We return to the original problem, as in Section 2.1.1.

THEOREM 2.5. *Let $\vec{x}^*$ be a local minimizer of $f : \mathbb{R}^n \to \mathbb{R}$ subject to $h(\vec{x}) = \vec{0}$, $h : \mathbb{R}^n \to \mathbb{R}^m$, $m \leq n$. Assume that the rank of $Dh(\vec{x}^*)$ is m, maximal. Then there exists a (vector) multiplier $\vec{\lambda} \in \mathbb{R}^m$ such that*

$$Df(\vec{x}^*) + \vec{\lambda}^T Dh(\vec{x}^*) = \vec{0}^T. \tag{2.1}$$

Remarks: (1.) Note that what (2.1) says is that the gradient vector of f (at $\vec{x}^*$) is *in the span of* the gradient vectors of the constraint functions h_j (at $\vec{x}^*$). This is most easily seen by taking the transpose of (2.1).

(2.) Why might we see this as reasonable? If we consider two (linear) constraints intersecting in a line, then the span of the gradients of (assumed independent) h_j is a plane perpendicular to the line of intersection. If ∇f is *not* in this span, then there is a *component* of ∇f which is along the line of intersection: we could move along this line, *still satisfying the constraints* and *decrease* f. Thus we could not be at a minimizing point.

Tangent Spaces:

Key to the proof/understanding of Lagrange's theorem in 2D was the fact that we took a curve in the constraint set and differentiated it at $\vec{x}^*$. Now the constraint set is m-dimensional; we need the notion of a **tangent space** (we discussed the tangent *line* when the constraint set was one-dimensional).

DEFINITION 2.6. Given the constraint set $S = \{\vec{x} \ : \ h(\vec{x}) = \vec{0}\}$, $h : \mathbb{R}^n \to \mathbb{R}^m$, $m \leq n$, we define the *tangent space* $T_{\vec{x}^*}S$ to S at $\vec{x}^*$ to be the set of derivatives of curves in S through $\vec{x}^*$ at $\vec{x}^*$. More precisely, a vector $\vec{v}$ is a tangent vector to S at $\vec{x}^*$ if, and only if, there exists a curve $\vec{x}(t) : (-\varepsilon, \varepsilon) \to S$ with $\vec{x}(0) = \vec{x}^*$ such that $\vec{v} = D\vec{x}(0) = \vec{x}\,'(0)$.

It is not difficult to show that $T_{\vec{x}^*}S$ is a *vector space*. You should view it as the *(affine) tangent plane* to S at $\vec{x}^*$ *translated* to the origin.

EXAMPLE 2.7. Consider the unit sphere $\mathbb{S}^2$, the set $\mathbb{S}^2 = \{h(\vec{x}) = x^2 + y^2 + z^2 - 1 = 0\}$. If $\vec{x}^* \in \mathbb{S}^2$, then $T_{\vec{x}^*}\mathbb{S}^2$ is the plane perpendicular to $\vec{x}^*$ (through the origin).

The relationship between tangent spaces and the differential of h:

Necessary notions: the nullspace of a matrix (B.4.5); the Jacobian, or differential (B.7.2).

When the surface for which we are trying to describe the tangent space is given as the level set of a function (as it is here, $h(\vec{x}) = \vec{0}$), then it holds that $T_{\vec{x}^*}S \subset \mathcal{N}(Dh(\vec{x}^*))$. This can be understood as the set of vectors which are orthogonal to each of the rows of $Dh(\vec{x}^*)$, i.e. to each of the gradients $\nabla h_j(\vec{x}^*)$.

This is easy to see as follows: let $\vec{v} \in T_{\vec{x}^*}S$, so there exists $\vec{x}(t)$, $-\varepsilon < t < \varepsilon$, a curve in S such that $\vec{x}(0) = \vec{x}^*$ and $D\vec{x}(0) = \vec{x}\,'(0) = \vec{v}$. Then differentiating $h(\vec{x}(t)) = \vec{0}$ at $t = 0$ we obtain

$$Dh(\vec{x}(0))D\vec{x}(0) = Dh(\vec{x}(0))\vec{x}\,'(0) = \vec{0}$$

which is precisely the statement $\vec{v} = \vec{x}\,'(0) \in \mathcal{N}(Dh(\vec{x}^*))$.

Important Remark: In fact, if $Dh(\vec{x}^*)$ is full rank, then

$$T_{\vec{x}^*}S = \mathcal{N}(Dh(\vec{x}^*)), \tag{2.2}$$

and we can use $\mathcal{N}(Dh(\vec{x}^*))$ as the *definition* of the tangent space. (Note this is stronger than the *inclusion* shown above.) We will only be considering tangent spaces at points where the rank is maximal and so we will have the equivalent descriptions of the tangent space.

EXAMPLE 2.8. If $Dh(\vec{x}^*)$ is not full rank, then $\mathcal{N}(Dh(\vec{x}^*))$ can be larger than the tangent space to S. For example, let $\vec{x}^* = \vec{0} \in \mathbb{R}^3$, $h_1(x,y,z) = z$, and $h_2(x,y,z) = x^3 + z$. Then the gradients at $\vec{x}^*$ are both $[0,0,1]^T$. With $h = [h_1, h_2]^T$, $Dh(\vec{x}^*) = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}$ (which is clearly not of full rank). The constraint set S is the set $\{h_1 = 0,\ h_2 = 0\}$ which is the y-axis; the tangent space to S at $\vec{x}^*$ is also the y-axis. But $\mathcal{N}(Dh(\vec{x}^*))$ is the whole xy-plane. □

We will need:

EXERCISE 2.9. If A is a matrix, prove that $\mathcal{R}(A^T) = \mathcal{N}(A)^\perp$ where, if V is a subspace, $V^\perp = \{\vec{w} \ :\ \vec{w} \text{ is perpendicular to } \vec{v} \text{ for all } \vec{v} \in V\}$.

Proof of The Lagrange Multiplier Theorem. Now we are ready to prove (2.1): This is equivalent to $\nabla f(\vec{x}^*) = -Dh(\vec{x}^*)^T\vec{\lambda}$, which says that $\nabla f(\vec{x}^*)$ is in the range of $Dh(\vec{x}^*)^T : \mathbb{R}^m \to \mathbb{R}^n$. But by Exercise 2.9, $\mathcal{R}(Dh(\vec{x}^*)^T) = \mathcal{N}(Dh(\vec{x}^*))^\perp$ (B.4.6) and we just saw $\mathcal{N}(Dh(\vec{x}^*))^\perp = (T_{\vec{x}^*}S)^\perp$. Thus we must show $\nabla f(\vec{x}^*) \in (T_{\vec{x}^*}S)^\perp$. For this, let $\vec{v} \in T_{\vec{x}^*}S$ and $\vec{x} : (-\varepsilon, \varepsilon) \to S$ be such that $\vec{x}(0) = \vec{x}^*,\ \vec{x}\,'(0) = \vec{v}$. Now consider $f(\vec{x}(t))$. Since $\vec{x}^*$ is a minimizer,

$$0 = \frac{d}{dt} f(\vec{x}(t))\big|_{t=0} = Df(\vec{x}^*)D\vec{x}(0) = Df(\vec{x}^*)\vec{v} = \nabla f(\vec{x}^*)^T\vec{v}$$

and indeed $\nabla f(\vec{x}^*)$ is perpendicular to $\vec{v}$. □

Once again, the Lagrange Multiplier Theorem gives a *necessary* condition; it need not be sufficient.

EXAMPLE 2.10. We can use the example above to see why the full-rank condition is in the hypothesis of the theorem. Let the objective function be $f(x,y,z) = x + z$, and the constraints be $h_1 = z$, $h_2 = x^3 + z$. Then the feasible set S is the y-axis; f is constant on this set, so any point is a minimizer, in particular $\vec{0}$. But, at $\vec{0}$ it is not true that there exists $\vec{\lambda} \in \mathbb{R}^2$ such that $Df(\vec{0}) + \vec{\lambda}^T Dh(\vec{0}) = \vec{0}^T$. We see this since $Df(\vec{0}) = [1,0,1]$ while $\vec{\lambda}^T Dh(\vec{0}) = [0,0,\lambda_1+\lambda_2]$. □

EXAMPLE 2.11. Minimize $f(x_1,x_2) = x_1^3 + x_2^3$ subject to $x_1^2 + x_2^2 = 4$. Lagrange's theorem guarantees that at a minimum there exists λ such that

$$\begin{aligned} 3x_1^2 + 2\lambda x_1 = 0 \quad &\Rightarrow \quad x_1(3x_1 + 2\lambda) = 0 \\ 3x_2^2 + 2\lambda x_2 = 0 \quad &\Rightarrow \quad x_2(3x_2 + 2\lambda) = 0 \\ x_1^2 + x_2^2 = 4. & \end{aligned}$$

We consider the four cases implied by the first two equations:
$\boxed{x_1 = 0}$:

- $\boxed{x_2 = 0}$: this contradicts the third equation;

- $\boxed{x_2 = -2\lambda/3}$: the third equation gives $\lambda = \pm 3$ and then $x_2 = \mp 2$. We have

$$[0, 2]^T,\ \lambda = -3,\ \text{and } [0, -2]^T,\ \lambda = 3.$$

$\boxed{x_1 = -2\lambda/3}$:

- $\boxed{x_2 = 0}$: the third equation gives $\lambda = \pm 3$. We have

$$[2, 0]^T,\ \lambda = -3,\ \text{and } [-2, 0]^T,\ \lambda = 3.$$

- $\boxed{x_2 = -2\lambda/3}$: the third equation gives $\lambda = \pm 3/\sqrt{2}$ and $x_1 = x_2 = \mp\sqrt{2}$. We have

$$[\sqrt{2}, \sqrt{2}]^T,\ \lambda = -3/\sqrt{2},\ \text{and } [-\sqrt{2}, -\sqrt{2}]^T,\ \lambda = 3/\sqrt{2}.$$

Evaluating the objective function at these locations,

$$\begin{aligned} &f(0, 2) = f(2, 0) = 8, \text{ (maxima)},\\ &f(\sqrt{2}, \sqrt{2}) = 4\sqrt{2},\ f(-\sqrt{2}, -\sqrt{2}) = -4\sqrt{2}, \text{ and},\\ &f(0, -2) = f(-2, 0) = -8, \text{ (minima)}. \end{aligned}$$

□

EXERCISE 2.12. Extend the above example to $f(x_1, x_2, x_3) = x_1^3 + x_2^3 + x_3^3$ subject to $x_1^2 + x_2^2 + x_3^2 = 4$.

EXAMPLE 2.13. (An objective function of three variables, with two constraints. This is the original reference Example 2.1.) Optimize $f(x, y, z) = 3xz + 4yz$ subject to $y^2 + z^2 = 4$ and $xz = 3$. Figure 2.3 shows the constraint sets, and the optimal points found below. We need

$$\begin{bmatrix} 3z & 4z & 3x + 4y \end{bmatrix} + \begin{bmatrix} \lambda_1 & \lambda_2 \end{bmatrix} \begin{bmatrix} 0 & 2y & 2z \\ z & 0 & x \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}.$$

So, $3z + \lambda_2 z = 0 \Rightarrow z = 0$ or $\lambda_2 = -3$ ($z = 0$ gives a contradiction). Then $4z + 2y\lambda_1 = 0$ and $4y + 2\lambda_1 z = 0$ which together give $y = \pm z$. Thus we obtain

$$\underset{f=17}{\left(\frac{3}{\sqrt{2}}, \sqrt{2}, \sqrt{2}\right)}, \quad \underset{f=17}{\left(-\frac{3}{\sqrt{2}}, -\sqrt{2}, -\sqrt{2}\right)}, \quad \underset{f=1}{\left(-\frac{3}{\sqrt{2}}, \sqrt{2}, -\sqrt{2}\right)}, \quad \underset{f=1}{\left(\frac{3}{\sqrt{2}}, -\sqrt{2}, \sqrt{2}\right)}$$

Note: at this point we might be tempted to conclude that the minimum is 1 and the maximum is 17 (each achieved twice at the given locations). If the constraint set is *compact* (closed and bounded in $\mathbb{R}^n$) then this would be a correct conclusion. Here, however, the constraint set is unbounded so we have to consider also what happens to f as we move to the unbounded part. That is, what happens as $x \to \pm\infty$? We can answer this by noticing that $xz = 3$, so in fact

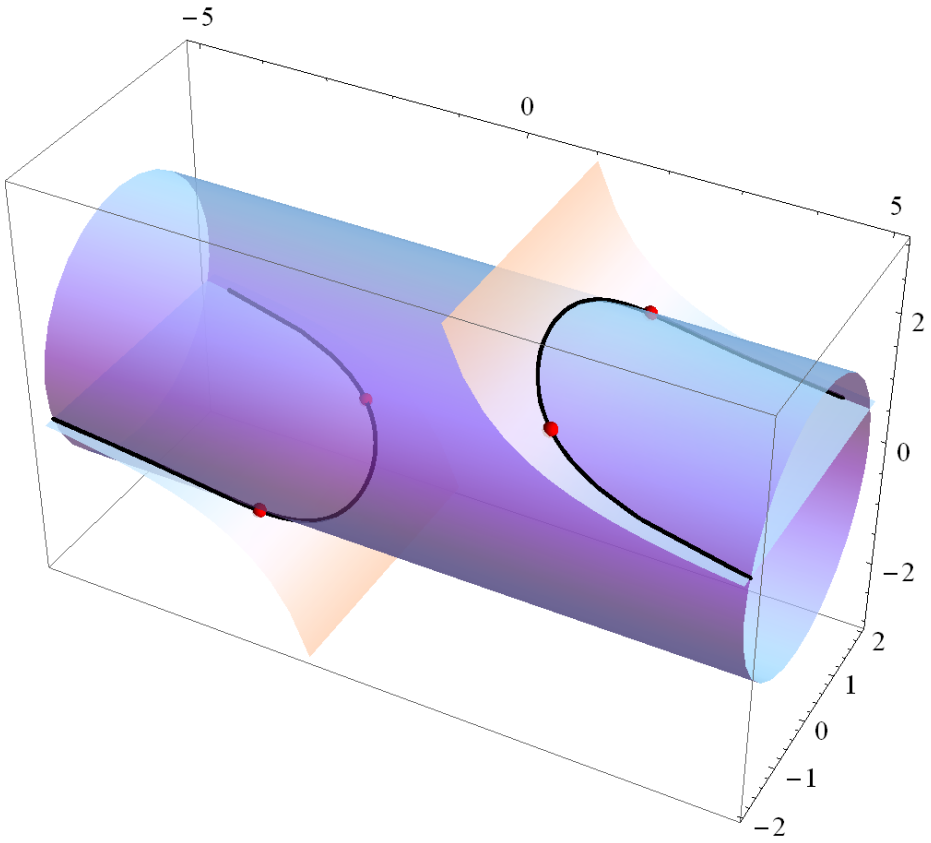


Figure 2.3: (Example 2.13) the constraints and optimal points.

$f(x, y, z) = 9 + 4yz$; as $x \to \pm\infty$, $z \to 0$, so $f \to 9$ which is *between* the values we found. Thus it turns out 1 and 17 are the extrema.

In Figure 2.4, we zoom in on one of the optimal points and show: the constraint surfaces and their sphere of intersection; the optimal point; and a patch of the *level set* of the objective function through this points. What is important to see is that the constraint set (the curve of intersection of the constraints) is *tangent* to the level set of the function. That is, moving infinitessimally along the constraint from an optimal point will not (instantaneously) change the value of the objective function. Thus, it cannot be increased or decreased by moving along the constraint. □

EXAMPLE 2.14. We consider the example of Exercise 2.12 but impose the additional constraint $x_1 + x_2 + x_3 = 1$. What is interesting about this example is that we can use the *existence* of λ_1, λ_2 to derive the solution. The condition from Lagrange's theorem requires

$$\begin{aligned} 3x_1^2 + 2\lambda_1 x_1 + \lambda_2 &= 0 \\ 3x_2^2 + 2\lambda_1 x_2 + \lambda_2 &= 0 \\ 3x_3^2 + 2\lambda_1 x_3 + \lambda_2 &= 0. \end{aligned}$$

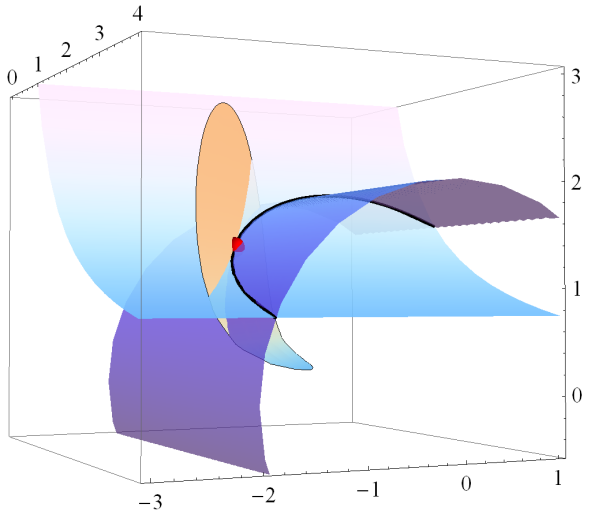


Figure 2.4: (Example 2.13) constraints and level sets of the objective function.

Mere existence of λ_1, λ_2 means that the matrix

$$\begin{bmatrix} x_1^2 & x_1 & 1 \\ x_2^2 & x_2 & 1 \\ x_3^2 & x_3 & 1 \end{bmatrix}$$

must have determinant zero because the equation

$$\begin{bmatrix} x_1^2 & x_1 & 1 \\ x_2^2 & x_2 & 1 \\ x_3^2 & x_3 & 1 \end{bmatrix} \begin{bmatrix} 3 \\ 2\lambda_1 \\ \lambda_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

must have a solution, and it clearly cannot be $[0,0,0]^T$. This matrix has a special form – it is known as a Vandermonde matrix, and its determinant is known to be $(x_1-x_2)(x_1-x_3)(x_2-x_3)$. Now, for example, if $x_1 = x_2$ then the constraints become $x_3 = 1-2x_1$ and $2x_1^2+(1-2x_1)^2 = 4$, or $6x_1^2 - 4x_1 - 3 = 0$ which has solutions $x_1 = (2 \pm \sqrt{22})/6$. The complete set of solutions turns out to be

$$\left[\frac{1}{6}(2-\sqrt{22}), \frac{1}{6}(2-\sqrt{22}), \frac{1}{3}(1+\sqrt{22})\right]^T \quad \left[\frac{1}{6}(2-\sqrt{22}), \frac{1}{3}(1+\sqrt{22}), \frac{1}{6}(2-\sqrt{22})\right]^T$$
$$\left[\frac{1}{6}(2+\sqrt{22}), \frac{1}{6}(2+\sqrt{22}), \frac{1}{3}(1-\sqrt{22})\right]^T \quad \left[\frac{1}{6}(2+\sqrt{22}), \frac{1}{3}(1-\sqrt{22}), \frac{1}{6}(2+\sqrt{22})\right]^T.$$

The corresponding $\vec{\lambda}$'s are (numerically) $[-2.17, -2.55]^T$, $[-2.15, -2.55]^T$, $[0.17, -0.44]^T$, and $[0.17, -4.11]^T$. The values of f at the four points are $6.64,\ 6.64,\ 0.91,\ 0.91$. □

The meaning of $\vec{\lambda}$. Once again, consider the constraints varying, $h(\vec{x}) = \vec{c} \approx \vec{0} \in \mathbb{R}^m$. Let $\vec{x}^*(\vec{0})$ be a minimizer of the constrained problem when $\vec{c} = \vec{0}$ and let the corresponding minimizers (near $\vec{x}^*(\vec{0})$) for general $\vec{c}$ be $\vec{x}^*(\vec{c}) \in \mathbb{R}^n$. Let $M : \mathbb{R}^m \to \mathbb{R}$ be given by $M(\vec{c}) =$

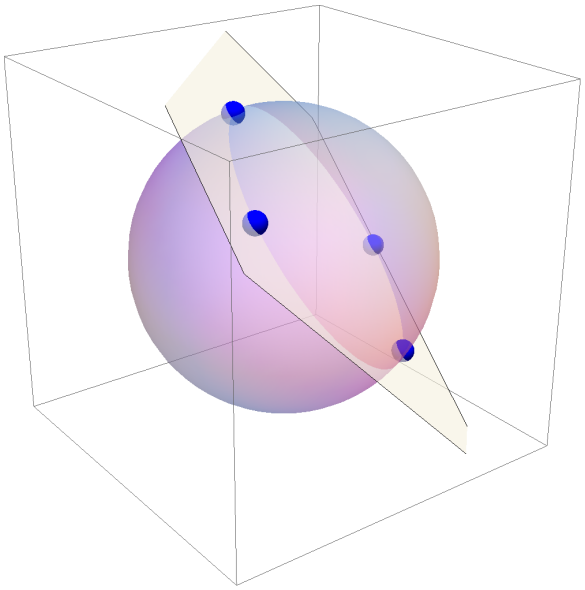


Figure 2.5: (Example 2.14) the constraints and optimal points.

$f(\vec{x}^*(\vec{c}))$, the values of the objective function at the minimizers $\vec{x}^*(\vec{c})$. Differentiating with respect to $\vec{c}$, at $\vec{c} = \vec{0}$,

$$\underbrace{DM(\vec{0})}_{1\times m} = \underbrace{Df(\vec{x}^*(\vec{0}))}_{1\times n}\underbrace{D\vec{x}^*(\vec{0})}_{n\times m} = -\underbrace{\vec{\lambda}^T}_{1\times m}\underbrace{Dh(\vec{x}^*(\vec{0}))}_{m\times n}\underbrace{D\vec{x}^*(\vec{0})}_{n\times m}.$$

This time, $h(\vec{x}^*(\vec{c})) = \vec{c}$, so (actually for all $\vec{c}$)

$$\underbrace{Dh(\vec{x}^*(\vec{c}))}_{m\times n}\underbrace{D\vec{x}^*(\vec{c})}_{n\times m} = I_{m\times m}.$$

Thus

$$\underbrace{DM(\vec{0})}_{1\times m} = -\underbrace{\vec{\lambda}^T}_{1\times m}.$$

That is, the *gradient* of M at $\vec{0}$ is $-\vec{\lambda}$. Now gradients always point in the directions of maximum increase, so to increase M the most rapidly we should change the constraint from $h = \vec{0}$ to $h = \vec{c}$, where $\vec{c}$ is (small and) in the direction of $-\vec{\lambda}$. Notice also that if $\vec{c}$ were a direction *perpendicular* to $\vec{\lambda}$, then the minimum value (at the new minimizer) would not change (instantaneously).

What this says is the rate of change of M in the direction of $\vec{c}/\|\vec{c}\|$ is $\nabla M(\vec{0}) \cdot \vec{c}/\|\vec{c}\| = -\vec{\lambda} \cdot \vec{c}/\|\vec{c}\|$. That is,

$$\frac{\Delta M}{\|\text{change in } \vec{c}\|} \approx -\vec{\lambda} \cdot \frac{\vec{c}}{\|\vec{c}\|} \quad \Rightarrow \quad \Delta M \approx -\vec{\lambda} \cdot \vec{c}.$$

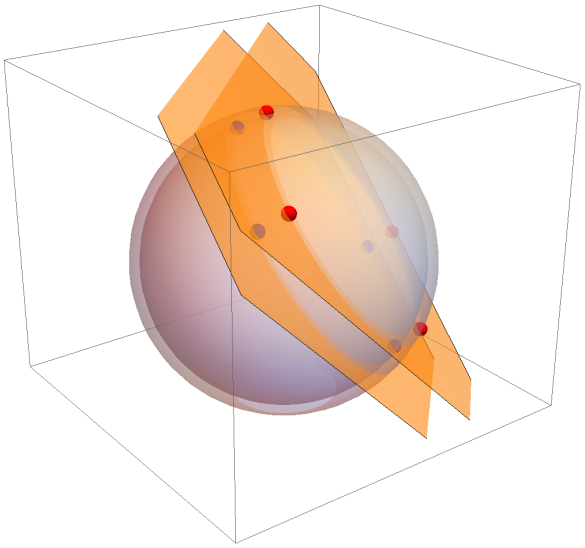


Figure 2.6: The constraint surfaces and optimal points for two sets of constraints. The blue points are optimal for $\vec{c} = [0, 0]^T$; the red points are optimal for $\vec{c} = [0.6, 0.8]^T$.

EXAMPLE 2.15. Consider Example 2.14 presented above. When we change the constraints from

$$\begin{aligned} x_1^2 + x_2^2 + x_3^2 - 4 &= 0 \\ x_1 + x_2 + x_3 - 1 &= 0 \end{aligned} \qquad \text{to} \qquad \begin{aligned} x_1^2 + x_2^2 + x_3^2 - 4 &= c_1 \\ x_1 + x_2 + x_3 - 1 &= c_2 \end{aligned}$$

(or from $h(\vec{x}) = \vec{0}$ to $h(\vec{x}) = \vec{c}$), we obtain a sphere of a slightly different radius, and a plane which has been shifted slightly. The optimal points also shift; in Figure 2.6, we have taken $\vec{c} = [c_1, c_2]^T = [0.6, 0.8]^T$ (much larger than realistic for instantaneous rates of change, but better for illustration). The blue (lower) points are the optimal points for the original constraints $\vec{c} = [0, 0]^T$, the red for the new constraints $\vec{c} = [0.6, 0.8]^T$. If we pick one, say the "closest" blue point $\vec{x} = \left[\frac{1}{6}\left(2 + \sqrt{22}\right), \frac{1}{3}\left(1 - \sqrt{22}\right), \frac{1}{6}\left(2 + \sqrt{22}\right)\right]^T$ where $f(\vec{x}) = 0.91$ achieves a minimum, then we can estimate by how much f changes in going to the corresponding red point (here M represents the value of f at the minimum):

$$\Delta M \approx -\lambda \cdot \vec{c} = -[0.17, -4.11] \begin{bmatrix} 0.6 \\ 0.8 \end{bmatrix} = 3.186.$$

That is, if one were to change the constraints as described, our (linear approximation) estimate of the new minimum would be $0.91 + 3.186 = 4.096$ (the old minimum plus the estimate of the change). Given how big $\vec{c}$ is, we might not be too confident in this estimate. It actually turns out that the new minimum is 4.29, so our estimate isn't too bad. □

2.2 Constrained Optimization with inequality constraints

2.2.1 Formulation

Formulation of the problem:

> Minimize $f(\vec{x}) : \mathbb{R}^n \to \mathbb{R}$ subject to $h(\vec{x}) = \vec{0}$, $h : \mathbb{R}^n \to \mathbb{R}^m$ and $g(\vec{x}) \leq \vec{0}$, $g : \mathbb{R}^n \to \mathbb{R}^p$.

Once again, $g(\vec{x}) \leq \vec{0}$ is in fact p inequality constraints $g_j(\vec{x}) \leq 0$, $1 \leq j \leq p$.

Terminology: an inequality constraint $g_j \leq 0$ is said to be *active* (or *binding*) at $\vec{x}$ if in fact $g_j(\vec{x}) = 0$; if $g_j(\vec{x}) < 0$ it is *inactive* (*non-binding*).

If $\vec{x}^*$ satisfies $h(\vec{x}^*) = \vec{0},\ g(\vec{x}^*) \leq \vec{0}$ then we define $J(\vec{x}^*)$ to be the set of indices j for which g_j is active at $\vec{x}^*$:

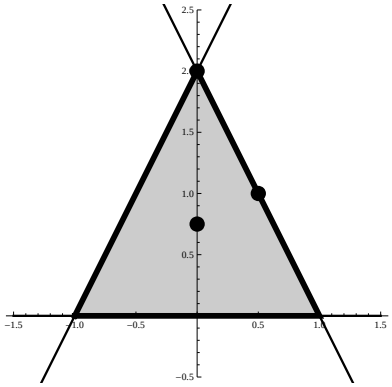
$$J(\vec{x}^*) = \{j \ : \ g_j(\vec{x}^*) = 0\}.$$

EXAMPLE 2.16. The triangle has $g_1(x, y) = y + 2x - 2$, $g_2(x, y) = y - 2x - 2$, $g_3(x, y) = -y$.

$\vec{x}^* = [0, 0.75]^T$: No constraints are active, $J(\vec{x}^*) = \{\ \}$;

$\vec{x}^* = [0, 2]^T$: Only g_1 and g_2 are active, $J(\vec{x}^*) = \{1, 2\}$;

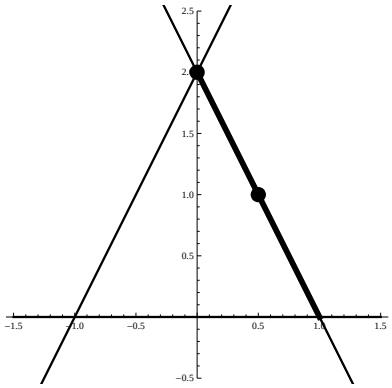
$\vec{x}^* = [0.5, 1]^T$: Only g_1 is active, $J(\vec{x}^*) = \{1\}$.



EXAMPLE 2.17. Here, $h(x, y) = y + 2x - 2$, $g_1(x, y) = y - 2x - 2$, and $g_2(x, y) = -y$.

$\vec{x}^* = [0, 2]^T$: Only g_1 is active, $J(\vec{x}^*) = \{1\}$;

$\vec{x}^* = [0.5, 1]^T$: None of g_1, g_2 are active, $J(\vec{x}^*) = \{\ \}$.



2.2.2 The Karush-Kuhn-Tucker Theorem

When there are inequality constraints, the analog to Lagrange's theorem is the Karush-Kuhn-Tucker Theorem (KKT). Recall that we use $\vec{\mu} \geq \vec{\nu}$ to mean that for *each component* $\mu_j \geq \nu_j$.

THEOREM 2.18. *Let $\vec{x}^*$ be a point where $\{\nabla h_i(\vec{x}^*), \nabla g_j(\vec{x}^*) \ : \ 1 \leq i \leq m, \ j \in J(\vec{x}^*)\}$ is a linearly independent set, and such that $\vec{x}^*$ is a local minimizer of f subject to $h(\vec{x}) = \vec{0}$, $g(\vec{x}) \leq \vec{0}$. Then there exist $\vec{\lambda} \in \mathbb{R}^m$ and $\vec{\mu} \in \mathbb{R}^p$ such that*

1. $\vec{\mu} \geq \vec{0}$,
2. $Df(\vec{x}^*) + \vec{\lambda}^T Dh(\vec{x}^*) + \vec{\mu}^T Dg(\vec{x}^*) = \vec{0}^T$, *and*
3. $\vec{\mu}^T g(\vec{x}^*) = 0$.

Maximization: In seeking the maximum of f subject to the same constraints, we will see that we consider the same equations but with 1. replaced by $\vec{\mu} \leq \vec{0}$.

Important Observation: KKT 1. ($\vec{\mu} \geq 0$) and KKT 3. ($\vec{\mu}^T g(\vec{x}^*) = 0$) can be written in another, more useful, way:

- Since we require $g(\vec{x}^*) \leq 0$, and $\vec{\mu} \geq 0$, it certainly must hold that $\mu_j g_j(\vec{x}^*) \leq 0$ for each j;
- if, for some j, $\mu_j > 0$ then it must hold that the corresponding $g_j(\vec{x}^*) = 0$ (if not, then $\mu_j g_j(\vec{x}^*) < 0$ and it is then impossible for KKT 3. to hold);
- similarly, if, for some j, $g_j(\vec{x}^*) < 0$ then it must hold that the corresponding $\mu_j = 0$ (for the same reason).

Thus, KKT 3. can be replaced by the p conditions

$$\mu_1 g_1(\vec{x}^*) = 0, \quad \mu_2 g_2(\vec{x}^*) = 0, \quad \ldots, \quad \mu_p g_p(\vec{x}^*) = 0.$$

A look at $\vec{\mu}$: as in the observation above, if for some j, $g_j(\vec{x}^*) < 0$, then $\mu_j = 0$. That is, if the j^{th} inequality constraint g_j is *inactive* then the corresponding $\mu_j = 0$; it is only the *active* constraints that may give rise to non-zero components of $\vec{\mu}$; again, it is only the $j \in J(\vec{x}^*)$ for which μ_j may be non-zero. Note, however, that $\mu_j = 0$ does not imply that the constraint g_j is necessarily inactive.

In fact, as becomes apparent in the proof of KKT, when a constraint is not active, it plays no role, and any constraint that is active is an equality constraint as in Lagrange's theorem. The technical condition that $\{\nabla h_i(\vec{x}^*), \nabla g_j(\vec{x}^*) \ : \ 1 \leq i \leq m, \ j \in J(\vec{x}^*)\}$ be linearly independent

is simply the full-rank condition of Lagrange's theorem, applied to the set of *active* constraints. In the following two-dimensional example, we can see that the points found correspond to places where the constraints are tangent to the level curves of the objective function.

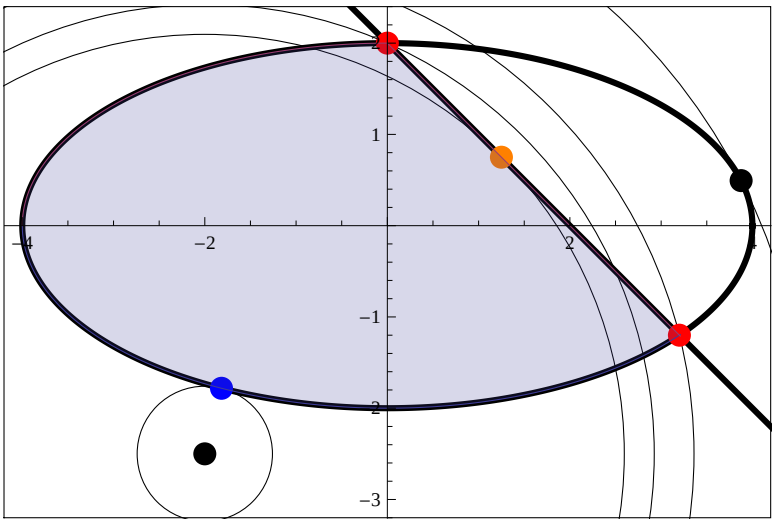
EXAMPLE 2.19. We present an example before proceeding to the proof of the KKT theorem. Consider optimizing $f(x,y) = (x+2)^2+(y+2.5)^2$ subject to $x^2+4y^2 \leq 16$ and $x+y \leq 2$. The objective function is simply a parabolic bowl whose minimum is shifted to $x = -2$, $y = -2.5$. The feasible set is shown in the picture below. Ignoring the sign of μ_j and feasibility for the moment, we seek solutions to

$$\begin{aligned} 2(x+2) + 2x\mu_1 + \mu_2 &= 0 \\ 2(y+2.5) + 8y\mu_1 + \mu_2 &= 0 \\ \mu_1(x^2+4y^2-16) &= 0 \\ \mu_2(x+y-2) &= 0. \end{aligned}$$

Without discussing how the solutions are found, one obtains:

$$\begin{aligned} &\vec{x} = [0,2]^T,\ \vec{\mu} = [-0.31,-4]^T && \vec{x} = [3.2,-1.2]^T,\ \vec{\mu} = [-0.49,-7.28]^T \\ &\vec{x} = [-1.81,-1.78]^T,\ \vec{\mu} = [0.1,0]^T && \vec{x} = [3.88,0.49]^T,\ \vec{\mu} = [-1.52,0]^T \\ &\vec{x} = [-2,-2.5]^T,\ \vec{\mu} = [0,0]^T && \vec{x} = [1.25,0.75]^T,\ \vec{\mu} = [0,-6.5]^T \end{aligned}$$

The points are shown in the figure below, together with some key contour curves of f:



- At $[0,2]^T$ and $[3.2,-1.2]^T$, $\vec{\mu} < \vec{0}$ – both constraints are active, and moving off either one of them into the feasible region *decreases* f. The negative sign of $\vec{\mu}$ discards these points for the minimization problem; they are (potentially) optimal only for the maximization problem.
- At $[-1.81,-1.78]^T$ we have $\mu = [0.1,0]^T$ – only the elliptical constraint is active. Moving off this constraint into the feasible region *increases* f. The fact that $\vec{\mu} \geq \vec{0}$ keeps this point as potentially optimal for minimization.

- At $[3.88, 0.49]^T$, $\vec{\mu} = [-1.52, 0]^T$ – again only the elliptical constraint is active but now moving off this constraint into the region allowable by this constraint *decreases* f. If this were a feasibly point, it would be (potentially) optimal for maximizing f. However, it is not feasible.

- At $[-2, -2.5]^T$ we are at a critical point of f. Looking at equation 2. of KKT, this clearly occurs if $\vec{\mu} = \vec{0}$, as in the case here. However, $[-2, -2.5]^T$ is not feasible.

- At $[1.25, 0.75]^T$, $\vec{\mu} = [0, -6.5]^T$ – only the linear constraint is active, and the sign of μ_2 indicates that it is (potentially) a point of maximum. We see that, indeed, moving into the feasible region from here will *decrease* f. It might appear that it is possible to move into the feasible region in such a way as to *increase* f, but it is important to remember that Lagrange's Theorem (and KKT) are based on *first derivatives* of f and the constraint functions; KKT cannot "see" the fact that the level curve of f curves away from the constraint at this point.

□

As we proceed to the proof of KKT, consider an example where h is absent, g_1 describes the unit disk centered at $(0, 0)$ and g_2 describes the unit disk centered at $(1, 0)$; let $\vec{x}^*$ be at $(1, 0)$ say. In Figure 2.7, the feasible set is the shaded region, together with its boundary, the bold arcs.

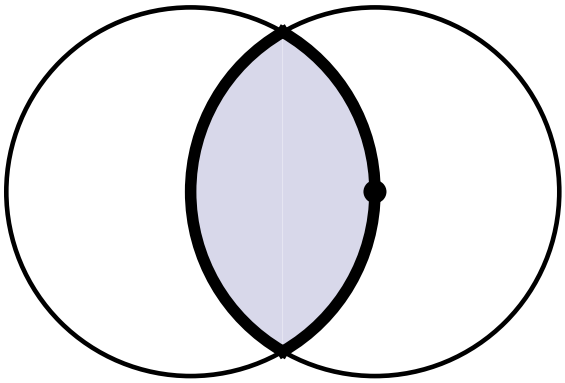


Figure 2.7: Illustrative example for the proof of KKT.

Proof of KKT, Step 1:

Let U be the feasible set; let

$$\widehat{U} = \{\vec{x} \ : \ h(\vec{x}) = \vec{0};\ g_j(\vec{x}) = 0,\ j \in J(\vec{x}^*);\ g_k(\vec{x}) < 0,\ k \notin J(\vec{x}^*)\}.$$

In words, $\widehat{U}$ is the part of the feasible set in which the *active* constraints are changed to be equality constraints. In the example, $\widehat{U}$ is the part of the *circle* centered at $(0, 0)$ which is inside

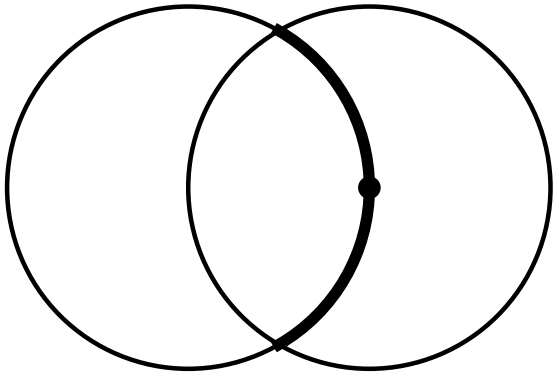


Figure 2.8: The active constraint is considered as an *equality* constraint.

the *disk* centered at $(1, 0)$. In Figure 2.8, it is the bold arc. Notice that for $\vec{x}^*$, g_1 is active, but g_2 is inactive. Certainly $\vec{x}^* \in \widehat{U}$. Since $g_k(\vec{x}) < 0$ is an "open" condition, there is $\varepsilon > 0$ such that $g_k(\vec{x}) < 0$ for all $x \in B_\varepsilon(\vec{x}^*)$, $k \notin J(\vec{x}^*)$ and such that $f(\vec{x}^*)$ is minimal on $\widehat{U} \cap B_\varepsilon(\vec{x}^*)$. Now set

$$\widetilde{U} = \{\vec{x} \,:\, h(\vec{x}) = \vec{0};\; g_j(\vec{x}) = 0,\; j \in J(\vec{x}^*)\} \cap B_\varepsilon(\vec{x}^*) = \widehat{U} \cap B_\varepsilon(\vec{x}^*) \subset \widehat{U}.$$

Certainly $\vec{x}^*$ is a local minimizer of f on $\widetilde{U}$.

In the example, the point is that not only is g_2 inactive at $\vec{x}^*$, but g_2 is inactive in a *neighborhood* of $\vec{x}^*$. $\widetilde{U}$ is shown in Figure 2.9 below – it is the intersection of the bold arc with a ball around $\vec{x}^*$ which is small enough to stay away from the inactive constraint. Within the bold arc, the

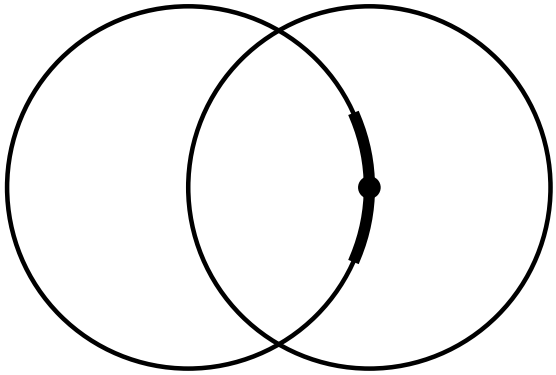


Figure 2.9: Only the active constraint(s) are relevant.

constraint g_2 can simply be forgotten!

It is instructive to think of the same example but in three dimensions. Replace g_1 by $g_1 = x_1^2 + x_2^2 + x_3^2$ and g_2 by $g_2 = (x_1 - 1)^2 + x_2^2 + x_3^2$, and let $\vec{x}^* = [1, 0, 0]^T$. Now the feasible set is the intersection of the two unit *balls*; the set $\widehat{U}$ is now the part of the unit *sphere* centered at $\vec{0}$ which intersects the unit ball centered at $[1, 0, 0]^T$, and $\widetilde{U}$ is a small curved disk within this surface, close to $\vec{x}^*$.

Proof of KKT, Step 2:

We have just proven that $\vec{x}^*$ is a local minimizer of the *equality constrained* problem where we consider only the active constraints. Thus we can apply the Lagrange multiplier theorem to conclude there exists $(\vec{\lambda}, \vec{\mu}) \in \mathbb{R}^m \times \mathbb{R}^p$ (we simply set $\mu_k = 0$ for $k \notin J(\vec{x}^*)$) such that

$$Df(\vec{x}^*) + \vec{\lambda}^T Dh(\vec{x}^*) + \vec{\mu}^T Dg(\vec{x}^*) = \vec{0}^T.$$

Note that we have $\vec{\mu}^T g(\vec{x}^*) = 0$ since for each j, either $g_j(\vec{x}^*) = 0$ when $j \in J(\vec{x}^*)$ or $\mu_j = 0$ when $j \notin J(\vec{x}^*)$.

Proof of KKT, Step 3:

We must show that $\mu_j \geq 0$ for $j \in J(\vec{x}^*)$. This is significantly more difficult and is where KKT differs from Lagrange's theorem. Th idea of the proof is: if there exists a $\mu_j < 0$ then we can show there must exist a direction in which we can move (instantaneously) from $\vec{x}^*$ which *remains feasible*, but for which f is (again, instantaneously) *decreasing*. This cannot be true is $\vec{x}^*$ is a local minimum.

To illustrate this we consider an example with three inequality constraints: g_1 gives the inside of the sphere of radius 1 centered at $[0, 0, 0]^T$, g_2 the same centered at $[1, 0, 0]^T$ and g_3 the same centered at $[1/2, 0, -1]^T$. The feasible set U is shown in Figure 2.10 below. The local minimum shown is at $\vec{x}^* \approx [0.99, 0, -0.13]^T$. The set $J(\vec{x}^*)$ is $J(\vec{x}^*) = \{1, 3\}$. We prove the claim by

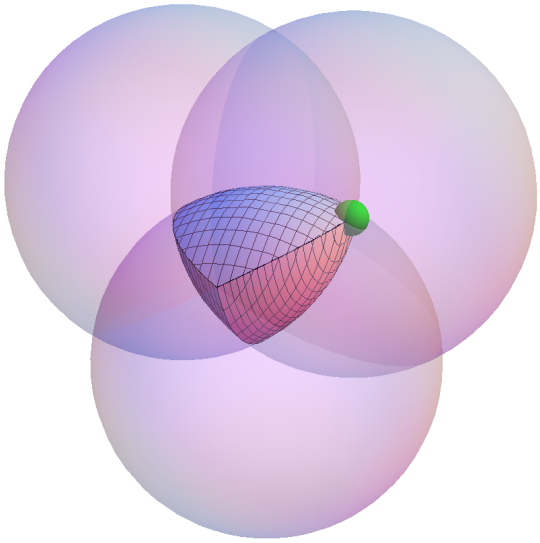


Figure 2.10: The feasible region satisfying three inequality constraints and an optimal point $\vec{x}^*$.

contradiction. Suppose (without loss of generality) that $\mu_1 < 0$. We define the surface

$$\widehat{S} = \{\vec{x} \ : \ h(\vec{x}) = \vec{0}, \ g_j(\vec{x}) = 0, \ j \in J(\vec{x}^*), \ j \neq 1\} \cap U.$$

This is the part of the feasible set containing $\vec{x}^*$ corresponding to the active constraints *other than* the one for which we assume the sign of μ is incorrect. In our example, it is part of the constraint $g_3 = 0$ and is shown in Figure 2.11. Now consider the corresponding tangent space

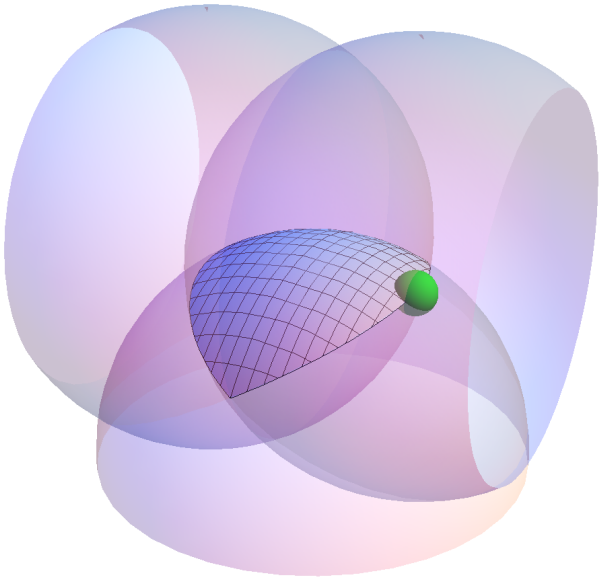


Figure 2.11: The feasible part of constraint $g_3 = 0$.

$T_{\vec{x}^*}\widehat{S}$, which, as we saw earlier, can be described as the null-space of appropriate derivatives:

$$T_{\vec{x}^*}\widehat{S} = \{\vec{v} \ : \ Dh(\vec{x}^*)\vec{v} = \vec{0}, \ Dg_j(\vec{x}^*)\vec{v} = 0, \ j \in J(\vec{x}^*), \ j \neq 1\}. \tag{2.3}$$

In our example, $T_{\vec{x}^*}\widehat{S}$ is the set of vectors perpendicular to ∇g_3.

Step 3.1. Claim: There must exist $\vec{v} \in T_{\vec{x}^*}\widehat{S}$ such that $Dg_1(\vec{x}^*)\vec{v} \neq 0$. This is equivalent to saying there is a direction in $T_{\vec{x}^*}\widehat{S}$ which is not tangent to the surface $g_1(\vec{x}^*) = 0$. In our example, we see this easily in Figure 2.13. To prove the claim, suppose otherwise, suppose that $Dg_1(\vec{x}^*)\vec{v} = 0$ for *all* $\vec{v} \in T_{\vec{x}^*}\widehat{S}$ then $\nabla g_1(\vec{x}^*) \in (T_{\vec{x}^*}\widehat{S})^{\perp}$. But this implies

$$\nabla g_1(\vec{x}^*) \in \text{span}\{\nabla h_i(\vec{x}^*), \ i = 1, \ldots, m, \ \nabla g_j(\vec{x}^*), \ j \in J(\vec{x}^*), \ j \neq 1\}$$

which contradicts the assumption that the full set of gradient vectors forms a linearly independent set. Thus there must exist such a $\vec{v}$ with $Dg_1(\vec{x}^*)\vec{v} \neq 0$; we can take $\vec{v}$ so that $Dg_1(\vec{x}^*)\vec{v} < 0$ (if necessary, simply take $-\vec{v}$).

Remark: This says that $\vec{v}$ is a direction in which g_1 decreases (from $\vec{x}^*$), and since $g_1(\vec{x}^*) = 0$, moving in this direction yields $g_1 \leq 0$, and so is in a direction which *remains* within the feasible set.

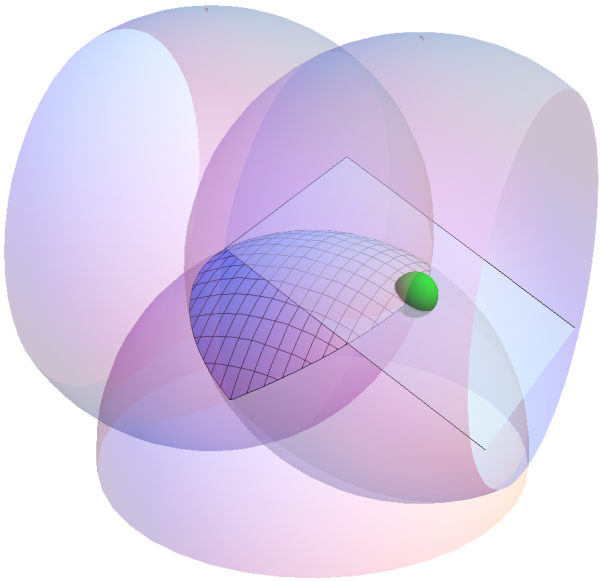


Figure 2.12: The tangent plane to $g_3 = 0$ at $\vec{x}^*$.

Step 3.2. We have (from KKT)

$$Df(\vec{x}^*) + \vec{\lambda}^T Dh(\vec{x}^*) + \mu_1 Dg_1(\vec{x}^*) + \sum_{j=2}^{p} \mu_j Dg_j(\vec{x}^*) = \vec{0}^T.$$

We multiply on the right by our $\vec{v} \in T_{\widehat{S}}(\vec{x}^*)$. Looking at (2.3) we see that $Dh(\vec{x}^*)\vec{v} = 0$, and for each $j > 1$, either $\mu_j = 0$ (if the constraint is inactive) or $Dg_j(\vec{x}^*)\vec{v} = 0$. Thus we obtain

$$Df(\vec{x}^*)\vec{v} = -\mu_1 Dg_1(\vec{x}^*)\vec{v} < 0$$

since $Dg_1(\vec{x}^*)\vec{v} < 0$ and $\mu_1 < 0$. But this means there is a direction *tangent to* $\widehat{S}$ in which f is *decreasing*. Thus we can decrease f within the feasible set $\widehat{S}$ and so $\vec{x}^*$ cannot be a local minimum.

This concludes the proof of KKT. □

EXAMPLE 2.20. A simple example with only one equality and one inequality constraint, both of which are linear: minimize $f(x, y) = (x - 1)^2 + y - 2$ subject to $h(x, y) = y - x - 1 = 0$, $g(x, y) = x + y - 2 \leq 0$.

Here $Dh(\vec{x}) = [-1, 1]$ and $Dg(\vec{x}) = [1, 1]$ which are always linearly independent. KKT, plus

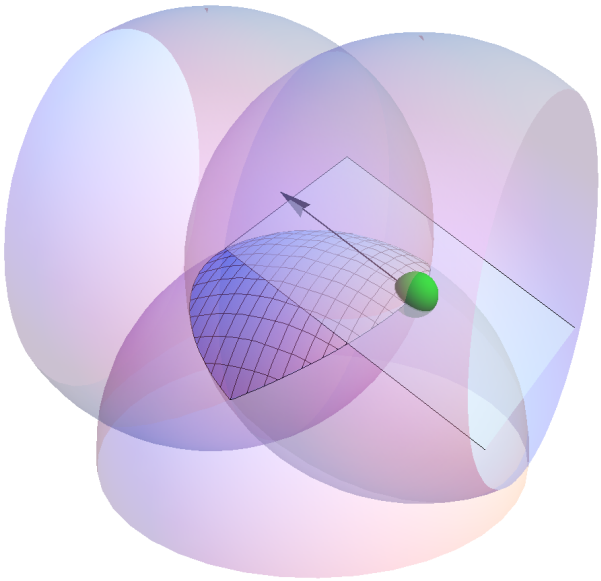


Figure 2.13: A direction in which the objective function decreases within the feasible set.

feasibility, becomes

$$\begin{aligned}
[2x - 2 - \lambda + \mu, 1 + \lambda + \mu] &= \vec{0}^T \\
\mu(x + y - 2) &= 0 \\
\mu &\geq 0 \\
y - x - 1 &= 0 \\
x + y - 2 &\leq 0.
\end{aligned}$$

The general approach is to consider the system of *equalities* and then check solutions to that system in the *inequalities*. We see that either $\mu = 0$ or $g(\vec{x}) = x + y - 2 = 0$. If $g(\vec{x}) = 0$, then we require

$$\begin{aligned}
2x - 2 - \lambda + \mu &= 0 \\
1 + \lambda + \mu &= 0 \\
y - x - 1 &= 0 \\
x + y - 2 &= 0
\end{aligned}$$

to which we find the solution has $\mu = 0$, so in fact both cases are treated at the same time: with $\mu = 0$, we have

$$\begin{aligned}
2x - 2 - \lambda &= 0 \\
1 + \lambda &= 0 \\
y - x - 1 &= 0
\end{aligned}$$

yielding $x = 1/2$, $y = 3/2$, $\lambda = -1$. We have to check feasibility: it is satisfied here since $g(\vec{x}) = 0$. See Figure 2.14. We should also consider the fact that the feasible set is unbounded and so see what happens as $(x, y) \to (-\infty, -\infty)$. Since $y - x - 1 = 0$, we can substitute $y = x + 1$ into f to obtain $(x-1)^2 + (x-1) = (x-1)x$ which clearly goes to infinity as $x \to -\infty$. Thus the point found is the minimum. □

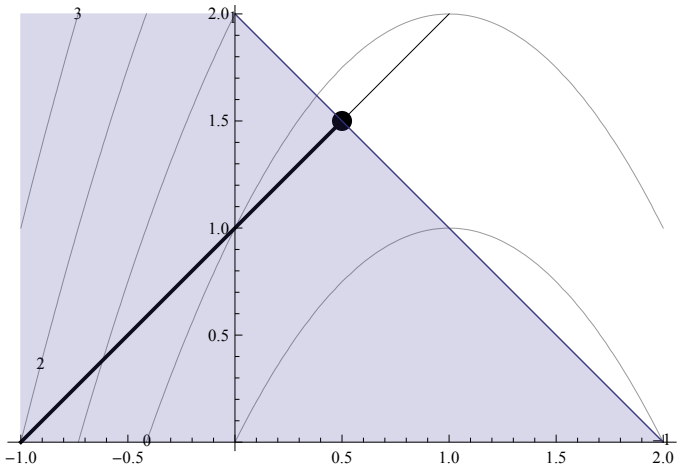


Figure 2.14: (Example 2.20) level sets of the objective function, the inequality constraint (shaded), the feasible set (the thick line), and the optimal point.

EXAMPLE 2.21. Minimize $f(x, y) = -3x + \frac{1}{2}y^2$ subject to $x^2 + y^2 - 1 \leq 0$, $x \geq 0$, $y \geq 0$. We re-write the inequality constraints as $-x \leq 0$, and $-y \leq 0$. Then KKT reads

$$-3 + 2x\mu_1 - \mu_2 = 0 \tag{2.4}$$
$$y + 2y\mu_1 - \mu_3 = 0 \tag{2.5}$$
$$\mu_1(x^2 + y^2 - 1) = 0 \tag{2.6}$$
$$\mu_2 x = 0 \tag{2.7}$$
$$\mu_3 y = 0 \tag{2.8}$$
$$\vec{\mu} \geq 0, \tag{2.9}$$

plus feasibility. By (2.6), $\mu_1 = 0$ or $x^2 + y^2 = 1$.

If $\boxed{\mu_1 = 0}$ then by (2.4), $\mu_2 = -3$ which contradicts (2.9).

If $\boxed{x^2 + y^2 = 1}$, then by (2.7) either $\mu_2 = 0$ or $x = 0$.

- If $\boxed{\mu_2 = 0}$ then by (2.8) either $\mu_3 = 0$ or $y = 0$.
  - If $\boxed{\mu_3 = 0}$ then (2.5) $\Rightarrow y(1 + 2\mu_1) = 0$. If $1 + 2\mu_1 = 0$ then $\mu_1 < 0$ which contradicts (2.9). Thus $y = 0$, and then since $x^2 + y^2 = 1$, $x = \pm 1$.

When $x = 1$, (2.4) $\Rightarrow \mu_1 = \dfrac{3}{2}$, and so $[1,0]^T$ is a candidate (it is feasible and has $\vec{\mu} = [\frac{3}{2}, 0, 0]^T$).

$x = -1$ is not feasible.

– If $\boxed{y = 0}$ then $x = \pm 1$, which we have already considered in the last case.

- If $\boxed{x = 0}$ then (2.4) $\Rightarrow \mu_2 = -3$ which contradicts (2.9).

Thus, the only candidate is $[1,0]^T$, where $f(1,0) = -3$. It is interesting to interpret the answer in terms of the active/inactive constraints. Despite the fact that $\mu_3 = 0$, the constraint $y \geq 0$ is still active (we remarked earlier that while inactive implies the corresponding $\mu = 0$, the converse need not be true); the constraint $x^2 + y^2 - 1 \leq 0$ is also active, that is, $x^2 + y^2 - 1 = 0$. What, then, does it mean that $\mu_3 = 0$? While the constraint is, technically, active, it is playing no role (one might argue that it is really "not active"). The gradient of f at $[1,0]^T$ is $\nabla f(1,0) = [-3,0]^T$ which is *tangent* to the constraint $y = 0$, so this constraint is playing no role; if it weren't for the other constraint ($x^2 + y^2 \leq 1$), we could decrease f by moving further out along the x-axis. See Figure 2.15. □

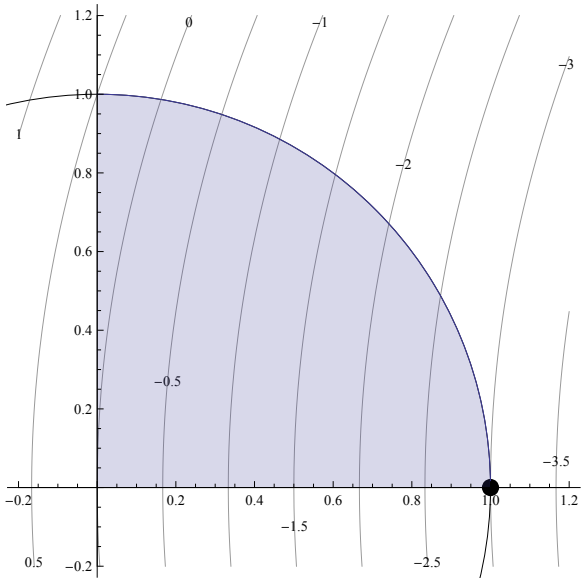


Figure 2.15: (Example 2.21) level sets of the objective function, the feasible set, and the optimal point.

Example 2.21 above was certainly manageable by hand, but it is believable that the system of equations one needs to solve can easily become unmanageable. We consider here how to use the symbolic computation ability of *Mathematica* to help us. The basic function we use is `Solve[ ]`. This example is simple enough that we can ask *Mathematica* to solve the full system, as follows: (See the *Mathematica* notebook "Constrained Ex 2-21.nb")

```
results = { x, y, μ1, μ2, μ3 } /.
  Solve[{
    -3 + 2x μ1 - μ2 == 0,
    y + 2y μ1 - μ3 == 0,
    μ1(x^2 + y^2 - 1) == 0,
    x μ2 == 0,
    y μ3 == 0,
    μ1 >= 0,
    μ2 >=0 },
  { x, y, μ1, μ2, μ3 }]
```

Coding notes:

- `Solve[ x^2 == 4, x]` returns the list of *replacement rules*, `{ {x -> 2}, {x -> -2} }`. Putting `x /. { {x -> 2}, {x -> -2} }` causes application of these replacements, resulting the list `{ 2, -2 }`. This explains the replacement command in the above code.

- Use of a single = causes *assignment*; use of double == causes *testing of equality* and returns either `True` or `False`.

The above code returns a single set of solutions: `{{1, 0, 3/2, 0, 0}}`. You will notice that there is no test for *feasibility* of the point(s) found. This test could be included as further inequalities, or we can now check by hand (or using *Mathematica*) which, if any, of the found solutions are feasible. We also need to evaluate the objective function at the places found to see which is optimal. All these steps could be built into the above code, but it is advisable not to try to do too much in one step. Indeed, in more complicated examples, you might find the system is too much for *Mathematica* to handle in this full form. Progress can still be made by asking *Mathematica* to do less at the first step. For example, a good option is to first solve without any *inequalities* as in:

```
results = { x, y, μ1, μ2, μ3 } /.
  Solve[{
    -3 + 2x μ1 - μ2 == 0,
    y + 2y μ1 - μ3 == 0,
    μ1(x^2 + y^2 - 1) == 0,
    x μ2 == 0,
    y μ3 == 0,
  { x, y, μ1, μ2, μ3 }]
```

This results in invalid solutions, such as complex valued solutions, and vectors $\vec{\mu} < \vec{0}$. Thus, we must post-process the solutions to remove unwanted ones. The command below first keeps only those solutions which are real-valued:

```
realresults = Cases[results, _?(Element[#, Reals]&)]
```

Here `Cases[list, _?(test function)]` keeps only those elements of `list` which pass the `test function`. This `test function` must be a *pure function* – see below. The test here is whether the element of `list` is real; `Element[#, Reals]&` returns `True` if `#` is real.

Pure functions: the way to encode a pure function in *Mathematica* is to use `#` as the variable (the fact that it is un-specified is what makes the function a pure function – any variable can take the place of `#`) and to follow the definition with &. For example `#^2&` is the pure "squared" function. It is shorthand for `Function[ #^2 ]`. This long-hand version actually allows you to use temporary variables, for example `Function[ {x, y}, x^2 + y^2 ]` is a pure function. The short-hand version of this would be `(#1^2 + #2^2)&`.

We still have to remove solutions which have $\vec{\mu} < \vec{0}$ (if we are looking for minima). To do this we can use:

```
newresults =
  DeleteCases[realresults, _?(Min[Take[#, {3, 5}]]<0&)]
```

Here, `Take[#, {3, 5}]` extracts elements 3 through 5 (those corresponding to $\vec{\mu}$). Next, `Min[ ]` returns the minimum of these elements 3 through 5. Finally, `DeleteCases[list, _?(test function)]` removes any elements which satisfies the test. Thus we remove any elements for which at least one of the μ_j is negative.

EXAMPLE 2.22. (An example from Pedregal, "Introduction to Optimization".) Minimize

$$f(x_1, x_2, x_3) = x_3 + \frac{1}{2}\left(x_1^2 + x_2^2 + \frac{1}{10}x_3^2\right)$$

subject to $x_1 + x_2 + x_3 = r$, $x_i \geq 0$. Assume $r > 0$. KKT gives:

$$x_1 - \mu_1 + \lambda = 0 \tag{2.10}$$
$$x_2 - \mu_2 + \lambda = 0 \tag{2.11}$$
$$1 + \frac{1}{10}x_3 - \mu_3 + \lambda = 0 \tag{2.12}$$
$$\mu_1 x_1 = \mu_2 x_2 = \mu_3 x_3 = 0 \tag{2.13}$$
$$\vec{\mu} \geq \vec{0} \tag{2.14}$$
$$-x_j \leq 0, \quad j = 1, 2, 3 \tag{2.15}$$
$$x_1 + x_2 + x_3 - r = 0. \tag{2.16}$$

$\boxed{x_1 = 0}$:

- $\boxed{x_2 = 0}$:
  - $\boxed{x_3 = 0}$: contradicts (2.16).

  - $\boxed{\mu_3 = 0}$: $\Rightarrow x_3 = r \Rightarrow \mu_1 = \mu_2 = \lambda = -1 - r/10$; contradicts (2.14).

- $\boxed{\mu_2 = 0}$:

  - $\boxed{x_3 = 0}$: $\Rightarrow x_2 = r$, $\lambda = \mu_1$, $x_2 = -\lambda$, which together imply $\mu_1 = -r$; contradicts (2.14).
  - $\boxed{\mu_3 = 0}$: $\Rightarrow \lambda = \mu_1$, $x_2 = -\lambda$, which together give $\mu_1 = -x_2 \leq 0 \Rightarrow \mu_1 = x_2 = 0$ which we have already dealt with.

$\boxed{\mu_1 = 0}$:

- $\boxed{x_2 = 0}$:

  - $\boxed{x_3 = 0}$: this is equivalent to $x_1 = \mu_2 = 0$, $x_3 = 0$ above.
  - $\boxed{\mu_3 = 0}$: this is equivalent to $x_1 = \mu_2 = 0$, $\mu_3 = 0$ above.

- $\boxed{\mu_2 = 0}$:

  - $\boxed{x_3 = 0}$: $\Rightarrow x_1 = x_2 = -\lambda$, $\mu_3 = 1 + \lambda$ and $x_1 + x_2 = r$ which gives $x_1 = x_2 = \dfrac{r}{2}$; we find $\lambda = -\dfrac{r}{2}$ and then $\mu_3 = 1 - \dfrac{r}{2}$. In order to satisfy (2.14), we thus need $r \leq 2$. For such r, this yields the candidate $\boxed{\vec{x} = \left[\dfrac{r}{2}, \dfrac{r}{2}, 0\right]^T}$, with $\boxed{p(\vec{x}) = \dfrac{1}{4}r^2}$;
  - $\boxed{\mu_3 = 0}$: $\Rightarrow x_1 = x_2 = -\lambda$, $1 + x_3/10 = -\lambda$ and $x_1 + x_2 + x_3 = r$. This gives $\boxed{\vec{x} = \left[\dfrac{r+10}{12}, \dfrac{r+10}{12}, \dfrac{10(r-2)}{12}\right]^T}$. Notice this can only hold for $r \geq 2$ for otherwise we would fail $x_3 \geq 0$. We have $\boxed{p(\vec{x}) = \dfrac{1}{24}(-20 + 20r + r^2)}$.

So, for any $r > 0$ we have a unique minimizer (the two coincide when $r = 2$). The nature of the solution changes from being on the x_1x_2-plane *and* on $x_1 + x_2 + x_3 = r$ for $r \leq 2$ (the constraint $x_3 \geq 0$ is active) to being on $x_1 + x_2 + x_3 = r$ with the inequality constraints inactive for $r \geq 2$. This is demonstrated graphically in Example 2.23 below, for a lower dimensional case. □

EXAMPLE 2.23. A lower dimensional example of the above is to consider: minimize

$$f(x_1, x_2) = x_2 + \frac{1}{2}\left(x_1^2 + \frac{1}{10}x_2^2\right)$$

subject to $x_1 + x_2 = r$, $x_i \geq 0$. Assume $r > 0$. KKT gives:

$$\begin{aligned} x_1 - \mu_1 + \lambda &= 0 \\ 1 + \frac{1}{10}x_2 - \mu_2 + \lambda &= 0 \\ \mu_1 x_1 = \mu_2 x_2 &= 0 \\ \vec{\mu} &\geq \vec{0} \\ -x_j \leq 0, \quad & j = 1, 2 \\ x_1 + x_2 - r &= 0. \end{aligned}$$

Solving this system of equations yields (in a manner similar to Example 2.22) optimal point $\vec{x}^* = [r, 0]^T$ for $r \leq 1$ and $\vec{x}^* = \left[\frac{10 + r}{11}, \frac{10(r-1)}{11}\right]$ for $r \geq 1$ ($r = 1$ is the transition value for this problem). Looking at Figure 2.16, we see that the nature of the solution changes. For $r \leq 1$, the minimum occurs on the active constraint $x_2 = 0$; for $r \geq 1$, the minimum occurs at a location where the constraint is tangent to the level curve of f. □

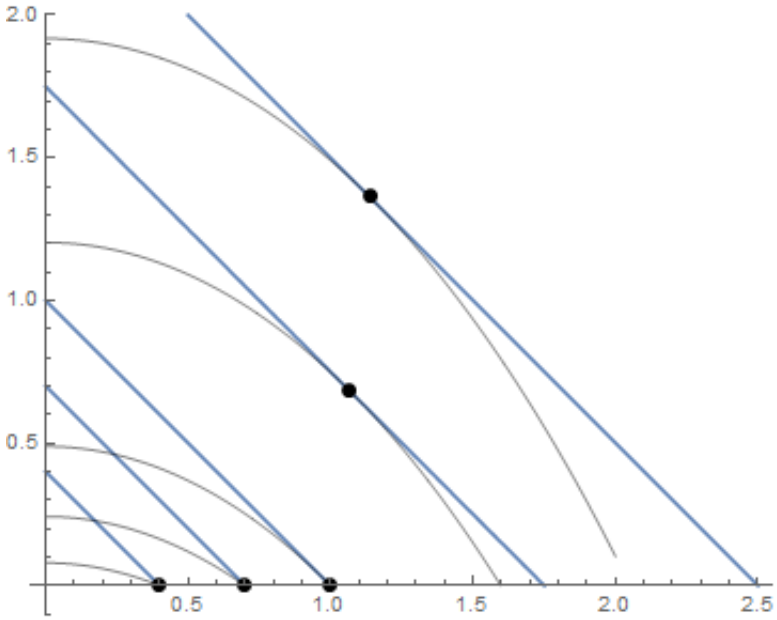


Figure 2.16: For Example 2.23, feasible sets (the line segments) and locations of the optimal points for $r = 0.5, 0.7, 1.7, 1.75, 2.5$, and the level curves of the objective function.

Example 2.22 above introduces difficulties when using *Mathematica* to find solutions, due to the presence of the parameter r. Having produced the list `results` as in the code below Example 2.21, we first remove any complex valued elements – the difficulty is that *Mathematica* doesn't know that r is real-valued. We can use `Simplify[ ]` together with `Assuming[ ]` as follows: (See the *Mathematica* notebook "Constrained Ex 2-22.nb")

```
realresults =
  Cases[results, _?(
    Assuming[r>0,
      Simplify[Element[#, Reals]]]&)
  ]
```

Next, we want to remove any negative $\vec{\mu}$'s. Again, we use `Simplify[ ]` together with `Assuming[ ]`:

```
DeleteCases[realresults,
  _?(Assuming[r>0, Simplify[Min[Take[#, {5, 7}]]<0]]]&)
]
```

Note that `{5, 7}` corresponds to the ordering `{x1, x2, x3, λ, μ1, μ2, μ3 }`.

2.3 Convexity and Optimization Problems

When the feasible set is unbounded, there may not exist a global (or even) local minimum. To know whether Lagrange's theorem, or KKT, has provided a minimum then requires analyzing the behavior of the objective function "at infinity" which can be a difficult task. Furthermore, it is sometimes not clear whether a given feasible set is bounded or not. To overcome these problems, we wish to provide some *sufficient conditions* for KKT to guarantee that we have a minimum/maximum. Convexity provides this.

2.3.1 Convexity and optimization

For functions of one variable: the necessary condition for a differentiable function to have a minimum at x^* is $f'(x^*) = 0$. This is KKT in dimension one. How about sufficient conditions?

- If $f(x) \to \infty$ as $x \to \pm\infty$, then $f'(x) = 0$ has *at least one* solution; at least one of the solutions must correspond to the global minimum of f.
- If $f''(x) > 0$ for all x, then f is strictly convex (concave up) and $f'(x) = 0$ can have *at most one* solution.

If we can (somehow) guarantee both of these, then the unique solution to $f'(x) = 0$ gives the global minimum.

Convex sets: (B.6.1, B.6.5.) A set K in $\mathbb{R}^n$ is convex if whenever $\vec{x}, \vec{y} \in K$ it holds that

$$t\vec{x} + (1-t)\vec{y} \in K \qquad \text{for all } 0 \leq t \leq 1.$$

Geometrically, this says that if $\vec{x}$ and $\vec{y}$ are in K then so too is the *entire straight line-segment* joining $\vec{x}$ to $\vec{y}$.

Convex functions: $f : K \subset \mathbb{R}^n \to \mathbb{R}$ is a convex function if K is convex and

$$f(t\vec{x} + (1-t)\vec{y}) \leq tf(\vec{x}) + (1-t)f(\vec{y}), \qquad 0 \leq t \leq 1. \tag{2.17}$$

We say f is strictly convex if the inequality is strict.

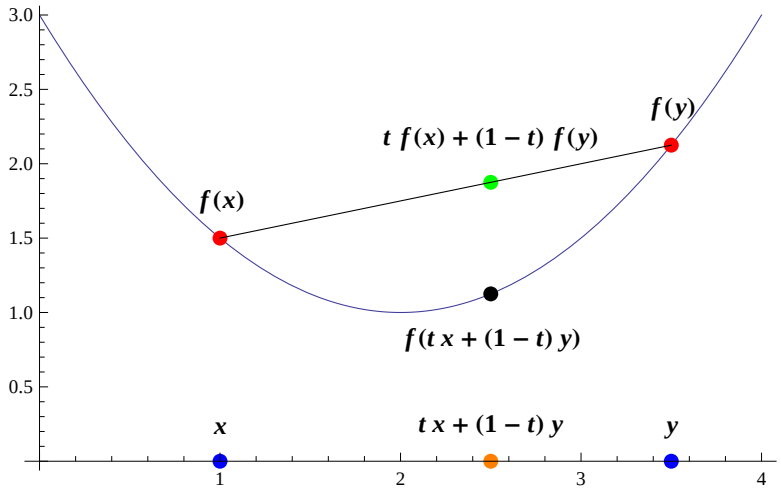


Figure 2.17: Graphical representation of the definition of convexity of a function.

EXERCISE 2.24. If $f : K \subset \mathbb{R}^n \to \mathbb{R}$ is a convex function, show that the set lying above the graph of f is a convex set. This explains the terminology.

THEOREM 2.25. *If $f : K \subset \mathbb{R}^n \to \mathbb{R}$ is convex, then $\vec{x}^*$ is a local minimizer for f if, and only if, it is a global minimizer on K.*

Proof. The "if" is obvious. Now suppose that $\vec{x}^*$ is a local minimizer, so there is $\varepsilon > 0$ such that $f(\vec{x}^*) \leq f(\vec{x})$ for all $\vec{x} \in B_\varepsilon(\vec{x}^*) \cap K$. Now let $\vec{y} \in K$ be arbitrary. For all $0 \leq t \leq \varepsilon/\|\vec{x}^* - \vec{y}\|$, $(1-t)\vec{x}^* + t\vec{y} \in B_\varepsilon(\vec{x}^*)$ and by convexity,

$$f(\vec{x}^*) \leq f\big(\big(1-t\big)\vec{x}^* + t\vec{y}\big) \leq (1-t)f(\vec{x}^*) + tf(\vec{y}) \qquad \Rightarrow \qquad tf(\vec{x}^*) \leq tf(\vec{y}).$$

Taking $t > 0$ yields the result. □

2.3.2 Other characterizations of convexity

If f is differentiable, we can characterize convexity in easier ways. We need the notion of positive definiteness of a quadratic form, and of a matrix – see B.5. A quadratic form is a function $f : \mathbb{R}^n \to \mathbb{R}$ which can be written in the form $f(\vec{x}) = \vec{x}^T Q \vec{x}$, where we can assume

that Q is a symmetric, real-valued, $n \times n$ matrix. The quadratic form is said to be *positive definite* if $f(\vec{x}) > 0$ for all $\vec{x} \neq \vec{0}$, and *positive semi-definite* if $f(\vec{x}) \geq 0$ for all $\vec{x}$ (see B.5.1). We carry these definitions over to the matrix Q itself. There are various ways to determine the definiteness of a matrix. See B.5.2, B.5.3 and B.5.5.

PROPOSITION 2.26. *Let $f : K \subset \mathbb{R}^n \to \mathbb{R}$ with K convex and open. ("Open" means, loosely, that K does not include a boundary. See B.6.7 for a precise definition.)*

1. *If $f \in \mathcal{C}^1$ (B.7.7) then f is convex iff*

$$f(\vec{y}) \geq f(\vec{x}) + Df(\vec{x})(\vec{y} - \vec{x}), \qquad \text{for all } \vec{x}, \vec{y} \in K. \tag{2.18}$$

2. *If $f \in \mathcal{C}^2$ then f is convex iff $D^2 f(\vec{x})$ is positive semi-definite (B.5) at all $\vec{x} \in K$.*

Remarks:

- $D^2 f$ is the Hessian matrix of second derivatives of f; it is symmetric if $f \in \mathcal{C}^2$ for then the mixed partial derivatives are independent of the order of differentiation.
- Condition 1. has an informative geometric meaning: in dimensions $n = 1, 2$, it says that the graph of f lies *above* the tangent line/plane based at any point. For higher dimensions, the same is true but for the tangent hyperplane.
- Condition 2. is a generalization of "concave up." We've actually seen something similar to this before in Math 224: recall the second derivative test:

$$f_{xx}(\vec{x}) f_{yy}(\vec{x}) - f_{xy}(\vec{x})^2 = \det \begin{bmatrix} f_{xx}(\vec{x}) & f_{xy}(\vec{x}) \\ f_{xy}(\vec{x}) & f_{yy}(\vec{x}) \end{bmatrix} = \det\ D^2 f(\vec{x}).$$

  Now if this is positive, *and* f_{xx} (or f_{yy}) is positive, then we know there is a "point of minimum," i.e. the function is convex there! Notice that f_{xx} and $\det\ D^2 f(\vec{x})$ are the determinants of the leading principal minors (see B.5.2). If these are *strictly* positive, then Sylvester's criterion (B.5.2) guarantees that $D^2 f(\vec{x})$ is positive definite. When $D^2 f(\vec{x})$ is *positive* definite, then Condition 2. is a direct generalization of the second derivative test to higher dimensions; for $\vec{x}$ where $D^2 f(\vec{x})$ is only positive *semi*-definite, we don't have Sylvester's criterion making the connection to the second derivative test.

Proof of Part 1. of Proposition 2.26. 1.1. ($\Rightarrow$) For any $\vec{x}, \vec{y} \in K$ we define the unit vector $\vec{u} = \dfrac{\vec{y} - \vec{x}}{\|\vec{y} - \vec{x}\|}$. Then for $0 < t < \|\vec{x} - \vec{y}\|$, $\vec{x} + t\vec{u}$ is a point on the line between $\vec{x}$ and $\vec{y}$, so by convexity of f, the value of f at $\vec{x} + t\vec{u}$ is less than or equal to the value of the linear function joining $[\vec{x}, f(\vec{x})]^T$ to $[\vec{y}, f(\vec{y})]^T$. This latter value can be found by considering the slope of the line and the fact that the distance between $\vec{x}$ and $\vec{x} + t\vec{u}$ is t: see Figure 2.18. Thus,

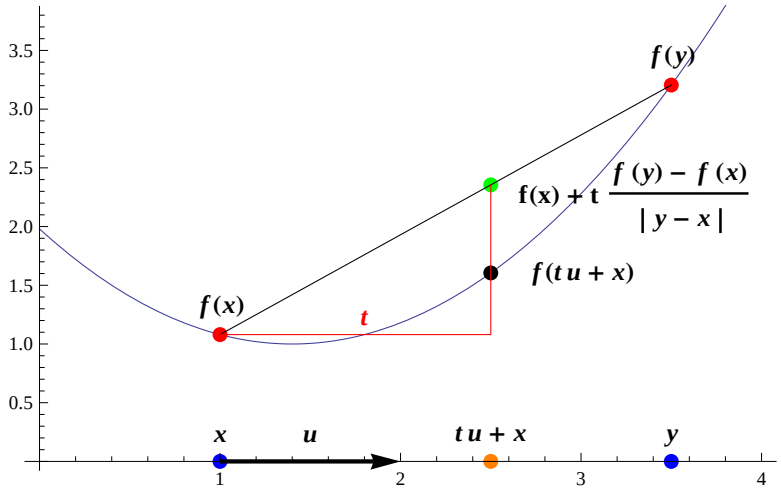


Figure 2.18: Graphical explanation of (2.19).

$$f\Big(\vec{x} + t\frac{\vec{y}-\vec{x}}{\|\vec{y}-\vec{x}\|}\Big) \le f(\vec{x}) + t\frac{f(\vec{y})-f(\vec{x})}{\|\vec{x}-\vec{y}\|}. \tag{2.19}$$

Recall that for $f : \mathbb{R}^n \to \mathbb{R}$, if $\vec{u}$ is a *unit* vector, then $\nabla f(\vec{x}) \cdot \vec{u}$ is the instantaneous rate of change of f in the direction $\vec{u}$ from x. Using the definition of instantaneous rate of change (the limit of change in f over change in position), recalling that $Df(\vec{x}) = \nabla f(\vec{x})^T$, and computing with the unit vector $\vec{u} = (\vec{y}-\vec{x})/\|\vec{y}-\vec{x}\|$, (2.19) implies

$$\begin{aligned} Df(\vec{x})\frac{\vec{y}-\vec{x}}{\|\vec{y}-\vec{x}\|} &= \lim_{t\to 0+} \frac{f\Big(\vec{x} + t\frac{\vec{y}-\vec{x}}{\|\vec{y}-\vec{x}\|}\Big) - f(\vec{x})}{t} \qquad \text{(definition of rate of change)} \\ &\le \frac{f(\vec{y})-f(\vec{x})}{\|y-x\|} \end{aligned}$$

Rearranging this gives (2.18).

1.2. ($\Leftarrow$) Let $\vec{x}, \vec{y} \in K$ and $0 < t < 1$. Then $\vec{w} = t\vec{x} + (1-t)\vec{y} \in K$ and by assumption

$$f(\vec{x}) \ge f(\vec{w}) + Df(\vec{w})(\vec{x}-\vec{w}), \qquad f(\vec{y}) \ge f(\vec{w}) + Df(\vec{w})(\vec{y}-\vec{w}).$$

Taking t times the first plus $(1-t)$ times the second,

$$tf(\vec{x}) + (1-t)f(\vec{y}) \ge f(\vec{w}) + Df(\vec{w})\big(t\vec{x} + (1-t)\vec{y} - \vec{w}\big) = f\big(t\vec{x} + (1-t)\vec{y}\big)$$

from the definition of $\vec{w}$. □

Proof of Part 2. of Proposition 2.26. (Optional)

We'll need **Taylor's Theorem** for functions of one variable: If $f \in \mathcal{C}^n$ then

$$\begin{aligned} f(x) = f(a) + f'(a)(x-a) + \frac{1}{2}f''(a)(x-a)^2 + \cdots \\ + \frac{1}{(n-1)!}f^{(n-1)}(a)(x-a)^{n-1} + \frac{1}{n!}f^{(n)}(s)(x-a)^n \end{aligned}$$

for some s between x and a. This follows simply from the Mean Value Theorem.

On with the proof. We first consider a function h of one variable and prove that $h : I \to \mathbb{R}$ is convex iff $h''(x) \geq 0$ on the open interval I.

2.1. ($\Rightarrow$) We prove the contrapositive: if somewhere $h''(x) < 0$ then we will show that h is not convex. Since I is open there is $\delta > 0$ so that $h''(z) < 0$ for all $x \leq z \leq x + \delta$ and $y = x + \delta \in I$. By Taylor's theorem there is $x < s < y$ such that

$$h(y) = h(x) + h'(x)(y - x) + \frac{1}{2}(y - x)^2 h''(s).$$

Since $x < s < y = x + \delta$, $h''(s) < 0$ so $h(y) < h(x) + h'(x)(y - x)$ which implies h is not convex by part 1.

2.2. ($\Leftarrow$) Fix $x, y \in I$. By Taylor's theorem there is $x < s < y$ such that

$$h(y) = h(x) + h'(x)(y - x) + \frac{1}{2}(y - x)^2 h''(s).$$

Since $h''(s) > 0$, $h(y) > h(x) + h'(x)(y - x)$ and h is convex by part 1.

Now extend this to $f : K \subset \mathbb{R}^n \to \mathbb{R}$:

2.3 ($\Rightarrow$) fix $\vec{x} \in K$ and unit vector $\vec{u} \in \mathbb{R}^n$, and define

$$h(t) = f(\vec{x} + t\vec{u}),$$

for $-\varepsilon \leq t \leq \varepsilon$ sufficiently small that $\vec{x} + t\vec{u} \in K$ for all such t. By the chain rule, $h'(s) = Df(\vec{x} + s\vec{u})\vec{u}$ and $h''(s) = \frac{1}{2}\vec{u}^T D^2 f(\vec{x} + s\vec{u})\vec{u}$. Now

$$\begin{aligned}
f \text{ is convex} \Rightarrow h(t) &= f(\vec{x} + t\vec{u}) \\
&\geq f(\vec{x} + s\vec{u}) + Df(\vec{x} + s\vec{u})(t - s)\vec{u},\ \forall s, t && \text{by (2.18)} \\
&= h(s) + h'(s)(t - s) \\
&\Rightarrow h \text{ is convex} && \text{by (2.18) again} \\
&\Rightarrow h''(s) \geq 0,\ \forall s && \text{by the 1-D proof above} \\
&\Rightarrow \vec{u}^T D^2 f(\vec{x} + s\vec{u})\vec{u} \geq 0.
\end{aligned}$$

In particular, $\vec{u}^T D^2 f(\vec{x})\vec{u} \geq 0$, and for arbitrary $\vec{v} \neq \vec{0}$, $\vec{v}^T D^2 f(\vec{x})\vec{v} = \|\vec{v}\|^2 \vec{u}^T D^2 f(\vec{x})\vec{u} \geq 0$, where $\vec{u} = \vec{v}/\|\vec{v}\|$. Since $\vec{u}$ and $\vec{x}$ were arbitrary, this proves $D^2 f$ is positive semi-definite on K.

2.4 ($\Leftarrow$) Fix $\vec{x}, \vec{y} \in K$, let $\vec{u} = \vec{y} - \vec{x}$, and define $h(t) = f(\vec{x} + t\vec{u})$ for $-\varepsilon < t < 1 + \varepsilon$, ε

sufficiently small that $x + t\vec{u} \in K$ for all such t. Then

$$\begin{aligned} D^2 f \text{ positive semi-definite} &\Rightarrow \vec{u}^T D^2 f(\vec{x} + t\vec{u})\vec{u} \geq 0, \ \forall t \\ &\Rightarrow h''(t) \geq 0, \ \forall t \\ &\Rightarrow h \text{ is convex} \\ &\Rightarrow h(t) \leq h(0) + h'(0)t \qquad \text{by (2.18)} \\ &\qquad = f(\vec{x}) + Df(\vec{x})t(\vec{y} - \vec{x}) \\ &\Rightarrow f(\vec{y}) \geq f(\vec{x}) + Df(\vec{x})(\vec{y} - \vec{x}) \qquad \text{setting } t = 1. \end{aligned}$$

Since $\vec{x}, \vec{y} \in K$ were arbitrary, this proves f is convex on K by (2.18) again. □

We also define **strictly convex functions** and Proposition 2.26 extends with strict inequalities ($\vec{x} \neq \vec{y}$). In particular, in Proposition 2.26 (2), we can use *Sylvester's criterion* (B.5.2) and check that the leading principal minors are all positive.

Examples of convex functions:

- affine functions are convex, not strictly convex;
- $T : \mathbb{R}^n \to \mathbb{R}^m$ linear, $f : \mathbb{R}^m \to \mathbb{R}$ convex $\Rightarrow f(T\vec{x})$ is convex;
- the supremum of a family of convex functions is again convex.

EXERCISE 2.27. Show that if $f : K \subset \mathbb{R}^n \to \mathbb{R}$ is convex, and if $g : \mathbb{R} \to \mathbb{R}$ is convex *and non-decreasing*, then $g(f(\vec{x}))$ is convex.

EXERCISE 2.28. Show that the functions $f(\vec{x}) = \|\vec{x}\|^m$, $m \geq 1$ are convex.

2.3.3 Sufficiency of KKT for convex problems

So long as there are no *equality* constraints, what is needed is that the feasible set K be a convex set, and that f restricted to K be a convex function. If, in fact, the constraint functions g_j are themselves globally convex, and the objective function f is globally convex, then the proof of sufficiency is more straight-forward, and so we present this first. However, convexity of the constraint functions g_j *implies* that the feasible set is convex, and are only interested in f on the feasible set, so requiring global convexity of f isn't needed. Thus, the subsequent theorem (Theorem 2.30) implies this first theorem.

THEOREM 2.29. *Assume $f, g \in \mathcal{C}^1(\mathbb{R}^n)$ are convex. Suppose there exist $\vec{x}^*, \vec{\mu}$ such that the KKT conditions*

1. $\vec{\mu} \geq \vec{0}$,

2. $Df(\vec{x}^*) + \vec{\mu}^T Dg(\vec{x}^*) = \vec{0}^T$,

3. $\vec{\mu}^T g(\vec{x}^*) = 0$

hold. Then $\vec{x}^*$ is necessarily *a (global) minimizer of f subject to the constraint $g(\vec{x}) \leq \vec{0}$.*

If, further, f is strictly convex, then $\vec{x}^*$ is the unique minimizer.

Proof. Let $\vec{y}$ be such that $g(\vec{y}) \leq \vec{0}$; then

$$\begin{aligned}
f(\vec{y}) &\geq f(\vec{x}^*) + Df(\vec{x}^*)(\vec{y} - \vec{x}^*) && \text{by convexity of } f \text{ and Proposition 2.26,} \\
&= f(\vec{x}^*) - \vec{\mu}^T Dg(\vec{x}^*)(\vec{y} - \vec{x}^*) && \text{by 2. in KKT} \\
&\geq f(\vec{x}^*) + \vec{\mu}^T(g(\vec{x}^*) - g(\vec{y})) && \text{by (global) convexity of (each component of) } g \\
& && \text{and Proposition 2.26, and by 1. in KKT} \qquad (2.20) \\
&= f(\vec{x}^*) - \vec{\mu}^T g(\vec{y}) && \text{by 3. in KKT} \\
&\geq f(\vec{x}^*) && \text{since } \vec{\mu} \geq 0 \text{ and } g(\vec{y}) \leq 0.
\end{aligned}$$

In the case of strict convexity, if $\vec{x} \neq \vec{y}$ then

$$\begin{aligned}
f(\vec{y}) &> f(\vec{x}^*) + Df(\vec{x}^*)(\vec{y} - \vec{x}^*) && \text{by strict convexity of } f \text{ and Proposition 2.26,} \\
&\geq f(\vec{x}^*) && \text{as above.}
\end{aligned}$$

□

As discussed prior to the statement, the hypotheses of Theorem 2.29 require more than is necessary. To see this we need to understand how convexity of the feasible set Ω can be used to obtain the same inequality (2.20) in the proof of Theorem 2.29 where global convexity of the components of g was used. With $\vec{x}^*$ and $\vec{y}$ as in Theorem 2.29, for any $0 \leq t \leq \|\vec{y} - \vec{x}^*\|$, since Ω is convex, $\vec{x}^* + t\dfrac{\vec{y} - \vec{x}^*}{\|\vec{y} - \vec{x}^*\|} \in \Omega$ is feasible, so $g(\vec{x}^* + t\vec{u}) \leq \vec{0}$ there (where $\vec{u}$ is the unit vector $\vec{u} = (\vec{y} - \vec{x}^*)/\|\vec{y} - \vec{x}^*\|$). Since $\vec{\mu} \geq \vec{0}$, we thus have $\vec{\mu}^T g(\vec{x}^* + t\vec{u}) \leq \vec{0}$, and since $\vec{\mu}^T g(\vec{x}^*) = 0$, we have,

$$\vec{\mu}^T g(\vec{x}^* + t\vec{u}) - \mu^T g(\vec{x}^*) \leq 0.$$

Dividing this by $t > 0$ we then have

$$\vec{\mu}^T \Big(\frac{g(\vec{x}^* + t\vec{u}) - g(\vec{x}^*)}{t}\Big) \leq 0. \qquad (2.21)$$

Taking the limit as $t \to 0^+$, we have

$$\begin{aligned}
\lim_{t \to 0^+} \frac{g(\vec{x}^* + t\vec{u}) - g(\vec{x}^*)}{t} &= \text{the directional derivative of } g \text{ at } \vec{x}^* \text{ in the direction of } \vec{u} \\
&= \nabla g(\vec{x}^*) \cdot \vec{u} \\
&= Dg(\vec{x}^*) \frac{\vec{y} - \vec{x}^*}{\|\vec{y} - \vec{x}^*\|}.
\end{aligned}$$

Combining this with (2.21) (and multiplying through by $\|\vec{y} - \vec{x}^*\|$) we obtain

$$\vec{\mu}^T Dg(\vec{x}^*)(\vec{y} - \vec{x}^*) \leq 0.$$

This is precisely the inequality needed (2.20) and so following the proof of Theorem 2.29 we have proved:

THEOREM 2.30. *Let $f : \mathbb{R}^n \to \mathbb{R}$, $f \in \mathcal{C}^1$ be a convex function on the feasible set $\Omega = \{\vec{x} \in \mathbb{R}^n \ : \ g(\vec{x}) \leq \vec{0}\}$, where $g : \mathbb{R}^n \to \mathbb{R}^p$, $g \in \mathcal{C}^1$, and Ω is convex. Suppose there exist $\vec{x}^*, \vec{\mu}$ such that the KKT conditions*

1. $\vec{\mu} \geq \vec{0}$,
2. $Df(\vec{x}^*) + \vec{\mu}^T Dg(\vec{x}^*) = \vec{0}^T$,
3. $\vec{\mu}^T g(\vec{x}^*) = 0$

hold. Then $\vec{x}^*$ is a global minimizer of f over Ω.

EXERCISE 2.31. If $g : \mathbb{R}^n \to \mathbb{R}$ is convex, then for any $c \in \mathbb{R}$, the set $K_c = \{\vec{x} \in \mathbb{R}^n \ : \ g(\vec{x}) \leq c\}$ is a convex set.

Finally, if the objective function and the inequality constraint functions are convex functions *considered as functions on all of* $\mathbb{R}^n$, and the equality constraints are linear (or affine) then KKT is sufficient. This simply requires the observation that $h(\vec{x}) = 0$ is equivalent to $h(\vec{x}) \leq 0$ and $-h(\vec{x}) \leq 0$ and so convexity of h requires convexity of both h and $-h$, hence the requirement that h be linear (or affine).

EXAMPLE 2.32. Minimize $f(\vec{x}) = \|\vec{x}\|^4 + \|\vec{x} - \vec{a}\|^2$, $\|\vec{x}\|^2 \leq 1$. The following is useful:

$$\frac{\partial}{\partial x_i}\|\vec{x}\| = \frac{x_i}{\|\vec{x}\|},$$

so, for example,

$$\frac{\partial}{\partial x_i}\|\vec{x}\|^2 = 2\|\vec{x}\|^1 \cdot \frac{x_i}{\|\vec{x}\|} = 2x_i, \qquad \frac{\partial^2}{\partial x_i^2}\|\vec{x}\|^2 = 2$$
$$\frac{\partial}{\partial x_i}\|\vec{x}\|^4 = 4\|\vec{x}\|^3 \cdot \frac{x_i}{\|\vec{x}\|} = 4\|\vec{x}\|^2 x_i, \qquad \frac{\partial^2}{\partial x_i^2}\|\vec{x}\|^4 = 8x_i^2 + 4\|\vec{x}\|^2.$$

Notice that the feasible set is convex. Computing $D^2 f$, we find the diagonal elements are $2 + 8x_i^2 + 4\|\vec{x}\|^2$ and the off-diagonal i, j elements are $8x_i x_j$. It's not clear that this matrix is positive semi-definite. However, the function $\vec{x} \mapsto \|\vec{x}\|$ is convex since by the triangle inequality $\|t\vec{x}+(1-t)\vec{y}\| \leq t\|\vec{x}\|+(1-t)\|\vec{y}\|$. From this we can conclude that $x \mapsto \|\vec{x}\|^2$ and $\vec{x} \mapsto \|\vec{x}\|^4$ are

convex functions because α^2 and α^4 are increasing convex functions, for $\alpha \geq 0$, and certainly $\|\vec{x}\| \geq 0$. KKT gives

$$\begin{aligned} 4\|\vec{x}\|^2\vec{x} + 2(\vec{x} - \vec{a}) + 2\mu\vec{x} &= \vec{0} \\ \mu(\|\vec{x}\|^2 - 1) &= 0 \\ \mu &\geq 0. \end{aligned}$$

- $\boxed{\|\vec{x}\|^2 = 1}$: In this case, the first equation becomes $\vec{a} = 3\vec{x} + \mu\vec{x}$ and so $\vec{x} = t\vec{a}$ for some t; but since $\|\vec{x}\| = 1$, in fact $\vec{x} = \vec{a}/\|\vec{a}\|$, and the first equation becomes $\vec{a}(\|\vec{a}\| - 3 - \mu) = 0$. We can't have $\|\vec{a}\| = 0$, so $\mu = \|\vec{a}\| - 3$ and $\mu \geq 0$ implies $\|\vec{a}\| \geq 3$. Thus, for $\|\vec{a}\| \geq 3$, there is the candidate $\vec{x} = \vec{a}/\|\vec{a}\|$.

- $\boxed{\mu = 0}$: the first equation can be re-written $\vec{x}(2\|\vec{x}\|^2 + 1) = \vec{a}$ which again implies $\vec{x} = t\vec{a}$ for some t. Plugging this into the first equation yields $a_j(2t^3\|\vec{a}\|^2 + t - 1) = 0$, so either $a_j = 0$ for all j, or $a_j \neq 0$ for some j in which case $2t^3\|\vec{a}\|^2 + t - 1 = 0$. If $\vec{a} = \vec{0}$ then the problem clearly has minimizer $\vec{x} = \vec{0}$. Thus we seek the zeros of p:

  $$p(t) = 2t^3\|\vec{a}\|^2 + t - 1 = 0.$$

  To understand this cubic, notice that $p'(t)$ is never zero, so $p(t)$ always has a unique real root. All we need to do now is see whether these candidates ever occur at the same time as the candidate we found above. Writing $\|\vec{a}\| = \|\vec{x}\|/t$,

  $$\begin{aligned} 2t^3\|\vec{a}\|^2 + t - 1 = 0 \quad \Rightarrow \quad t(2\|\vec{x}\|^2 + 1) = 1 \quad &\Rightarrow \quad \|\vec{a}\|t(2\|\vec{x}\|^2 + 1) = \|\vec{a}\| \\ &\Rightarrow \quad \|\vec{x}\|(2\|\vec{x}\|^2 + 1) = \|\vec{a}\|. \end{aligned}$$

  Since $\|\vec{x}\| \leq 1$, we conclude

  $$0 < \|\vec{a}\| = \|\vec{x}\|\left|2\|\vec{x}\|^2 + 1\right| \leq 3.$$

  Thus these candidates do not coincide with the previous ones (except when $\|\vec{a}\| = 3$ when we can check that they give the same solution).

In the case $n = 2$, we can illustrate these results very clearly as in Figures 2.19, 2.20 and 2.21. □

2.4 Exercises

1. Exercise 2.9, page 21.

2. Exercise 2.12, page 22.

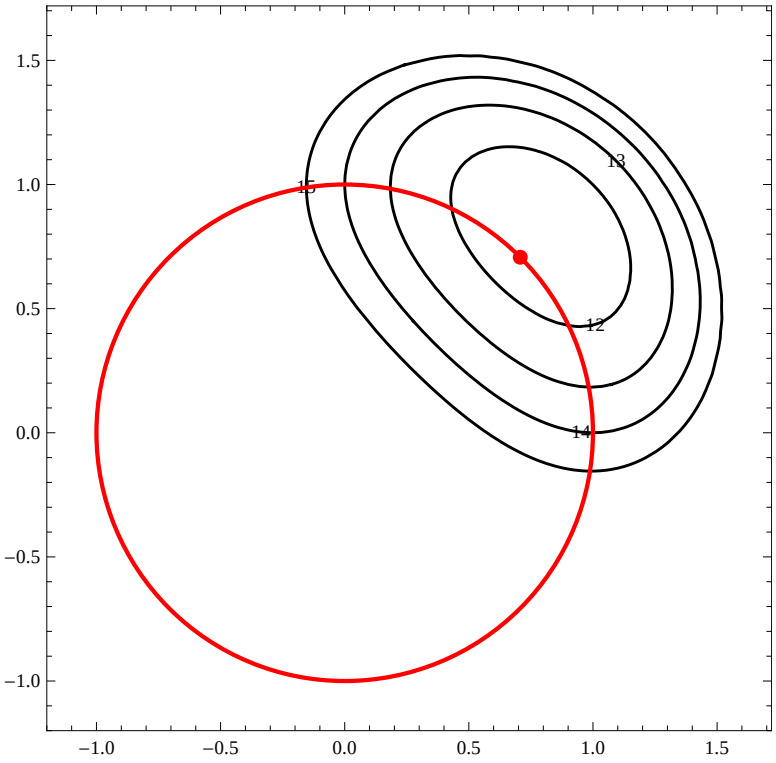


Figure 2.19: (Example 2.32) $\|\vec{a}\| > 3$, (e.g. $\vec{a} = [3,3]^T$ here).

3. Prove that the tangent space to the unit sphere, at the point $\vec{p} = [0,0,1]^T$, $T_{\vec{p}}\,\mathbb{S}^2$, is the set of vectors $S = \{[a,b,0]^T,\ a,b \in \mathbb{R}\}$. Note: if $\vec{v} = [a,b,0]^T$, it is quite easy to come up with a curve $\vec{x}(t)$ in the sphere with $\vec{x}(0) = [0,0,1]^T$ and with $\vec{x}\,'(0) = \vec{v}$; this shows that $S \subset T_{\vec{p}}\,\mathbb{S}^2$. To prove the reverse containment, you must start with an *arbitrary* curve $\vec{x}(t)$ in the sphere with $\vec{x}(0) = [0,0,1]^T$ and show that $\vec{x}\,'(0)$ is of the form $[a,b,0]^T$.

4. Find and classify the critical points of

   (a) $f(x,y) = (2-x-y)^2 + (1+x+y-xy)^2$.

   (b) $f(x,y) = (2-x-y)^2 + \dfrac{1}{3}(1+x+y-xy)^3$.

5. Let $a_1 \geq 0$ and $a_2 \geq 0$. The function $f(x_1,x_2) = a_1x_1^2 + a_2x_2^2$ is an elliptical parabolic bowl (perhaps degenerate if $a_1 = 0$ or $a_2 = 0$); the constraint $c_1x_1 + c_2x_2 = 1$ is clearly a straight line. Find the minimum and maximum of f subject to the given constraint. Be sure to consider all possible cases of the parameters a_j and c_j.

6. Consider the constrained problem of minimizing $f(\vec{x}) = \displaystyle\sum_{i=1}^{n} a_i x_i^2$ subject to $\displaystyle\sum_{i=1}^{n} c_i x_i = 1$, where $c_i \neq 0$ for all i and the $a_i \geq 0$ for all i (we allow the possibility of a_i's being zero). Note: the constraint is unbounded, so you need to consider what happens "at infinity." You will also need to consider the various cases where some, or all, the $a_i = 0$. Determine the location(s) of minima, when they exist.

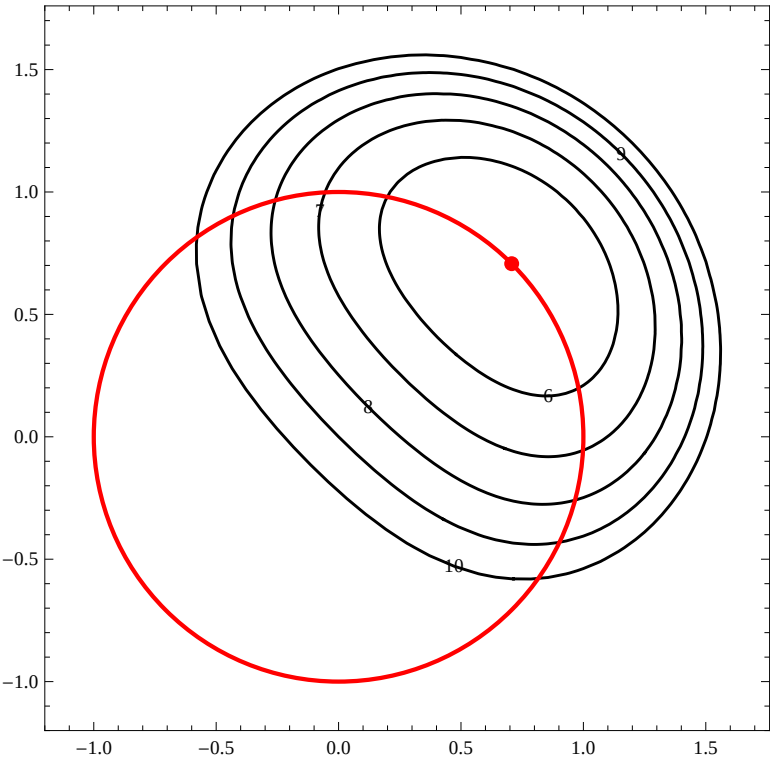


Figure 2.20: (Example 2.32) $\|\vec{a}\| = 3$ (e.g. $\vec{a} = [3/\sqrt{2}, 3/\sqrt{2}]^T$ here).

7. The function in Problem 5 has level sets which are ellipses with major and minor axes aligned with the coordinate axes, and centered at the origin (if $a_1 > 0$ and $a_2 > 0$). A more general elliptic bowl is given by

$$f(x_1, x_2) = a_{11}x_1^2 + 2a_{12}x_1x_2 + a_{22}x_2^2 + b_1x_1 + b_2x_2,$$

where the matrix $A = (a_{ij})$ is *positive definite*. Find the minimum and maximum of f subject to the constraint $c_1x_1 + c_2x_2 = 1$. Hint: work with matrices and vectors.

Does this problem, and your solution, generalize to $\mathbb{R}^n$?

8. Let

$$f(\vec{x}) = (x_1 - 1)^2 + \sum_{i=1}^{n-1}(x_{i+1} - x_i)^2 + x_n^2.$$

(a) Find the critical points of f. (Find an explicit formula for x_j.)

(b) Write the condition for a critical point in the form $A\vec{x} = \vec{b}$ and show that the matrix A is invertible. (One way to do this is to show that the unique solution to $A\vec{x} = \vec{0}$ is $\vec{x} = \vec{0}$.)

(c) Find the location of the minimum of f subject to the constraint $\sum_{i=1}^{n} a_ix_i = c$. (Hint: use part (b).) Note: the constraint function is unbounded, but f is bounded below (by 0 for example) and so must have a minimum.

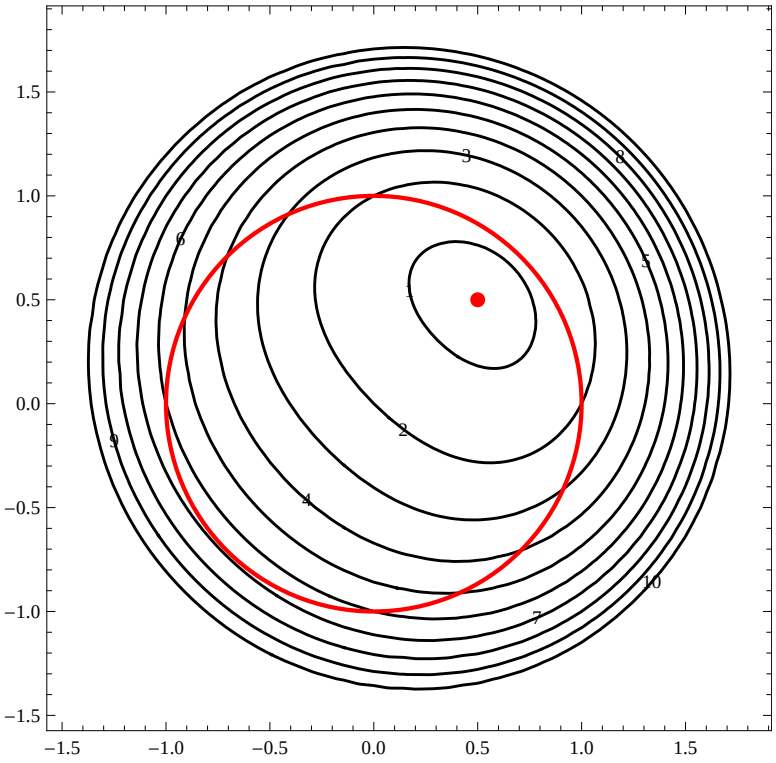


Figure 2.21: (Example 2.32) $\|\vec{a}\| < 3$ (e.g. $\vec{a} = [1, 1]^T$ here).

9. Find the minimum and maximum of $f(x, y) = \displaystyle\int_x^y \frac{1}{1+t^4}\, dt$ subject to the constraint $x^2y^2 = 1$. (Note: the feasible set is *unbounded* so you need to consider what happens as $\|[x, y]^T\| \to \infty$; *Mathematica* can find an anti-derivative of $1/(1 + t^4)$, but you should not use *Mathematica* here.)

10. Where is the surface $xy + yz + zx = 12$ closest to the origin? Without re-solving the problem, approximately how much closer to/further from the origin is the closest point on the surface $xy + yz + zx = 12.2$? Justify your answer.

11. Solve the shortest ladder problem: Problem 1. in Chapter 1.

12. Use *Mathematica* to find the minimum and maximum of $f(x, y, z) = (x+y)^2+z^3$ subject to the constraints $x^2 + y^2 + z^2 = 4$ and $x + y + z = 1$. If the constraints were changed to $x^2 + y^2 + z^2 = 4.2$ and $x + y + z = 0.9$, without re-solving the problem, estimate what the new minimum and maximum will be.

13. Minimize $f(x, y, z) = x - y + z$ subject to $x^2 + y^2 + z^2 \leq 1$.

14. Minimize $f(x, y, z) = (x - 10)^2 + (y - 10)^2 + (z - 10)^2$ subject to $x^2 + y^2 + z^2 \leq 12$ and $-x - y + 2z \leq 0$. If $\vec{x}^*$ is the optimal point, which of the constraints (if either) are active at $\vec{x}^*$? Is $\vec{x}^*$ a *global* minimizer? Is the global minimum unique?

15. Minimize $f(x, y) = x^2 + 4y^2 - 4xy - x + 12y$ subject to $x + y \geq 4$ and $x \geq 0$, $y \geq 0$.

16. Maximize and Minimize $f(x, y) = x^2 - y^2$ subject to $(x-2)^2 + y^2 - 4 \leq 0$.

17. Sketch the region satisfying $x^2 + y^2 \leq 4$ and $x^2 - y^2 \leq 1$. Find the minimum and maximum of $f(x, y) = x^2 + 2y^2 + xy$ over this region.

18. Solve Problem 3. in Chapter 1.

19. Solve Problem 4. in Chapter 1.

20. Use *Mathematica* to solve the scaffolding problem: Problem 2. in Chapter 1.

21. Exercise 2.24, page 43.

22. Exercise 2.27, page 47.

23. Exercise 2.28, page 47.

24. Exercise 2.31, page 49.

25. Let $f : \mathbb{R}^n \to \mathbb{R}$. Define, for each $a \in \mathbb{R}$, $S(a) = \{\vec{x} \ : \ f(\vec{x}) \leq a\}$.

    (a) Prove that if f is a convex function then, for any a, $S(a)$ is a convex set.

    (b) Suppose that $S(a)$ is a convex set for all a. Need f be a convex function?

26. Consider the function $f(\vec{x}) = 2x^2 + 2xy + y^2 - yz + 2z^2$.

    (a) Is f a convex function?

    (b) Is f a strictly convex function?

27. Minimize $f(x, y) = -x - 2y + y^2$ subject to $x + y \leq 1$ and $x \geq 0$, $y \geq 0$. Is this a convex optimization problem? Explain.

28. Determine whether or not $f(x, y) = \dfrac{x^2}{y}$ is convex on $k = \{(x, y) \ : \ x > 0, \ y > 0\}$. First try Sylvester's Criterion and show that it fails to show that f is strictly convex; now show that it is convex.

29. Show that $f(x, y, z) = x^2 + y^2 + z^2 - x - y - z$ is convex. Is the feasible set $x^2 + y^2 = 4, \ -1 \leq z \leq 1$ convex? Find the minimum and maximum of f subject to these constraints.

30. (a) If Q is an $n \times n$ symmetric matrix and $\vec{v} \in \mathbb{R}^n$, define $g(\vec{x}) = \vec{x}^T Q \vec{x} + \vec{v}^T \vec{x}$. Find $Dg(\vec{x})$ and $D^2 g(\vec{x})$.

    (b) Let A be an $m \times n$ matrix, and $\vec{b} \in \mathbb{R}^m$. Define $f : \mathbb{R}^n \to \mathbb{R}$ by $f(\vec{x}) = \|A\vec{x} - \vec{b}\|^2$. Show that f is a convex function (on $\mathbb{R}^n$).

    (c) For this problem, A is an $n \times n$ matrix which is not necessarily symmetric.

        i. Show that A is positive definite if, and only if $A + A^T$ is positive definite.

ii. Formulate a test for checking that A is positive definite (other than $\vec{x}^T A\vec{x} > 0$ for all $\vec{x}$!).

iii. Use your test to determine whether or not the matrix $A = \begin{bmatrix} 2 & 2 & 3 \\ 0 & 1 & -4 \\ -3 & 3 & 2 \end{bmatrix}$ is positive definite.

Chapter 3

Approximate Optimization

3.1 Introduction

In practice, we cannot expect to know a lot about the function we are trying to minimize. For example, to determine a derivative at a point would likely require evaluation of the function at two places to enable the difference quotient to be used as an approximation. To estimate the gradient would require doing this n times. To approximate a *second* derivative requires (at least) three function evaluations, and to find the matrix of all second order derivatives would require doing this $\frac{1}{2}n(n+1)$ times. If we assume that it is, in some sense, *expensive* to perform function evaluations, then we seek schemes for finding minima with as few evaluations as possible. It is fairly clear that first derivative information is necessary so as to know which direction is at least an improvement, but we will try to avoid using second derivative information.

Our goal is to find a sequence of successive approximations $\vec{x}_k$ to the optimal location $\vec{x}^*$ such that $\|\vec{x}_k - \vec{x}^*\| \to 0$ as k increases. Better still, we would like $\|\vec{x}_{k+1} - \vec{x}^*\| \leq C\|\vec{x}_k - \vec{x}^*\|^\alpha$ such that $C, \alpha > 0$. The bigger α and the smaller C the better. In these notes we won't actually analyze such convergence issues.

DEFINITION 3.1. **Search direction:** At any stage we are at, say, $\vec{x}_k$. A "search direction" is a vector $\vec{d}_k$, so that our next position $\vec{x}_{k+1}$ satisfies $\vec{x}_{k+1} - \vec{x}_k$ is parallel to (a positive multiple of) $\vec{d}_k$.

Step-size: If we have chosen a search direction, we need to specify how far we should move in that direction. This is the "step-size" $t_k > 0$ so that $\vec{x}_{k+1} = \vec{x}_k + t_k\vec{d}_k$. I.e. $t_k = \|\vec{x}_{k+1} - \vec{x}_k\|/\|\vec{d}_k\|$.

Note: We'll only consider unconstrained problems here.

A first order necessary condition to be at a minimum is that $\nabla f(\vec{x}^*) = \vec{0}$; as a test to know when to stop, we typically try to satisfy $\|\nabla f(\vec{x}_k)\| < \varepsilon$ for a given tolerance $\varepsilon > 0$.

A second order necessary condition to be at a minimum is that $\vec{u}^T D^2 f(\vec{x}^*)\vec{u} \geq 0$ for all $\vec{u}$, where $D^2 f(\vec{x}^*)$ is the Hessian of f (B.7.6). That is, $D^2 f$ is positive semidefinite (B.5.1).

A second order sufficient condition: $\nabla f(\vec{x}^*) = \vec{0}$ and $D^2 f$ is positive definite (B.5.1).

3.2 Line Searches

For the moment we assume that from $\vec{x}_k$ the direction $\vec{d}_k$ has somehow already been determined and we are in the position of determining the best t_k; that is we want to minimize $h(t) = f(\vec{x}_k + t\vec{d}_k)$. Solving this one-dimensional problem is called a *line search* scheme.

What makes one search method better than another is if it uses fewer function evaluations to achieve the same degree of accuracy in finding the minimum of h. The first two methods search from an initial point t_0. Without further prior knowledge about properties of h, we cannot determine the number of steps it will take to achieve a desired accuracy. We then consider methods which assume knowledge of an *interval* within which it is assumed the minimum must be achieved. Of course, we must address the issue of finding this interval in the first place.

3.2.1 Fixed step size

Simply step Δt until $h(t_{i+1}) \geq h(t_i)$ and then reduce Δt, changing its sign, and continue: assuming we have $h(t_i) < h(t_{i-1})$, suppose we then find $h(t_{i+1}) \geq h(t_i)$. We now know that a (local) minimum must occur *between* t_{i-1} and t_{i+1} – it's possible that the minimum is between t_{i-1} and t_i *or* t_i and t_{i+1} (see Figure 3.1):

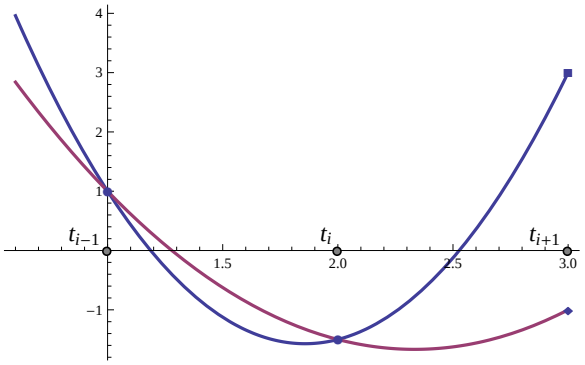


Figure 3.1: The minimum may occur between t_{i-1} and t_i, or between t_i and t_{i+1}.

So, at this point we "turn around" and step back in smaller steps; set Δt to $-\frac{1}{2}\Delta t$, say.

As an algorithm:

1. Choose "tolerance" ε. Choose t_0 and Δt. Set $k = 0$.
2. If $|h'(t_k)| \leq \varepsilon$, STOP.
3. Set $t_{k+1} = t_k + \Delta t$.
4. If $h(t_{k+1}) \geq h(t_k)$ then set $\Delta t \leftarrow -\frac{1}{2}\Delta t$.
5. $k \leftarrow k + 1$.
6. Go to 2.


Here is a way to encode this in *Mathematica*: (See the *Mathematica* notebook "Approx Fixed-Step.nb"

```
FixedStep[h_, t0_, dt_, tol_] :=
  Module[
    {FSList = {t0}, Dt = dt},
    While[Abs[h'[FSList[[-1]]]] > tol,
      AppendTo[FSList, FSList[[-1]]+ Dt];
      If[h[FSList[[-1]]] >= h[FSList[[-2]]], Dt = -Dt/2]
    ];
    FSList
  ]
```

Some explanation:

- `Module` is a way to combine several commands into a single routine. We have called the routine `FixedStep`. It takes as arguments: a function `h` (which has to be a *pure function* – see later); an initial starting point `t0`; an initial step-size `dt`; and a tolerance `tol`.
- When specifying arguments in defining a function or module like this, the underscores are necessary – this says that the values are "replaceable."
- The `:=` means "delayed evaluation" – *Mathematica* will not try to evaluate the function/module until it is called.
- A `Module` must consist of two components.

  The first is a list {`var1, var2, ...`} of *local variables*. Notice that these can be set to have initial values as here, though it is not necessary. It is also possible to have an empty list { }, but the list must be there. Following this list is a comma.

Following this is any number of commands separated by semi-colons. The *final* command is what is the *output* (or "return") of the module. If, as here, there is no semi-colon, then the module returns that last value. If there were a semi-colon, it would return nothing (though it might be doing some useful calculations none-the-less).

Remark: Why do we need to use `Dt` when we have already passed `dt`? The answer is that `dt` is only an *initial* Δt, and it needs to change. You are not allowed to change the value of an argument passed to a module/function. So, we set up a new variable, internal to the module which can be changed, and we initialize it to be `dt`.

- The guts of the module is a `While` loop – this says, while the derivative is too large, keep looping through the commands. `FSList` stands for "Fixed Step List" and will be the list of points t_i of the algorithm. Calling `FSList[[-1]]` grabs the *last* component of `FSList`.
- `AppendTo[list, value]` takes `list` and appends `value` to the end, the result being `list` one element longer than it was.
- `If[condition, do first, else do second]` does: if the `condition` is true, perform the first action, otherwise perform the second action. The second action doesn't have to be there (as in our case).
- The final line simply returns `FSList`.

What's this about **pure functions**? Usually one can define functions, like $h(x) = x^2$ simply by `h[x_] = x^2`. But now *Mathematica* needs to know that it is `x` that it should differentiate with respect to (for example). Better is to define the *pure function* `h = Function[#^2]`. The # plays the role of the variable, but it is not specified what it has to be. You should try to get into the habit of programming this way.

3.2.2 Newton's method

At t_i find the quadratic $q(t)$ which agrees with h at t_i to second order and set t_{i+1} to be the minimizer of q:

$$t_{i+1} = t_i - \frac{h'(t_i)}{h''(t_i)}.$$

Note: If the objective function has a zero tangent somewhere, Newton's method will likely get stuck there, even if it is not a minimum. In coding in *Mathematica*, it's advisable to compute *numerically* (`N[···]`); otherwise, the routine can run very slowly as *Mathematica* works to compute things exactly.

Newton's method can be shown to converge very quickly if the initial point is close enough to the location of the true minimum. However, this is not something that can necessarily be

assured, and when this is not the case, Newton's method can fail dramatically. Furthermore, we see that it is necessary to compute *second* derivatives of h, something that is computationally expensive.

EXERCISE 3.2. Write *Mathematica* code to implement this one-dimensional Newton's method.

3.2.3 Golden Section method.

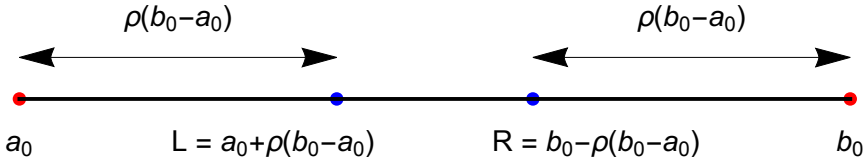
We assume here that that h has only one local minimum on the interval $[a_0, b_0]$ – in fact we assume more: we assume that h is *unimodal* on the interval. That means there is some point $a_0 \leq t^* \leq b_0$ so that h is monotonically decreasing for $a_0 < t < t^*$ and monotonically increasing for $t^* < t < b_0$ (if h is continuous, then having only one local minimum implies unimodal). We want to successively cut the interval into smaller and smaller pieces, guaranteeing at each stage that t^* is in the new smaller interval, but we want to do so with as few function evaluations as possible (think in terms of both speed and cost).

We will choose $\rho > 0$, without loss of generality $\rho < \dfrac{1}{2}$, and divide the current interval into pieces of proportion $\rho : 1 - \rho$.

It is important to see why we don't choose $\rho = \dfrac{1}{2}$. If we divide the interval in half and evaluate h at the midpoint, we have no way of knowing whether t^* lies in the left- or the right-hand subinterval. If we divide the interval into three pieces, and evaluate h at each of the two interior points, we *can* determine whether t^* lies in the union of the first two, or the union of the second two.

Whatever ρ will be, we obtain two new internal points L and R, say, by setting

$$L = a_0 + \rho(b_0 - a_0), \qquad R = b_0 - \rho(b_0 - a_0).$$



With knowledge of $h(a_0)$, $h(L)$, $h(R)$ and $h(b_0)$ we will set our *new interval* $[a_1, b_1]$ to be either $[a_0, R]$ (if $h(L) < h(R)$) or $[L, b_0]$ (if $h(L) > h(R)$).

Suppose (without loss of generality) that the new interval is $[a_1, b_1] = [a_0, R]$; we will then repeat the process on this new interval: define two new internal points, say

$$L' = a_0 + \rho(R - a_0), \qquad R' = b_1 - \rho(R - a_0).$$

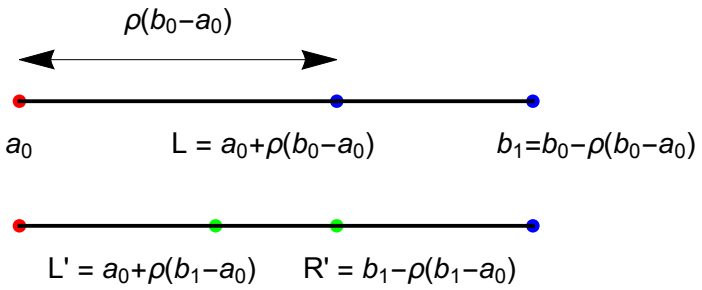
To minimize the number of function evaluations, when we determine L' and R' we make one of them coincide with either L or R, depending on which one we know is in the region where the minimum must lie. It is this condition which will determine ρ. Note that we have already determined the values of h at a_1 and b_1. This process guarantees that each sub-division requires only *one* more function evaluation, even though we are always involving *two* interior points. (If we didn't insist on this efficiency of evaluation, we could do better in terms of shrinking the intervals by taking $\rho = \frac{1}{2} - \varepsilon$ for very small ε.)

So, in our example where $h(L) < h(R)$, we know that $t^* \in [a_0, R]$; we want to choose our next interval so that one end of it coincides with L (which we are currently not using). I.e. we want ρ so that either $L' = L$ or $R' = L$. We want

$$L' := a_0 + \rho(R - a_0) = L, \qquad \text{or} \qquad R' := R - \rho(R - a_0) = L.$$

Notice here the *length* of the interval we currently have is $R - a_0$, not $R - L$ (do you see why?).

EXERCISE 3.3. As you would expect, solving $L' := a_0 + \rho(R - a_0) = L$ yields $\rho = 0$.



Using the definitions of L and R in $R - \rho(R - a_0) = L$ we have

$$b_0 - \rho(b_0 - a_0) - \rho\big(b_0 - \rho(b_0 - a_0) - a_0\big) = a_0 + \rho(b_0 - a_0)$$

which simplifies to

$$\rho^2(b_0 - a_0) - 3\rho(b_0 - a_0) + (b_0 - a_0) = 0$$

and since $b_0 - a_0 \neq 0$,

$$\rho^2 - 3\rho + 1 = 0 \qquad \Rightarrow \qquad \rho = \frac{3 - \sqrt{5}}{2} \qquad \text{using } \rho < \frac{1}{2}.$$

So we've determined ρ. Now we can define the *next new interval*: it will be either $[a_2, b_2] = [a_1, R']$ (if $h(L') < h(R')$), or $[a_2, b_2] = [L', b_1]$ (if $h(L') > h(R')$). Our choice guarantees that t^* lies within our new interval, and the point not used (L' in the first case, R' in the second) lies in the new interval and we already know the value of h there, so on the next iteration we will not need extra function evaluations.

Using this Golden Section method reduces the size of the interval by $1-\rho \approx 0.618$ at each iteration/each function evaluation. If we start with $[a_0, b_0]$ and we want the final interval to have length less than δ, say, then we need n so that

$$b_n - a_n = (1-\rho)^n (b_0 - a_0) < \delta, \qquad \text{that is,} \qquad n \geq \frac{\ln\big(\delta/(b_0-a_0)\big)}{\ln(1-\rho)}. \tag{3.1}$$

Why is this called the Golden Section method? The Golden ratio is the ratio of $\varphi = \dfrac{1+\sqrt{5}}{2}$ to 1. Our ρ is such that the ratio $1 \ : \ (1-\rho)$ is the same as $\varphi \ : \ 1$ (i.e. $\varphi = 1/(1-\rho)$). That is, the ratio of the old interval to the new interval is φ.

Efficiency: What, then, is t^*, or our approximation $\tilde{t}$ to t^*, and how accurate it is? All we know is that it lies in the final interval, $[a_n, b_n]$. Within this interval, we actually know the value of h at one point; it is either $a_n + \rho(b_n - a_n)$ (if $b_n = b_{n-1}$) or $b_n - \rho(b_n - a_n)$ (if $a_n = a_{n-1}$.). Our best choice for our approximate $\tilde{t}$ is then this internal value; we are not making an additional function evaluation, and we can be sure that the true value of t^* is within $(1-\rho)(b_n - a_n)$ of $\tilde{t}$. Thus,

$$|\tilde{t} - t^*| \leq (1-\rho)(b_n - a_n) = (1-\rho)^{n+1}(b_0 - a_0).$$

So, if our goal is to find t^* to within accuracy of ε, we should choose n so that

$$(1-\rho)^{n+1}(b_0 - a_0) < \varepsilon, \qquad \text{that is,} \qquad n \geq \frac{\ln\big(\varepsilon/(b_0-a_0)\big)}{\ln(1-\rho)} - 1. \tag{3.2}$$

Compare this to (3.1) above.

As an algorithm:

1. Set $\rho = 1 - 0.618 = 0.382$. Choose tolerance ε. Choose $a_0 < b_0$.
2. Compute the number of steps needed: $N = \left\lceil \dfrac{\ln\big(\varepsilon/(b_0-a_0)\big)}{\ln(1-\rho)} - 1 \right\rceil$;
   $k \leftarrow 1$.
3. If $k > N$, go to 7.
4. Interior points: $I_L = a_k + \rho(b_k - a_k)$, $I_R = b_k - \rho(b_k - a_k)$.
   If $h(I_L) < h(I_R)$ then $a_{k+1} \leftarrow a_k$ and $b_{k+1} \leftarrow I_R$
   Else $a_{k+1} \leftarrow I_L$ and $b_{k+1} \leftarrow b_k$.
5. $k \leftarrow k+1$.
6. Go to 3.
7. Return the last interior point:
   If $h(I_L) < h(I_R)$ then $\tilde{t} = I_L$
   Else $\tilde{t} = I_R$.

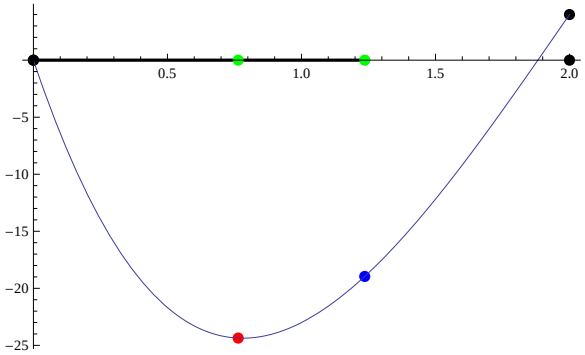
Notes: in this algorithm we don't specifically make use of the fact that we don't have to re-evaluate h at *both* the interior points; in practice, we would want to record the evaluation to be available for the next iteration. Similarly, we actually know that one of the next a_{k+1}, b_{k+1} will coincide with one of the a_k, b_k, but in the above we simply compute it once again.

EXAMPLE 3.4. Minimize $f(t) = t^4 - 14t^3 + 60t^2 - 70t$ in $[0, 2]$ finding t^* to within 0.3. Using (3.2) above, we need n so that $(0.618)^{n+1}|I_1| \leq 0.3$, and find $n = 3$. Following the idea of the algorithm above:

- With $a_0 = 0,\ b_0 = 2$ we obtain

$$f(a_0 + \rho(b_0 - a_0)) = f(0.7639) = -24.36$$
$$f(b_0 - \rho(b_0 - a_0)) = f(1.236) = -18.96$$

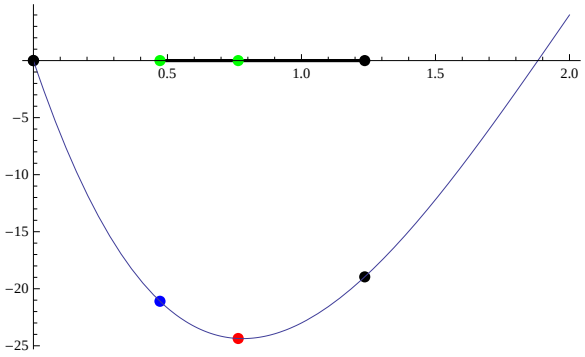
so our new interval is $[a_1, b_1] = [a_0, b_0 - \rho(b_0 - a_0)] = [0, 1.236]$;



- Now

$$f(a_1 + \rho(b_1 - a_1)) = f(0.4721) = -21.1$$
$$f(b_1 - \rho(b_1 - a_1)) = f(0.7639) = -24.36 \text{ (which we have for free)}$$

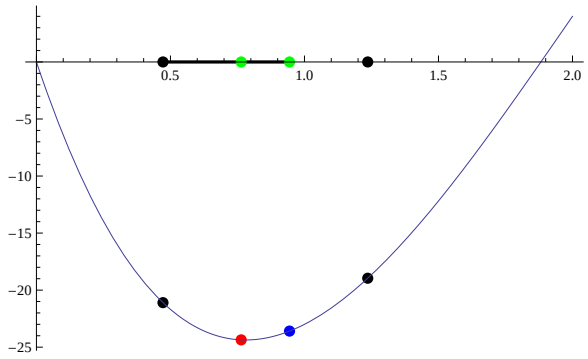
and so our next interval is $[a_2, b_2] = [0.4721, 1.236]$, $(b_2 = b_1)$;



- Next

$$f(a_2 + \rho(b_2 - a_2)) = f(0.7639) = -24.36 \text{ (which we have for free)}$$
$$f(b_2 - \rho(b_2 - a_2)) = f(0.9443) = -23.59$$

and so our next interval is $[a_3, b_3] = [0.4721, 0.9443]$ ($a_3 = a_2$);



- At this point we know that the true minimum is in the interval $[0.4721, 0.9443]$. We should take as our best estimate for t^* the point $\tilde{t} = 0.7639$ (for which we know the value of f). We have

$$|\tilde{t} - t^*| < 0.7639 - 0.4721 = 0.2918 < 0.3$$

as required.

□

EXERCISE 3.5. Write *Mathematica* code to implement the Golden Section method. Here is some skeleton code to get you started (you need to complete the code, in particular where there is "…"). (See the *Mathematics* notebook "Approx GoldenSection.nb")

```
GoldenSection[h_, p0_, tol_] :=
  Module[
    {GSList = {N[p0]}, ρ = N[...], IntPts = p0, step,
      best, NumSteps},
    NumSteps = ...;
    For[step = 1, step <= NumSteps,
      IntPts = ...; (calculate internal points)
      If[h[IntPts[[1]]] < h[IntPts[[2]]],
        (* minimum is in the left-hand part *)
        AppendTo[GSList, ...],
        (* otherwise minimum is in the right-hand part *)
        AppendTo[GSList, ...]
      ];
      step++;
    ];
    best = ...; (calculate best approximation)
    {GSList, best}
  ];
```

Remarks:

- We set `GSList = N[p0]` as *numerical* to force *Mathematica* to always compute numerically rather than symbolically. `GSList` is the list of pairs of endpoints of the successive intervals.

- The last pair of points found is the last element of `GList`, i.e. `GList[[-1]]`. The components of this last element are `GSList[[-1,1]]` and `GSList[[-1,2]]`.

- `IntPts` are the two new *interior points* which we use to determine the next interval. We initialize `IntPts = p0` for the following reason: suppose that the initial interval was *shorter* than the requested tolerance `tol`. Then `NumSteps` will be less than 1 and the `For[ ]` loop will not be visited. However, the declaration of `best` at the end of the algorithm will be in terms of the last values of `IntPts`. With this initialization of `IntPts`, the algorithm will return the initial interval if it is shorter than the tolerance, and the initial endpoint at which h is smaller as the `best` approximation.

- Comments can be included in *Mathematica* code by surrounding them by `(*` and `*)`, e.g. `(* comment *)`.

- What is returned is both the list of intervals and the best approximate $\tilde{t}$.

3.2.4 Fibonacci search method.

We proceed as in the Golden Section method but allow $\rho = \rho_j$ to change at each step. We will see that by choosing ρ_j judiciously it is possible to out-perform the Golden Section method. In the Golden Section derivation, we found an equation ρ must satisfy, based on knowing the locations within the previous interval of the internal points. Now, to determine ρ_{k+1}, we won't know the locations of the internal points of the previous interval explicitly, but we will know them *in terms of* ρ_k.

If at step k the interval has (without loss of generality) length 1 and $a_{k-1} = 0,\ b_{k-1} = 1$, we have $a_k = \rho_k$ and $b_k = 1 - \rho_k$. If we assume (again, without loss of generality) that the minimum lies in $[0, b_k]$, we need

$$\begin{aligned} b_{k+1} := (1-\rho_{k+1})b_k = (1-\rho_{k+1})(1-\rho_k) = a_k := \rho_k \\ \Rightarrow \qquad \rho_{k+1} = \frac{\rho_k}{\rho_k - 1} + 1 = 1 - \frac{\rho_k}{1-\rho_k}. \end{aligned}$$

This doesn't pin the sequence of ρ_k down (indeed, setting them all equal to the Golden Section ratio will work). After n iterations of this, the interval shrinks by a factor of $(1-\rho_n)\cdots(1-\rho_1)$. What we want is to *minimize* this, subject to the constraints that $\rho_k \geq 0$ and the above recurrence

relation holds. One sequence which yields a solution to this comes from the Fibonacci sequence enumerated as $F_{-1} = 0,\ F_0 = 1$, and $F_{k+1} = F_k + F_{k-1}$. The ρ_k turn out to be given by

$$\rho_1 = 1 - \frac{F_{n+1}}{F_{n+2}}, \quad \cdots \quad \rho_k = 1 - \frac{F_{n-k+2}}{F_{n-k+3}}, \quad \cdots \quad \rho_n = 1 - \frac{F_2}{F_3} = \frac{1}{3}.$$

In fact, this sequence is the unique minimizer of the constrained optimization problem.

To calculate by how much the interval size is reduced, we have to observe that the n^{th} interval $(b_n - a_n)$ is $(1 - \rho_n)$ times the length of the previous one, and so on, to obtain:

$$b_n - a_n = (1 - \rho_n)(1 - \rho_{n-1}) \cdots (1 - \rho_2)(1 - \rho_1)(b_0 - a_0) = \frac{2}{F_{n+2}}(b_0 - a_0).$$

In order to apply this algorithm, we have to decide *a priori* how many steps we are going to perform because ρ_1 is given in terms of the *last* Fibonacci number(s). Really what we have done is solved: given that we want to obtain a certain degree of accuracy, what is the minimal number of evaluations we can make, and how should we make them? To obtain a final interval of length less than δ, one first must find n so that $\frac{2}{F_{n+2}}L < \delta$ (where L is the length of the initial interval). This is the equivalent to (3.1) in the Golden Section method.

Efficiency: Within the final interval, we know the value of the function at the midpoint (because $\rho_n = 1/3$). Choosing the approximate value $\tilde{t}$ to be this midpoint, we obtain

$$|\tilde{t} - t^*| \leq \frac{1}{2}(b_n - a_n) = \frac{1}{F_{n+2}}(b_0 - a_0). \tag{3.3}$$

This is the equivalent of (3.2) in the Golden Section method.

EXERCISE 3.6. Prove that, for all n, $\frac{1}{F_{n+2}} < (1 - \rho)^{n+1}$.

EXAMPLE 3.7. Apply this method to: Minimize $f(x) = x^4 - 14x^3 + 60x^2 - 70x$ in $[0, 2]$ finding x^* to within 0.3. By (3.3), one finds that, again, $n = 3$ is needed.

- $\rho_1 = 1 - \frac{F_4}{F_5} = 1 - \frac{5}{8} = \frac{3}{8} = 0.375$;

  $$f(a_0 + \rho_1(b_0 - a_0)) = f(0.75) = -24.34$$
  $$f(b_0 - \rho_1(b_0 - a_0)) = f(1.25) = -18.65.$$

  so our second interval is $[a_1, b_1] = [a_0, b_0 - \rho(b_0 - a_0)] = [0, 1.25]$;

- $\rho_2 = 1 - \frac{3}{5} = 0.4$,

  $$f(a_1 + \rho_2(b_1 - a_1)) = f(0.5) = -21.69$$
  $$f(b_1 - \rho_2(b_1 - a_1)) = f(0.75) = -24.34 \text{ (which we have for free)}$$

  and so our next interval is $[a_2, b_2] = [0.5, 1.25]$;

- $\rho_3 = 1 - \dfrac{2}{3} = 0.33$,

  $$f(a_2 + \rho_3(b_2 - a_2)) = f(0.75) = -24.30 \text{ (which we have for free)}$$
  $$f(b_2 - \rho_3(b_2 - a_2)) = f(1.0) = -23$$

  and so our next interval is $[a_3, b_3] = [0.5, 1.0]$.

- At this point, we already know the value of f at the midpoint 0.75, and we take $\tilde{x} = 0.75$. We are guaranteed that $|\tilde{x} - x^*| < \dfrac{1}{2}(b_3 - a_3) = 0.25 < 0.3$ as expected. It has done a little better than the Golden Section method.

□

Important Remark: Using either of the Golden Section or Fibonacci methods requires an initial "bracket" within which you know (or expect) the solution to lie. One usually knows one end of this bracket since we know we are looking in a direction of descent. However, the other end is not known and in general we have to use a heuristic "rule of thumb" procedure to obtain this initial bracket. For example: suppose we are at position $\vec{x}_k$ and we have a descent direction $\vec{d}_k$; define $h(t) = f(\vec{x}_k + t\vec{d}_k)$. We could evaluate h at $t = \mu_0 = 0$, $t = \mu_1 = \alpha$, $t = \mu_2 = 2\alpha$, $t = \mu_3 = 4\alpha$,..., $t = \mu_k = 2^{k-1}\alpha$, with some α sufficiently small. At each step we check to see if the value of h is still decreasing. Once we see it have an increase $h(\mu_{j+1}) \geq h(\mu_j)$, we can use $\left[\mu_{j-1}, \mu_{j+1}\right]$ as our initial bracket, and then apply one of these line search methods. In choosing how small to take α, it is prudent to take into account the size of $\|\vec{d}_k\|$. We need $\alpha\|\vec{d}_k\|$ to be small, so something to try might be:

$$\alpha\|\vec{d}_k\| \leq \frac{1}{100} \qquad \Rightarrow \qquad \alpha = \frac{1}{100\|\vec{d}_k\|}.$$

The value 100 in the denominator is something that might be changed in a given situation if we believe we should be more (or less) careful.

Some *Mathematica* code which implements the above algorithm is (see the *Mathematica* notebook "Approx InitInt.nb")

```
InitInt[h_, t0_, α_] :=
  Module[{k = 0, L = {t0}, d, t, αloc = α},
    (* ensure a descent direction *)
    d = -Sign[D[h[t],t]/.t -> t0];
    (* ensure sufficiently small step *)
    While[h[t0+αloc] >= h[t0], αloc = αloc/2];
    While[h[L[[-1]]] > h[t0+2^k αloc d ],
      AppendTo[L, t0+2^k αloc d ];
      k++ ];
    AppendTo[L, t0+2^k αloc d ];
    Sort[{L[[-3]],L[[-1]]}]
  ]
```

Notes:

- The function `h` must be passed as a *pure function*;
- `t0` is the initial point, and α is a parameter controlling how fine the search for an initial interval should be;
- The reason for evaluating `d = -Sign[D[h[t],t]/.t -> t0]` is to ensure we move from `t0` in a direction where `h` decreases. Note that if `(D[h[t],t]/.t -> t0) = 0` then the algorithm will fail. In certain applications, it may not be feasible to compute $h'(t)$ (the Golden Section and Fibonacci methods don't use derivative information at all themselves) in which case this line of code should be removed and the algorithm calling `InitInt[ ]` must be robust enough to ensure the search direction (with increasing t) is indeed a descent direction;
- A problem can occur if the α passed is such that $h(t_0 + \alpha) > h(t_0)$, i.e. even for the smallest step size, the function goes *up*. We should first make sure that α is sufficiently small that we achieve a decrease. Thus, we define a local version of α, `αloc` and the first `While[ ]` loop repeatedly halves `αloc` until it is sufficiently small. In fact, the result of this is now an algorithm which either searches to the left of $t_0 + \alpha$ by increments of $2^{-j}\alpha$ or to the right of $t_0 + \alpha$ by increments of $2^j\alpha$, depending on whether h has increased or decreased from $h(t_0)$.
- `L` consists of the list of places where we evaluate h.
- `k++` is equivalent to `k = k+1`;
- The final `Sort` command is necessary since if we were moving "to the left" then the final interval would otherwise be in the wrong order.

3.3 Gradient Methods

3.3.1 Introduction

We've discussed how to decide how far to step assuming the direction is already determined; now we address the harder problem of determining in what direction to move.

Recall the gradient of a function $\mathbb{R}^n \to \mathbb{R}$: $\nabla f(\vec{x})$ is a vector orthogonal to the plane tangent to the level set of f passing through $(\vec{x}, f(\vec{x}))$; thus, it is easy to verify, $\nabla f(\vec{x})$ points in the direction in which f increases the most rapidly from $\vec{x}$.

DEFINITION 3.8. A **descent direction** for f at $\vec{x}$ is a $\vec{d}$ such that $\nabla f(\vec{x}) \cdot \vec{d} < 0$.

3.3.2 Steepest Descent Method

From an approximate minimum location $\vec{x}_k$ choose the direction $\vec{d_k} = -\nabla f(\vec{x}_k)$; if $\|\vec{d_k}\| \leq \varepsilon$ for some pre-assigned threshold ε then stop and $\vec{x}^* \approx \vec{x}_k$ (for at a local minimum $\vec{x}^*$, $\nabla f(\vec{x}^*) = \vec{0}$). Otherwise perform a *line search* to find

$$t_k = \underset{t \geq 0}{\operatorname{argmin}} f(\vec{x}_k + t\vec{d_k})$$

and set $\vec{x}_{k+1} = \vec{x}_k + t_k\vec{d_k}$.

Note: "$\underset{t\geq 0}{\operatorname{argmin}}$" means the *argument* (i.e. the t) which *achieves* the minimum (rather than the minimum value itself).

The sequence of approximations has an interesting, though not desirable, pattern:

PROPOSITION 3.9. *If $\{\vec{x}_k\}$ is a steepest descent sequence then $\vec{x}_{k+1} - \vec{x}_k$ is orthogonal to $\vec{x}_{k+2} - \vec{x}_{k+1}$.*

Proof. By definition,

$$\langle \vec{x}_{k+1} - \vec{x}_k, \vec{x}_{k+2} - \vec{x}_{k+1} \rangle = t_k t_{k+1} \langle \nabla f(\vec{x}_k), \nabla f(\vec{x}_{k+1}) \rangle$$

so we must prove $\langle \nabla f(\vec{x}_k), \nabla f(\vec{x}_{k+1}) \rangle = 0$. If $h(t) = f(\vec{x}_k - t\nabla f(\vec{x}_k))$ then

$$0 = h'(t_k) = \nabla f(\vec{x}_k - t_k \nabla f(\vec{x}_k))^T (-\nabla f(\vec{x}_k)) = -\langle \nabla f(\vec{x}_{k+1}), \nabla f(\vec{x}_k) \rangle.$$

□

EXAMPLE 3.10. Consider $f(x, y) = \frac{3}{2}x^2 + 2xy + 2y^2 - 4x - 2y$, with $\vec{x}_0 = [-3.5, 2]^T$. It takes 15 iterations to achieve $\|\nabla f\| < 0.01$ and we obtain the sequence of points shown in Figure 3.2. This zig-zag path can result in a very inefficient search. Consider the following example: $f(x, y) = \frac{1}{2}x^2 + 2xy + 4y^2 - 4x - 2y$, with $\vec{x}_0 = [-3.5, 2]^T$. This time it takes 60 iterations to achieve $\|\nabla f\| < 0.01$; we obtain Figure 3.3. □

EXAMPLE 3.11. A numerical example: $f(x, y, z) = (x - 4)^4 + (y - 3)^2 + 4(z + 5)^4$ (we can see that the minimum is at $[4, 3, -5]^T$), $\vec{x}_0 = [4, 2, -1]^T$.

We find $\vec{d_0} = -\nabla f(\vec{x}_0) = -[0, -2, 1024]^T$.

To compute $\vec{x}_1$ we need $t_0 = \underset{t\geq 0}{\operatorname{argmin}} f(\vec{x}_0 + t\vec{d_0})$. Set $h(t) = f(\vec{x}_0 + t\vec{d_0})$; to implement the Golden Section line search, we first must find an initial bracket and so evaluate (with $\alpha = 0.001$), $h(0)$, $h(\alpha)$, $h(2\alpha)$, $h(4\alpha)$,.... We find $h(0.002) = 59.1$, $h(0.004) = 0.98$, $h(0.008) = 1236$, so our initial bracket is $[0.002, 0.008]$. Figure 3.4 shows the slice of f which is the

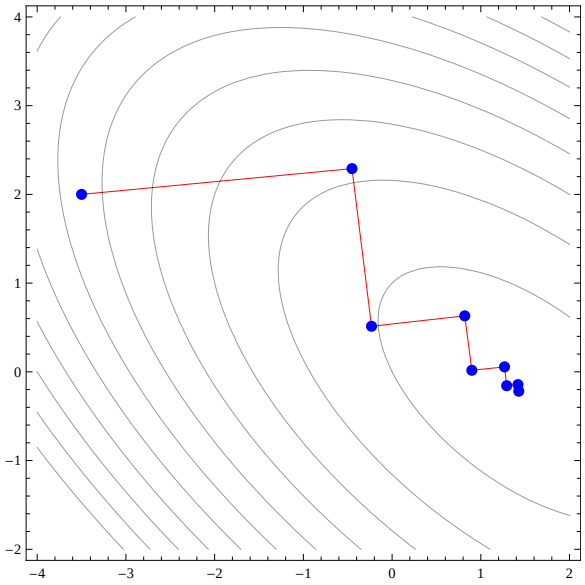


Figure 3.2: The zig-zag path of steepest descent.

function $h(t)$ together with the places we evaluate h in determining this bracket. Implementing the Golden Section method we then find $t_0 = 0.00398$ and so

$$\vec{x}_1 = \vec{x}_0 + t_0\vec{d}_0 = [4, 2, -1]^T + 0.00398[0, 2, -1024]^T = [4, 2.0079, -5.0755]^T.$$

Next, we find $\vec{d}_1 = -\nabla f(\vec{x}_1) = -[0, -1.984, -0.00689]^T$, and $\|\nabla f(\vec{x}_1)\| = 1.984$. To compute $\vec{x}_2$ we need $t_1 = \underset{t\geq 0}{\operatorname{argmin}} f(\vec{x}_1 + t\vec{d}_1)$. Figure 3.5 shows the new slice of f and the process of determining the initial bracket for a line search. Implementing the Golden Section line search, we find $t_1 = 0.49996$ and so

$$\begin{aligned}\vec{x}_2 &= \vec{x}_1 + t_1\vec{d}_1 = [4, 2.0079, -5.0755]^T + 0.49996[0, 1.984, 0.00689]^T \\ &= [4, 2.99992, -5.07206]^T.\end{aligned}$$

We find $\vec{d}_2 = -\nabla f(\vec{x}_2) = -[0, -0.00036, -0.006]^T$, and $\|\nabla f(\vec{x}_2)\| = .005996$. Depending on how accurate we want to be, we could stop or continue. □

EXERCISE 3.12. Prove that if $\{\vec{x}_k\}$ is a steepest descent sequence and $\nabla f(\vec{x}_k) \neq \vec{0}$ then $f(\vec{x}_{k+1}) < f(\vec{x}_k)$.

EXERCISE 3.13. Let f be a quadratic function of the form

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

with Q symmetric and positive definite.

(a) Compute $\nabla f(\vec{x})$.

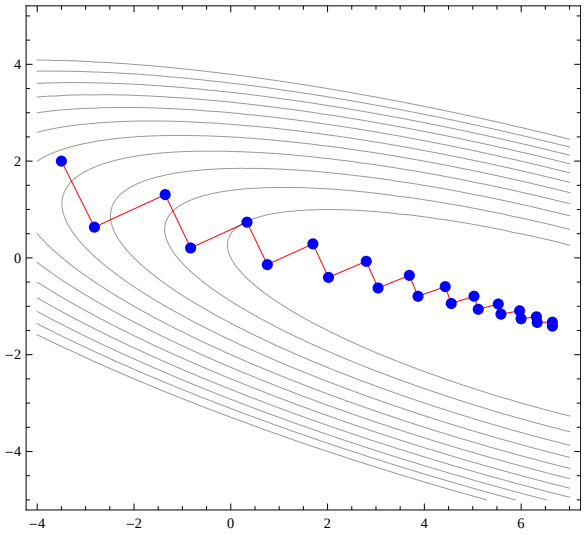


Figure 3.3: The zig-zag path can be very inefficient.

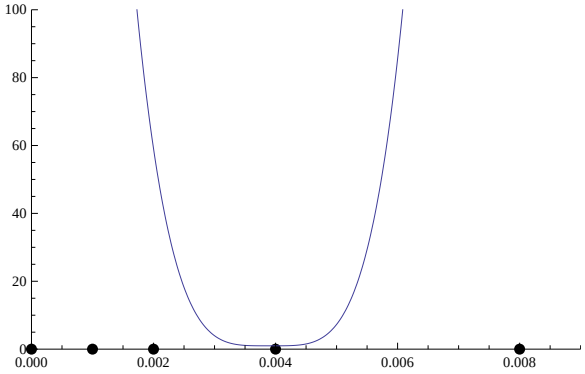


Figure 3.4: Determination of the initial bracket for the first line search.

(b) Find the (unique) minimizer of this function (analytically, not by way of an iterative algorithm).

(c) Defining $\alpha_k(t) = f(\vec{x}_k - t\nabla f(\vec{x}_k))$, find $t_k = \underset{t\geq 0}{\operatorname{argmin}} f(\vec{x}_k + t\vec{d}_k)$ by solving (analytically) $\alpha_k'(t) = 0$.

(d) Thus find the explicit formula for $\vec{x}_{k+1}$ in this case.

In *Mathematica*, functions of, say, two variables are declared with the argument being what is called a *sequence*. For example

```
f = Function[{x, y}, x^2 + y^2];
```

results in the function `f[x, y]`. The *input* to this function is the *sequence* `x, y`. This can be annoying when we are dealing with *points* in $\mathbb{R}^2$, namely `{x, y}`. The way `f` is defined,

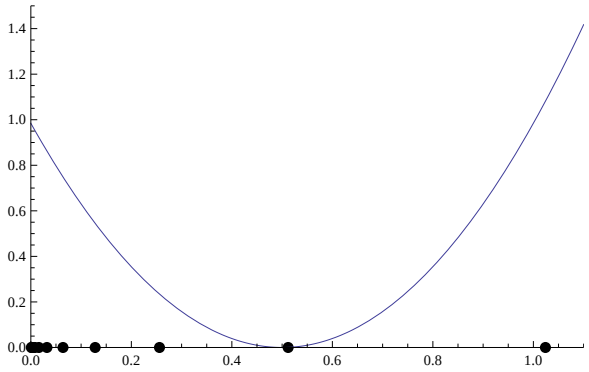


Figure 3.5: Determination of the initial bracket for the second line search.

`f[{x, y}]` produces an error. A work-around to this is to define our own function-defining command which results in a function which takes *vectors* as input instead of sequences. So, we define

```
VFunction[vars_, f_] := Function[vars, f][Apply[Sequence, #]]&
```

What this does is define a function using *Mathematica*'s `Function` command, but, when passed a vector, first converts the vector to a sequence so that the function can accept it. I suggest that you include and execute this command at the beginning of your *Mathematica* files and from then on define functions as `VFunctions`. As an example, if we define

```
f = VFunction[{x, y}, x^2 + y^2];
```

then `f[{a, b}]` results in the output `a^2+b^2` as expected. (Now, `f[a, b]` will produce an error.)

It is also clear that we will need commands to compute gradients, and perhaps Hessians, of functions. *Mathematica* can do this for us, but because gradients are dependent on choice of coordinates used and *Mathematica* wants to accommodate this generality, it is more cumbersome than it needs to be for us. So, here is some code you can use to compute gradients (see the *Mathematica* notebook "Approx Utilities.nb")

```
Grad2D[f_] := VFunction[{x, y},
  {D[f[{x, y}], x], D[f[{x, y}], y]}];
Hess2D[f_] := VFunction[{x, y},
  {{D[f[{x, y}], x, x], D[f[{x, y}], x, y]},
    {D[f[{x, y}], y, x], D[f[{x, y}], y, y]}}];
```

These commands take pure functions which take *vectors* as their arguments. In other words, you should use the above `VFunction` to define the functions which get fed to `Grad2D` and

`Hess2D`. You can extend these to define `Grad3D` and `Hess3D` (etc.) in the obvious way. The *Mathematica* command `D[ ]` takes derivatives of the passed function with respect to the specified variables.

EXERCISE 3.14. Implement (and understand) the following *Mathematica* code for the steepest descent algorithm. We assume that you have already encoded:

- `GoldenSection[ h, p0, tol]` (see page 65),
- `InitInt[ h, t0, α]` (page 68),
- `VFucntion[vars, f]` (page 73),
- `Grad2D[f]` (page 73).

In the below, `x0` is an initial point (a 2D vector), `tol` is a desired accuracy, and `LStol` is the accuracy to be used in the line search – it is the accuracy *relative to* the length of the initial interval. (See the *Mathematica* notebook "Approx SteepestDescent.nb")

```
SteepestDescent2D[f_, x0_, tol_, LStol_] :=
  Module[
    {x, y, a0, b0, XList = {x0}, df, h, dDirs, count = 0},
    df = Grad2D[f];
    dDirs = {-df[x0]};
    While[Norm[Last[dDirs]] > tol && count < 4000,
      h = Function[t, f[Last[XList] + t*Last[dDirs]]];
      {a0, b0} = InitInt[h,0,1/(100Norm[Last[dDirs]])];
      AppendTo[XList,
        Last[XList] +
          GoldenSection[h, {a0, b0}, LStol(b0 - a0)][[2]]*
                                                  Last[dDirs]];
      AppendTo[dDirs, -df[Last[XList]]];
      count++
    ];
    If[count >= 4000,
      Print[``Algorithm did not converge from'', x0,
        `` within 4000 steps; aborting.''];
      AppendTo[XList, {``fail''}]
    ];
    {XList, {XList[[-1]], count}}
  ];
```

Notes:

1. `df` is a local version of `Grad2D[f]`. It simply facilitates a shorter version of `Grad2D[f]`;

2. `XList` is the list of points found;

3. `dDirs` is the list of descent directions; there is no need to keep track of the whole list. We could replace `dDirs = {-df[x0]}` by `dDir = -df[x0]`, every occurrence of `Last[dDirs]` by `dDir`, and instead of using `AppendTo[ ]` to update, simply set `dDir = ...;`

4. `h` is the function of *one* variable to which we will apply the line search algorithm;

5. `{a0, b0}` is set to be the initial interval to use in `GoldenSection[ ]`; the "α" to use is chosen to be $1/(100\|\nabla f(\vec{x}_k)\|)$ as discussed at the end of Section 3.2 above;

6. Included is a "catch" in case the algorithm is failing to converge. This is a good feature to include. At each iteration, `count` is incremented. The test in the `While[ ]` loop uses `&& count<= 4000`; the symbol `&&` means "logical and." If `count` gets greater than 4000 (chosen for no particular reason) then the algorithm is forced to terminate and a message is printed. The last point in `XList` is set to be {``fail''}.

7. If we call the results `res` then `res[[1]]` consists of the full list of points found; `res[[2[[` consists of two pieces of information: the final point found (or {``fail''}), and the number of iterations it took, `count`. This is valuable in comparing algorithms.

Test the code on the function $f(x, y) = (x-1)^2 + (y-2)^2$ with `tol` set to 0.0001. Investigate the effect of setting `LStol` to values of 0.1, 0.01 and 0.001.

Try implementing `SteepestDescent2D[ ]` on the 2D functions in Section 3.5. A good way to test your algorithm is to first create a grid of start points within the rectangle $[x_1, x_2] \times [y_1, y_2]$:

```
StartPoints =
   Flatten[Table[{x, y}, {x, x1, x2}, {y, y1, y2}], 1];
```

(The `Flatten[ ]` command reduces the three dimensional table to a two dimensional one.) Then execute

```
Map[SteepestDescent2D[f, #, 10^(-3), 10^(-4)][[2]]&,
   StartPoints]
```

Here, `f` is the function, `#` iteratively gets replaced by the elements of `StartPoints` (this is what `Map[ ]` does), `10^(-3)` is the accuracy goal, and `10^(-4)` is the accuracy to use in the line search. Calling the `[[2]]` component returns only the *last* point from the algorithm and the number of steps it took. You will find that the number of steps it takes to converge (if it does) varies with function, and with starting point. You will also find that for some functions

the algorithm converges, but not to the location of the true minimum. You should try different accuracies: `tol = 10^(-2)`, `tol = 10^(-3)` and `tol = 10^(-3)`, for example.

You can also try using the (fewer) start-points suggested in Section 3.5. Simply define, for example,

```
StartPoints = {{1, 2}, {3, -4}}
```

in place of the `StartPoints` code above. □

3.3.3 Newton/Quasi-Newton Method

Newton's method in 1D uses a quadratic approximation of the function at the current location and updates to the new location by taking the minimum of this quadratic approximation. This has two features: the *descent direction* is (trivially) chosen, and the *step-size* is determined. In particular, no line search is used.

Newton's method in higher D: The second order Taylor approximation for $f : \mathbb{R}^n \to \mathbb{R}$ based at $\vec{x}_k$ is

$$f(\vec{x}) \approx q(\vec{x}) := f(\vec{x}_k) + (\vec{x} - \vec{x}_k)^T \nabla f(\vec{x}_k) + \frac{1}{2}(\vec{x} - \vec{x}_k)^T D^2 f(\vec{x}_k)(\vec{x} - \vec{x}_k).$$

At a minimum for $q : \mathbb{R}^n \to \mathbb{R}$ we must have

$$\vec{0} = \nabla q(\vec{x}) = \nabla f(\vec{x}_k) + D^2 f(\vec{x}_k)(\vec{x} - \vec{x}_k)$$

so if $D^2 f(\vec{x}_k) > 0$ we then set $\vec{x}_{k+1}$ to be the minimizer of q:

$$\vec{x}_{k+1} = \vec{x}_k - D^2 f(\vec{x}_k)^{-1} \nabla f(\vec{x}_k).$$

Notes: (1) Once again, we are not using a line search here; from position $\vec{x}_k$ we go to position $\vec{x}_{k+1}$ by computing $\vec{d}_k = -D^2 f(\vec{x}_k)^{-1} \nabla f(\vec{x}_k)$ and setting $\vec{x}_{k+1} = \vec{x}_k + \vec{d}_k$.

(2) On a large problem, rather than find $D^2 f(\vec{x}_k)^{-1}$ we would solve $D^2 f(\vec{x}_k)\vec{d}_k = -\nabla f(\vec{x}_k)$ by Gaussian elimination.

Newton's method does not guarantee that the sequence generated is a "descent sequence" – it may not hold that $f(\vec{x}_{k+1}) < f(\vec{x}_k)$, even if $D^2 f(\vec{x}_k) > 0$. Note that even if we are guaranteed that at each step we have a descent *direction*, we might just go too far in that direction. This is easy to see in examples for functions of one variable. However, if the start point is sufficiently close to a local minimum, it can be shown that Newton's method has superior convergence performance.

EXERCISE 3.15. Write *Mathematica* code to implement Newton's method in 2D. Use the code for `SteepestDescent2D[ ]` on page 74 as a structural guide. Note that if `M` is a matrix, then its inverse can be found by `Inverse[M]`, and if `v = {v1, v2}` is a vector, the matrix-vector product of `M` with `v` is given by `M.v`. Warning: in `SteepestDescent[ ]` we check to see if we have met the desired tolerance by computing `Norm[Last[dDir]]`. In the steepest descent algorithm, the descent direction *coincides* with $\nabla f(\vec{x}_k)$. Here, however, we have to compute $\|\nabla f(\vec{x}_k)\|$.

The problem of not being a descent sequence can be remedied when we consider the following result:

THEOREM 3.16. *Let $\{\vec{x}_k\}$ be a sequence generated by Newton's method. If $D^2 f(\vec{x}_k) > 0$ and $\nabla f(\vec{x}_k) \neq \vec{0}$ then the direction*

$$\vec{d}_k = -D^2 f(\vec{x}_k)^{-1} \nabla f(\vec{x}_k) = \vec{x}_{k+1} - \vec{x}_k$$

is *a descent direction for f in the sense that there exists $\hat{t} > 0$ such that for all $0 < t < \hat{t}$,*

$$f(\vec{x}_k + t\vec{d}_k) < f(\vec{x}_k).$$

Proof. Set $h(t) = f(\vec{x}_k + t\vec{d}_k)$; then

$$h'(0) = \nabla f(\vec{x}_k)^T \vec{d}_k = -\nabla f(\vec{x}_k)^T D^2 f(\vec{x}_k)^{-1} \nabla f(\vec{x}_k).$$

Now, since $D^2 f(\vec{x}_k) > 0$, the eigenvalues $\lambda_1, \ldots, \lambda_n$ are all $\lambda_j > 0$; the eigenvalues of $(D^2 f(\vec{x}_k))^{-1}$ are $1/\lambda_j > 0$ and so $(D^2 f(\vec{x}_k))^{-2} > 0$ also. Thus the expression above gives $h'(0) < 0$. This implies there is $\hat{t} > 0$ such that for all $0 < t < \hat{t}$, $h(t) < h(0)$. This yields the result. □

Modified Newton's method: the above motivates:

$$\begin{aligned} \vec{d}_k &= -D^2 f(\vec{x}_k)^{-1} \nabla f(\vec{x}_k) \\ t_k &= \operatorname*{argmin}_{t \geq 0} f(\vec{x}_k + t\vec{d}_k) \\ \vec{x}_{k+1} &= \vec{x}_k + t_k \vec{d}_k; \end{aligned}$$

determination of t_k is achieved via a *line search* as we studied earlier.

EXERCISE 3.17. Write *Mathematica* code to implement this modified Newton's method in 2D. If you have completed Exercise 3.15 then you should be able to modify that code to incorporate a line search in the manner of the code for `SteepestDescent2D[ ]` (page 74).

Quasi-Newton Methods: A drawback of Newton's method is its requirement that we compute the Hessian $D^2 f(\vec{x}_k)$ at every step (and then, for large n, solving $D^2 f(\vec{x}_k)\vec{d}_k = -\nabla f(\vec{x}_k)$ can

be computationally expensive also). Quasi-Newton methods use first-derivative information only to approximate the inverse of the Hessian matrix. If H_k is an approximation to $D^2 f(\vec{x}_k)^{-1}$ then the update $\vec{x}_{k+1}$ is defined as for the modified Newton method above,

$$\begin{aligned} \vec{d_k} &= -H_k \nabla f(\vec{x}_k) \\ t_k &= \operatorname*{argmin}_{t \geq 0} f(\vec{x}_k + t\vec{d_k}) \\ \vec{x}_{k+1} &= \vec{x}_k + t_k \vec{d_k}. \end{aligned}$$

So, we see, a quasi-Newton method has the same structure as Newton's method; the difference is that instead of using (the inverse of) $D^2 f(\vec{x}_k)$ we use an approximation H_k.

We need to come up with a scheme for obtaining the sequence of matrices H_k. To come up with possible schemes, we study what is true of *quadratic* functions, where we know $D^2 f(\vec{x}_k) = Q$. We'd like to guarantee that $\vec{x}_{k+1} - \vec{x}_k$ is a descent direction; we'll now show that this can be achieved by requiring that H_k be positive definite at each step. We will need the following notion: (B.7.11) a function f is "little-o" of $\|\vec{x}\|^r$, $f(\vec{x}) = o(\|\vec{x}\|^r)$, if $\lim_{\vec{x} \to \vec{0}} \dfrac{\|f(\vec{x})\|}{\|\vec{x}\|^r} = 0$.

If we expand f about $\vec{x}_k$ then (by Taylor's theorem)

$$\begin{aligned} f(\vec{x}_{k+1}) &= f(\vec{x}_k) + \nabla f(\vec{x}_k)^T (\vec{x}_{k+1} - \vec{x}_k) + o(\|\vec{x}_{k+1} - \vec{x}_k\|) \\ &= f(\vec{x}_k) - t_k \nabla f(\vec{x}_k)^T H_k \nabla f(\vec{x}_k) + o(\|H_k \nabla f(\vec{x}_k)\| t_k) \end{aligned}$$

where we use $\vec{x}_{k+1} - \vec{x}_k = t_k \vec{d_k} = -t_k H_k \nabla f(\vec{x}_k)$. If $\nabla f(\vec{x}_k)^T H_k \nabla f(\vec{x}_k) > 0$, which in turn can be guaranteed by requiring H_k to be positive definite, then the second term is certainly negative. We just need to verify that, for sufficiently small t_k, the error (the $o(\|H_k \nabla f(\vec{x}_k)\| t_k)$ term) is smaller than (the absolute value of) the second term. But this is precisely what *little-o* means.

How the H_k are determined separates one quasi-Newton method from another. To see how H_k might be defined, we consider the case where f is a quadratic function $f(\vec{x}) = \dfrac{1}{2} \vec{x}^T Q \vec{x} - \vec{b}^T \vec{x}$, in which case the Hessian $D^2 f(\vec{x}) = Q$ for all $\vec{x}$, Q a symmetric $n \times n$ matrix. Then

$$\nabla f(\vec{x}_{k+1}) - \nabla f(\vec{x}_k) = Q(\vec{x}_{k+1} - \vec{x}_k).$$

To make the notation a little more manageable, define $\Delta \vec{g}_k := \nabla f(\vec{x}_{k+1}) - \nabla f(\vec{x}_k)$ and $\Delta \vec{x}_k := \vec{x}_{k+1} - \vec{x}_k$. For given k, we trivially have

$$Q^{-1} \Delta \vec{g}_i = \Delta \vec{x}_i, \qquad \text{for all } 0 \leq i \leq k$$

so we impose the same on H_{k+1}: having determined $H_0, \ldots, H_k$ we require that

$$H_{k+1} \Delta \vec{g}_i = \Delta \vec{x}_i, \qquad \text{for all } 0 \leq i \leq k. \tag{3.4}$$

At the nth step, we obtain

$$H_n \big[\Delta \vec{g}_0, \ldots, \Delta \vec{g}_{n-1}\big] = \big[\Delta \vec{x}_0, \ldots, \Delta \vec{x}_{n-1}\big], \qquad \text{or} \qquad H_n G = X, \text{ say.}$$

The matrix Q^{-1} also satisfies this equation, so provided the matrix $G = \left[\Delta\vec{g}_0, \ldots, \Delta\vec{g}_{n-1}\right]$ is non-singular, after n steps, $H_n = XG^{-1} = Q^{-1}$. We will now see the significance of this: Newton's method solves *quadratic* problems in *one step* (be sure you see why this is the case!). With the quasi-Newton algorithm above, implementing the $(n+1)^{\text{st}}$ step on the quadratic function f is actually implementing a step of *Newton's method* – this last step therefore finds the minimum. In conclusion, such a quasi-Newton method will find the minimum of a quadratic function in at most $n+1$ steps. (In fact it can be shown that such quasi-Newton algorithms solve quadratic problems in at most n steps.)

We had to assume above that G was non-singular; it can be shown that for quadratic functions the set of $\Delta\vec{g}_j$ are indeed linearly independent – we'll see this later when we study "conjugate gradient methods" of which quasi-Newton methods are an example.

We use this as motivation to impose these conditions on any scheme for determining the H_k approximations to $D^2 f(\vec{x}_k)^{-1}$. A standard way to determine H_{k+1} is to add a correction to H_k – note that *any* matrix can be obtained this way, so this is no restriction. As discussed above, an additional desirable property is that each of the H_k be positive definite, for then we are guaranteed that $\vec{x}_{k+1} - \vec{x}_k$ is always a *descent* direction.

Rank-one correction: We can derive a *rank-one correction*, which means that $H_{k+1} = H_k + \vec{u}\vec{u}^T$ for some choice of $\vec{u}$ (the matrix $\vec{u}\vec{u}^T$ has rank one). We leave this as an exercise. It turns out that rank one corrections aren't rich enough to change H_k in an efficient way and for non-quadratic functions, H_{k+1} may fail to be positive definite, and then $\vec{d}_{k+1}$ can fail to be a descent direction.

DFP Algorithm: (A rank two correction.)

1. Set $k = 0$; select $\vec{x}_0$ and a real symmetric positive definite matrix H_0 (e.g. $H_0 = I$).
2. If $\nabla f(\vec{x}_k) = \vec{0}$ (or $\|\nabla f(\vec{x}_k)\| \leq \varepsilon$) stop; otherwise $\vec{d}_k = -H_k \nabla f(\vec{x}_k)$.
3. Perform a line search (or approximate line search)

$$t_k = \underset{t \geq 0}{\operatorname{argmin}} f(\vec{x}_k + t\vec{d}_k)$$

   and set $\vec{x}_{k+1} = \vec{x}_k + t_k\vec{d}_k$.

4. Compute $\Delta\vec{x}_k = \vec{x}_{k+1} - \vec{x}_k = t_k\vec{d}_k$, $\Delta\vec{g}_k = \nabla f(\vec{x}_{k+1}) - \nabla f(\vec{x}_k)$, and

$$\begin{aligned} H_{k+1} &= H_k + \frac{\Delta\vec{x}_k \Delta\vec{x}_k^T}{\Delta\vec{x}_k^T \Delta\vec{g}_k} - \frac{(H_k\Delta\vec{g}_k)(H_k\Delta\vec{g}_k)^T}{\Delta\vec{g}_k^T H_k \Delta\vec{g}_k} \\ &= H_k - t_k \frac{(H_k\nabla f(\vec{x}_k))(H_k\nabla f(\vec{x}_k))^T}{(H_k\nabla f(\vec{x}_k))^T \Delta\vec{g}_k} - \frac{(H_k\Delta\vec{g}_k)(H_k\Delta\vec{g}_k)^T}{\Delta\vec{g}_k^T H_k \Delta\vec{g}_k} \end{aligned}$$

5. Increment k and go to step 2.

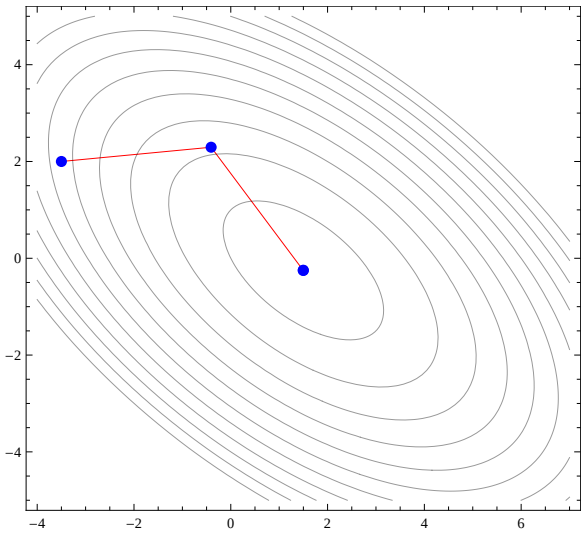
A comment regarding implementing this in *Mathematica*: As you've seen before, if $\vec{x}$ and $\vec{y}$ are n-vectors, then $\vec{x}^T\vec{y}$ is a scalar and is called the (an) inner product of $\vec{x}$ with $\vec{y}$. The product $\vec{x}\vec{y}^T$, however, is an $n \times n$ matrix, and is sometimes referred as an *outer product*. In *Mathematica*, if `x, y` are vectors, we can compute this outer product via `Outer[Times, x, y]`.

We now claim that DFP preserves positive definiteness (though we won't prove it here):

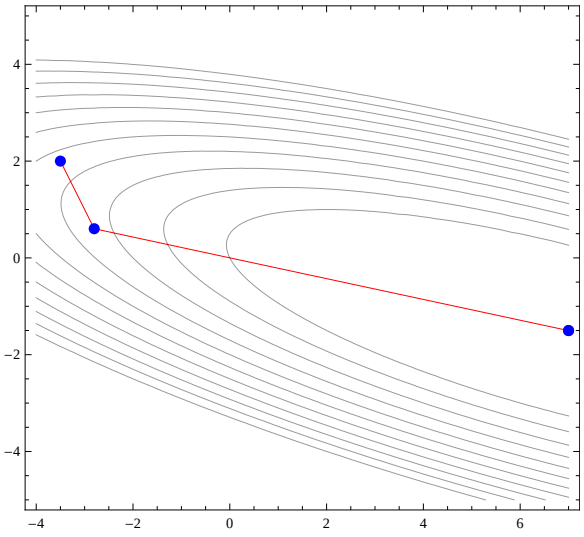
THEOREM 3.18. *Suppose $\nabla f(\vec{x}_k) \neq \vec{0}$. In the DFP algorithm, if H_k is positive definite, then so too is H_{k+1}.*

EXAMPLE 3.19. Applying DFP to the quadratics we used to demonstrate the Steepest Descent algorithm we obtain:

For $f(x,y) = \frac{3}{2}x^2 + 2xy + 2y^2 - 4x - 2y$, $\vec{x}_0 = [-3.5, 2]^T$



For $f(x,y) = \frac{1}{2}x^2 + 2xy + 4y^2 - 4x - 2y$, $\vec{x}_0 = [-3.5, 2]^T$



□

The DFP algorithm improves on the rank one correction; in certain applications it can get "stuck" if H_k becomes close to singular. Since H_k is supposed to be approximating $(D^2 f)^{-1}$, the inverse of the Hessian matrix, this means, most likely, that the determinant of $(D^2 f)^{-1} \approx 0$. In such a case, we might be better off dealing with $(D^2 f)$ itself. The BFGS algorithm below takes this approach and can alleviate the problem of the DFP algorithm getting stuck in some situations.

BFGS Algorithm: Quasi-Newton methods are based on finding matrices H which approximate the inverse of Q for quadratic functions. Specifically we required

$$H_{k+1}\Delta\vec{g}_i = \Delta\vec{x}_i, \qquad \text{for all } 0 \leq i \leq k,$$

which came from $Q^{-1}\Delta\vec{g}_i = \Delta\vec{x}_i$, for all $0 \leq i \leq k$. But these in turn came from $\Delta\vec{g}_i = Q\Delta\vec{x}_i$. The BFGS algorithm is simply based on approximating Q (by a sequence B_k) instead of Q^{-1} and requiring

$$\Delta\vec{g}_i = B_{k+1}\Delta\vec{x}_i, \qquad \text{for all } 0 \leq i \leq k.$$

One way to come up with such a sequence is to simply interchange the roles of B_k and H_k, and of $\Delta\vec{g}_k$ and $\Delta\vec{x}_k$ in a way of coming up with a sequence of H_k. If we apply this idea to DFP, we get BFGS, a scheme to update B_k:

$$B_{k+1} = B_k + \frac{\Delta\vec{g}_k \Delta\vec{g}_k^T}{\Delta\vec{g}_k^T \Delta\vec{x}_k} - \frac{(B_k\Delta\vec{x}_k)(B_k\Delta\vec{x}_k)^T}{\Delta\vec{x}_k^T B_k \Delta\vec{x}_k}.$$

This is all well-and-good except that what we need is $H_{k+1} = B_{k+1}^{-1}$ and we recently argued that inverting a matrix is inefficient. What saves us is that inverting a *rank-one perturbation* of a matrix can be done analytically by

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

Thus, looking at the form of B_{k+1} we can obtain B_{k+1}^{-1} in terms of $B_k^{-1} = H_k$. It requires applying this identity twice. One obtains

$$\begin{aligned} H_{k+1} &= H_k + \Big(1 + \frac{\Delta\vec{g}_k^T H_k \Delta\vec{g}_k}{\Delta\vec{g}_k^T \Delta\vec{x}_k}\Big)\frac{\Delta\vec{x}_k \Delta\vec{x}_k^T}{\Delta\vec{x}_k^T \Delta\vec{g}_k} - \frac{H_k\Delta\vec{g}_k\Delta\vec{x}_k^T + (H_k\Delta\vec{g}_k\Delta\vec{x}_k^T)^T}{\Delta\vec{g}_k^T\Delta\vec{x}_k} \\ &= -t_k\frac{(H_k\nabla f(\vec{x}_k))(H_k\nabla f(\vec{x}_k))^T}{(H_k\nabla f(\vec{x}_k))^T\Delta\vec{g}_k} \\ &\qquad + \Big(1 + \frac{H_k\nabla f(\vec{x}_k)\Delta\vec{g}_k^T}{(H_k\nabla f(\vec{x}_k))^T\Delta\vec{g}_k}\Big)H_k\Big(1 + \frac{\Delta\vec{g}_k(H_k\nabla f(\vec{x}_k))^T}{(H_k\nabla f(\vec{x}_k))^T\Delta\vec{g}_k}\Big). \end{aligned}$$

Apparently, BFGS is more robust even when the line search step is more sloppy and so one can save time at the line search step by not requiring such high tolerance. The BFGS algorithm is often more efficient than the DFP algorithm.

REMARK. In implementing a quasi-Newton method on a function $f : \mathbb{R}^n \to \mathbb{R}$, since the algorithm is based on achieving $H_n = Q^{-1}$ (at the n^{th} step) for a quadratic function with Hessian Q, there is no sense in which H_{n+1} is approximating $(D^2 f(\vec{x}_n))^{-1}$. After performing n steps, we should re-set $H_{n+1} = I_{n\times n}$.

3.4 Conjugate Gradient Methods

3.4.1 Introduction

The methods we've studied so far essentially differ in how we choose our next descent direction $\vec{d}_k$. To judge the efficiency of an algorithm we can consider how well it performs on quadratic functions. We've seen that even for quadratic functions, the steepest descent direction is far from optimal and can be very slow to converge (this is typified by the "zig-zag" path we saw). On the other hand, Newton's method converges quickly (indeed, on quadratic functions it converges in *one step*!) but requires computation of Hessian matrices. Conjugate gradient methods lie somewhere in between: they solve quadratic problems in n steps and require no Hessian computations.

Note: the quasi-Newton methods we presented earlier are in fact themselves conjugate gradient methods. We will show this later.

We can derive the ideas behind **conjugate gradient methods** by studying the quadratic function $f(\vec{x}) = \frac{1}{2}\vec{x}^T Q\vec{x} - \vec{x}^T\vec{b}$.

DEFINITION 3.20. Let Q be a real symmetric $n \times n$ matrix. Two directions (vectors) $\vec{d}_1, \vec{d}_2$ are said to be "Q-conjugate" if $\vec{d}_1^T Q\vec{d}_2 = 0$. Note that what this says when $Q = I$ is that $\vec{d}_1$ and $\vec{d}_2$ are *orthogonal*. Thus we are requiring $\vec{d}_1$ and $\vec{d}_2$ to be "orthogonal with respect to Q."

A *set* of vectors $\{\vec{d}_j\}_{j=1}^m$ is said to be Q-conjugate if, for all $i \neq j$, $\vec{d}_i^T Q \vec{d}_j = 0$.

LEMMA 3.21. *Let Q be a symmetric positive definite $n \times n$ matrix. If $\{\vec{d}_j\}_{j=1}^k$, $k \leq n$ are non-zero and Q-conjugate, then they are linearly independent.*

EXERCISE 3.22. Prove the above lemma.

***Q*-Gramm-Schmidt:** If we start with a linearly independent set of vectors, and apply the Gramm-Schmidt process *with inner product defined in terms of Q*, then the resulting set of vectors is a Q-conjugate set. Here we present the Gramm-Schmidt algorithm with respect to both the standard inner product on $\mathbb{R}^n$ and the inner product defined by Q. At the end of Section 3.4.3 we prove that the resulting set of vectors is indeed Q-conjugate.

The regular Gramm-Schmidt process starts with $\{\vec{v}_1, \ldots, \vec{v}_n\}$ linearly independent, sets $\vec{u}_1 = \vec{v}_1$ and then

$$\vec{u}_k = \vec{v}_k - \sum_{j=1}^{k-1} \frac{\vec{v}_k \cdot \vec{u}_j}{\vec{u}_j \cdot \vec{u}_j} \vec{u}_j.$$

The resulting set $\{\vec{u}_1, \ldots, \vec{u}_n\}$ is an orthogonal set; replacing each $\vec{u}_j$ by $\vec{u}_j / \|\vec{u}_j\|$ results in an orthonormal set.

Within the above formula is the dot-product – it is an example of an "inner product." Another example of an inner product is $\langle \vec{x}, \vec{y} \rangle = \vec{x}^T Q \vec{y}$. (Note that the dot-product is simply this with $Q = I$.) We can define a Gramm-Schmidt process with respect to this inner product by: set $\vec{u}_1 = \vec{v}_1$, and then

$$\vec{u}_k = \vec{v}_k - \sum_{j=1}^{k-1} \frac{\vec{v}_k^T Q \vec{u}_j}{\vec{u}_j^T Q \vec{u}_j} \vec{u}_j.$$

EXERCISE 3.23. If $\{\vec{u}_1, \ldots, \vec{u}_n\}$ are obtained by applying Q-Gramm-Schmidt to a collection of linearly independent vectors, then the $\{\vec{u}_j\}$ are Q-conjugate.

3.4.2 The basic Conjugate Direction algorithm

Here, we present the algorithm assuming we have some way of coming up with the Q-conjugate directions $\vec{d}_k$; we address how to find these in the next section. We also restrict our attention to the quadratic function $f(\vec{x}) = \frac{1}{2} \vec{x}^T Q \vec{x} - \vec{b}^T \vec{x}$, $Q = Q^T > 0$. (Of course the minimum in this case is simply found via $Q \vec{x}^* = \vec{b}$.) Later, we will apply the same algorithm to functions which are not quadratic, hoping that the algorithm will still perform well, because all (reasonably nice) functions are "locally like quadratics."

EXERCISE 3.24. Letting $h(t) = f(\vec{x} + t\vec{d})$, with $f(\vec{x})$ the quadratic function given above, show that the minimum of h occurs at $t = -\dfrac{\nabla f(\vec{x})^T \vec{d}}{\vec{d}^T Q \vec{d}}$.

Let $\vec{x}_0$ be a starting point and let $\vec{d}_0, \ldots, \vec{d}_{n-1}$ be non-zero conjugate directions. (Note, there can be at most n; why?) For $k \geq 0$, define

$$
\begin{aligned}
(\nabla f(\vec{x}_k) &= Q\vec{x}_k - \vec{b}) \\
t_k &= -\frac{\nabla f(\vec{x}_k)^T \vec{d}_k}{\vec{d}_k^T Q \vec{d}_k}, \qquad \text{(this minimizes a line search by Exercise 3.24 above)} \\
\vec{x}_{k+1} &= \vec{x}_k + t_k \vec{d}_k.
\end{aligned}
$$

LEMMA 3.25. *If $\vec{d}_0, \ldots, \vec{d}_{n-1}$ are non-zero Q-conjugate directions, and $\vec{x}_k$ is defined as above, then for each $1 \leq k \leq n$,*

$$\nabla f(\vec{x}_k)^T \vec{d}_j = 0 \quad \textit{for all } 0 \leq j \leq k-1.$$

Proof. Outline:

- True when $k = 1$: if $h(t) = f(\vec{x}_0 + t\vec{d}_0)$ then since t_0 minimizes h, $0 = h'(t_0) = \nabla f(\vec{x}_1)^T \vec{d}_0$.
- Assuming it is true for k we must show it is true for $k+1$. For $0 \leq j \leq k-1$ we have

$$
\begin{aligned}
\nabla f(\vec{x}_{k+1})^T \vec{d}_j &= \vec{x}_{k+1}^T Q \vec{d}_j - \vec{b}^T \vec{d}_j && \text{since } \nabla f = Q\vec{x} - \vec{b} \\
&= (\vec{x}_k^T + t_k \vec{d}_k^T) Q \vec{d}_j - \vec{b}^T \vec{d}_j && \text{since } \vec{x}_{k+1} = \vec{x}_k + t_k \vec{d}_k \\
&= \vec{x}_k^T Q \vec{d}_j - \vec{b}^T \vec{d}_j && \text{since } \vec{d}_k^T Q \vec{d}_j = 0 \text{ for } j < k \\
&= \nabla f(\vec{x}_k)^T \vec{d}_j = 0 && \text{by the induction hypothesis.}
\end{aligned}
$$

- What remains is to prove that $\nabla f(\vec{x}_{k+1})^T \vec{d}_k = 0$; for this mimic the proof of the case $k = 1$ above.

□

In particular, at x_n, $\nabla f(\vec{x}_n)^T \vec{d}_j = 0$ for all $0 \leq j \leq n-1$; but the set of conjugate directions form a basis for $\mathbb{R}^n$ so $\nabla f(\vec{x}_n) = \vec{0}$. Since there is a unique minimizer for the quadratic function f, $\vec{x}_n$ must be it!

3.4.3 The conjugate gradient algorithm – quadratic functions

We haven't yet addressed the issue of how to find the conjugate directions. The conjugate gradient algorithm progressively chooses new directions, based on the gradient of f, such that at each step the "set so far" is Q-conjugate. Once again we are working with the quadratic function $f(\vec{x}) = \frac{1}{2}\vec{x}^T Q\vec{x} - \vec{b}^T\vec{x}$ with $Q = Q^T > 0$.

- Let $\vec{x}_0$ be an initial point; for the first direction, we take the steepest descent direction $\vec{d_0} = -\nabla f(\vec{x}_0)$.
- To get to the next point we thus set $\vec{x}_1 = \vec{x}_0 + t_0\vec{d_0}$ where

$$t_0 = \underset{t\geq 0}{\operatorname{argmin}} f(\vec{x}_0 + t\vec{d_0}) = -\frac{\nabla f(\vec{x}_0)^T\vec{d_0}}{\vec{d_0}^T Q\vec{d_0}}.$$

  (Recall here we have an explicit f for which we can find the minimum.)

- We will show below that the new search direction $\vec{d_1}$, which must be Q-conjugate to $\vec{d_0}$, is given by

$$\vec{d_1} = -\nabla f(\vec{x}_1) + \beta_0\vec{d_0}, \quad \text{where} \quad \beta_0 = \frac{\nabla f(\vec{x}_1)^T Q\vec{d_0}}{\vec{d_0}^T Q\vec{d_0}}.$$

  In general,

$$\vec{d}_{k+1} = -\nabla f(\vec{x}_{k+1}) + \beta_k\vec{d_k}, \quad \text{where} \quad \beta_k = \frac{\nabla f(\vec{x}_{k+1})^T Q\vec{d_k}}{\vec{d_k}^T Q\vec{d_k}}.$$

We'll now derive the above formulae for β_k. At the first step, we have $\vec{d_0} = -\nabla f(\vec{x}_0)$ and set $\vec{x}_1 = \vec{x}_0 + t_0\vec{d_0}$. We know that $\nabla f(\vec{x}_1)$ is orthogonal to $\vec{d_0}$ (by Lemma 3.25), in particular linearly independent, so apply Q-Gramm-Schmidt to extend $\{\vec{d_0}\}$ to span$\{-\nabla f(\vec{x}_1), \vec{d_0}\}$:

$$\vec{d_1} = -\nabla f(\vec{x}_1) - \frac{(-\nabla f(\vec{x}_1))^T Q\vec{d_0}}{\vec{d_0}^T Q\vec{d_0}}\vec{d_0} = -\nabla f(\vec{x}_1) + \beta_0\vec{d_0}, \quad \text{as defined above.}$$

Next, since $\nabla f(\vec{x}_2)$ is perpendicular to both $\vec{d_0}$ and $\vec{d_1}$ (again by Lemma 3.25), apply Q-Gramm-Schmidt to extend $\{\vec{d_1}, \vec{d_0}\}$ to span$\{-\nabla f(\vec{x}_2), \vec{d_1}, \vec{d_0}\}$:

$$\vec{d_2} = -\nabla f(\vec{x}_2) - \frac{(-\nabla f(\vec{x}_2))^T Q\vec{d_1}}{\vec{d_1}^T Q\vec{d_1}}\vec{d_1} - \frac{(-\nabla f(\vec{x}_2))^T Q\vec{d_0}}{\vec{d_0}^T Q\vec{d_0}}\vec{d_0}.$$

The first term is β_1; we must show that the second term is zero. We'll do this in general: at the k^{th} step,

$$\vec{d}_{k+1} = -\nabla f(\vec{x}_{k+1}) - \frac{(-\nabla f(\vec{x}_{k+1}))^T Q\vec{d}_k}{\vec{d}_k^T Q\vec{d}_k}\vec{d}_k - \cdots - \frac{(-\nabla f(\vec{x}_{k+1}))^T Q\vec{d}_0}{\vec{d}_0^T Q\vec{d}_0}\vec{d}_0.$$

The coefficient of $\vec{d}_k$ is β_k; we will now show that the coefficient of $\vec{d}_j$ is zero for $0 \le j \le k-1$.

LEMMA 3.26. *With the $\{\vec{d}_j\}$ as defined above,*

$$\nabla f(\vec{x}_{k+1})^T Q\vec{d}_j = 0$$

for $0 \le j \le k-1$.

Proof. Let $j \le k-1$. Since $\vec{x}_{j+1} = \vec{x}_j + t_j\vec{d}_j$, and $Q\vec{x} = \nabla f(\vec{x}) + \vec{b}$ (with our explicit f), we have

$$Q\vec{d}_j = \frac{1}{t_j}Q(\vec{x}_{j+1} - \vec{x}_j) = \frac{1}{t_j}\big(\nabla f(\vec{x}_{j+1}) - \nabla f(\vec{x}_j)\big).$$

Now by construction, both $\nabla f(\vec{x}_{j+1})$ and $\nabla f(\vec{x}_j)$ are in the span of $\{\vec{d}_0, \ldots, \vec{d}_{j+1}\}$. By Lemma 3.25, $\nabla f(\vec{x}_{k+1})$ is perpendicular to $\vec{d}_0,\ldots,\vec{d}_{j+1}$ because $j+1 \le k < k+1$. Thus $\nabla f(\vec{x}_{k+1})^T Q\vec{d}_j = 0$ as desired. □

It remains to prove that the directions $\vec{d}_0, \ldots, \vec{d}_{n-1}$ are Q-conjugate.

LEMMA 3.27. *The sequence $\{\vec{d}_j\}_{j=0}^{n-1}$ is Q-conjugate.*

Proof. This follows from Exercise 3.23 following the description of Q Gramm-Schmidt. □

We now deal with some house-keeping: in deriving the quasi-Newton methods we claimed earlier that the matrix $[\Delta\vec{g}_0, \ldots, \Delta\vec{g}_{n-1}]$ was non-singular, and we claimed at the beginning of this section that quasi-Newton methods are in fact conjugate gradient methods. We'll address these claims now.

Since $\Delta\vec{g}_j = \nabla f(\vec{x}_{j+1}) - \nabla f(\vec{x}_j) = Q(\vec{x}_{j+1} - \vec{x}_j) = t_j Q\vec{d}_j$, if we show that the set $\{\vec{d}_0, \ldots, \vec{d}_{n-1}\}$ is Q-conjugate, then we will have proven the second claim, *and* we can conclude that the set $\{\Delta\vec{g}_0, \ldots, \Delta\vec{g}_{n-1}\}$ is linearly independent, thus proving the first claim. The latter follows because Q-conjugate implies $\{\vec{d}_0, \ldots, \vec{d}_{n-1}\}$ is linearly independent, and then since Q is non-singular, this implies $\{Q\vec{d}_0, \ldots, \vec{Q}d_{n-1}\}$ is linearly independent.

THEOREM 3.28. *Let $f(\vec{x}) = \frac{1}{2}\vec{x}^T Q\vec{x} - \vec{b}^T\vec{x}$ and H_{k+1} be symmetric matrices with $H_{k+1}\Delta\vec{g}_j = \Delta\vec{x}_j$ for $0 \le k < n-1$, $0 \le j \le k$. Then with $\vec{d}_j = -H_j\nabla f(\vec{x}_j)$ and $\vec{x}_{j+1} = \vec{x}_j + t_k\vec{d}_j$, the set $\vec{d}_0, \ldots, \vec{d}_{k+1}$ is Q-conjugate.*

Proof. The proof is outlined in the exercises. □

3.4.4 The conjugate gradient algorithm – non-quadratic functions

To extend the CGA to non-quadratic functions, we argue that it should behave similarly near the minimum because of the quadratic approximation to f. We could apply this directly if we were willing to compute the Hessian matrix; Q appears in the computation of t_k and of β_k. We can replace the formula for t_k by a line search algorithm to get an approximate t_k. Assuming we have a scheme to come up with β_k we have the algorithm (for $f : \mathbb{R}^n \to \mathbb{R}$)

1. Set $k = 0$, choose $\vec{x}_0$.
2. If $\nabla f(\vec{x}_0) = \vec{0}$, stop, otherwise set $\vec{d}_0 = -\nabla f(\vec{x}_0)$.
3. Perform a line search
$$t_k = \operatorname*{argmin}_{t \geq 0} f(\vec{x}_k + t\vec{d}_k)$$
and set $\vec{x}_{k+1} = \vec{x}_k + t_k\vec{d}_k$.
4. If $\nabla f(\vec{x}_{k+1}) = \vec{0}$, stop.
5. Compute β_k (dependent on choice of scheme).
6. Set $\vec{d}_{k+1} = -\nabla f(\vec{x}_{k+1}) + \beta_k\vec{d}_k$.
7. Set $k = k + 1$; go to 3.

As for coming up with β_k without computation of a Hessian matrix we present the following formulae:

Hestenes–Steifel: In the quadratic case, $\nabla f(\vec{x}_{k+1}) - \nabla f(\vec{x}_k) = t_k Q\vec{d}_k$, so we can set

$$Q\vec{d}_k = \frac{1}{t_k}\big(\nabla f(\vec{x}_{k+1}) - \nabla f(\vec{x}_k)\big),$$

and do so even in the non-quadratic case. Replacing this for $Q\vec{d}_k$ in the formula for β_k we obtain

$$\beta_k = \frac{\nabla f(\vec{x}_{k+1})^T\big(\nabla f(\vec{x}_{k+1}) - \nabla f(\vec{x}_k)\big)}{\vec{d}_k^T\big(\nabla f(\vec{x}_{k+1}) - \nabla f(\vec{x}_k)\big)}. \tag{3.5}$$

Polak–Ribiere: In the denominator in (3.5), we observe that $\vec{d}_k^T\nabla f(\vec{x}_{k+1}) = 0$; furthermore,

$$\nabla f(\vec{x}_k)^T\vec{d}_k = \nabla f(\vec{x}_k)^T\big(-\nabla f(\vec{x}_k) + \beta_{k-1}\vec{d}_{k-1}\big) = -\nabla f(\vec{x}_k)^T\nabla f(\vec{x}_k)$$

by the same argument. Thus

$$\beta_k = \frac{\nabla f(\vec{x}_{k+1})^T \big(\nabla f(\vec{x}_{k+1}) - \nabla f(\vec{x}_k)\big)}{\nabla f(\vec{x}_k)^T \nabla f(\vec{x}_k)}. \tag{3.6}$$

Fletcher–Reeves:

LEMMA 3.29. *Let $\{\vec{x}_j\}$ be a sequence of points generated by way of a conjugate gradient method. Then for $0 \leq k \leq n-1$, $0 \leq j \leq k-1$, $\nabla f(\vec{x}_{k+1})^T \nabla f(\vec{x}_j) = 0$.*

Proof. From the claim in Section 3.4.2,

$$\begin{aligned} 0 = \nabla f(\vec{x}_{k+1})^T \vec{d}_j &= -\nabla f(\vec{x}_{k+1})^T \nabla f(\vec{x}_j) + \beta_{j-1} \nabla f(\vec{x}_{k+1})^T \vec{d}_{j-1} \\ &\Rightarrow \quad \nabla f(\vec{x}_{k+1})^T \nabla f(\vec{x}_j) = 0. \end{aligned} \tag{3.7}$$

□

By (3.7), $\nabla f(\vec{x}_{k+1})^T \nabla f(\vec{x}_k) = 0$. Applying this to (3.6) we obtain

$$\beta_k = \frac{\nabla f(\vec{x}_{k+1})^T \nabla f(\vec{x}_{k+1})}{\nabla f(\vec{x}_k)^T \nabla f(\vec{x}_k)}. \tag{3.8}$$

REMARK. Note that all the above are derived from the *quadratic* case, so there's really no sense of the vectors $\vec{d}_k$ being Q conjugate (what Q?). If we are close to the minimum, however, then they could be close to being Q-conjugate where Q is the Hessian matrix at the minimum. Furthermore, as was the case for quasi-Newton methods, the algorithm really only makes sense for n steps; for this reason, and because of numerical "wandering" it is usual to run the algorithm for n steps, then re-set $\beta_n = 0$ so that we start again in a steepest descent direction.

3.5 Test functions

We present here some functions of two, three and four variables on which to implement the algorithms discussed. See the *Mathematica* notebook "Approx Test Functions.nb".

by the same argument. Thus

$$\beta_k = \frac{\nabla f(\vec{x}_{k+1})^T \big(\nabla f(\vec{x}_{k+1}) - \nabla f(\vec{x}_k)\big)}{\nabla f(\vec{x}_k)^T \nabla f(\vec{x}_k)}. \tag{3.6}$$

Fletcher–Reeves:

LEMMA 3.29. *Let $\{\vec{x}_j\}$ be a sequence of points generated by way of a conjugate gradient method. Then for $0 \leq k \leq n-1$, $0 \leq j \leq k-1$, $\nabla f(\vec{x}_{k+1})^T \nabla f(\vec{x}_j) = 0$.*

Proof. From the claim in Section 3.4.2,

$$\begin{aligned} 0 = \nabla f(\vec{x}_{k+1})^T \vec{d}_j &= -\nabla f(\vec{x}_{k+1})^T \nabla f(\vec{x}_j) + \beta_{j-1} \nabla f(\vec{x}_{k+1})^T \vec{d}_{j-1} \\ &\Rightarrow \quad \nabla f(\vec{x}_{k+1})^T \nabla f(\vec{x}_j) = 0. \end{aligned} \tag{3.7}$$

□

By (3.7), $\nabla f(\vec{x}_{k+1})^T \nabla f(\vec{x}_k) = 0$. Applying this to (3.6) we obtain

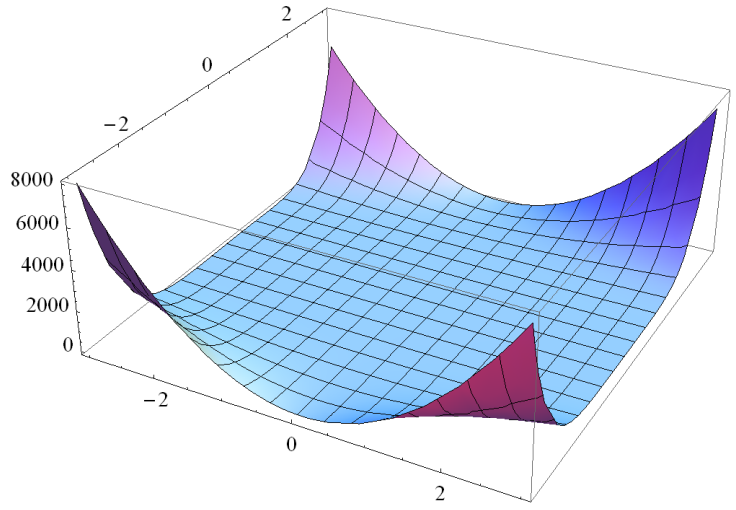
$$\beta_k = \frac{\nabla f(\vec{x}_{k+1})^T \nabla f(\vec{x}_{k+1})}{\nabla f(\vec{x}_k)^T \nabla f(\vec{x}_k)}. \tag{3.8}$$

REMARK. Note that all the above are derived from the *quadratic* case, so there's really no sense of the vectors $\vec{d}_k$ being Q conjugate (what Q?). If we are close to the minimum, however, then they could be close to being Q-conjugate where Q is the Hessian matrix at the minimum. Furthermore, as was the case for quasi-Newton methods, the algorithm really only makes sense for n steps; for this reason, and because of numerical "wandering" it is usual to run the algorithm for n steps, then re-set $\beta_n = 0$ so that we start again in a steepest descent direction.

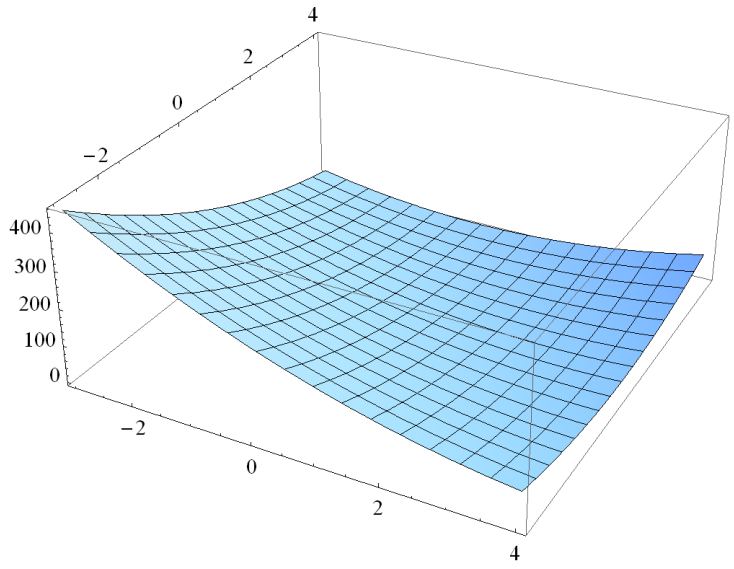
3.5 Test functions

We present here some functions of two, three and four variables on which to implement the algorithms discussed. See the *Mathematica* notebook "Approx Test Functions.nb".

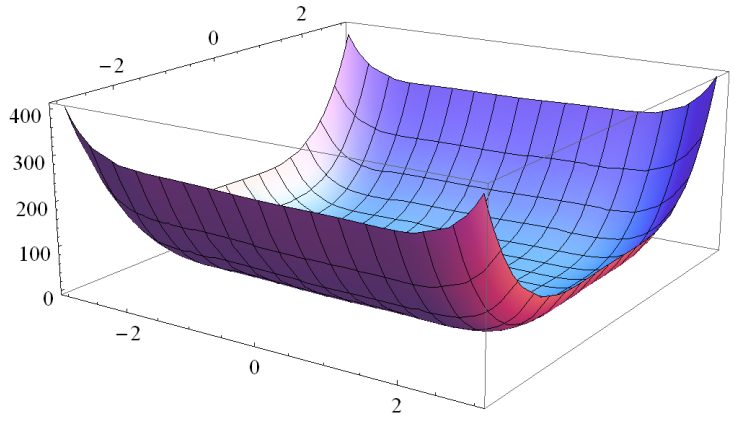
1. Beale Function: $f(x, y) = (1.5 - x - xy)^2 + (2.5 - x + xy^2)^2 + (2.625 - x + xy^3)^2$.
   Search region: $[-3, 3] \times [-3, 3]$.
   Search start-points: $[2.5, -0.3], [3, -1], [-1, 1], [1, 1]$
   Note: The minimum is near $[2.6, -0.35]^T$; there are three other locations where $\nabla f = \vec{0}$.



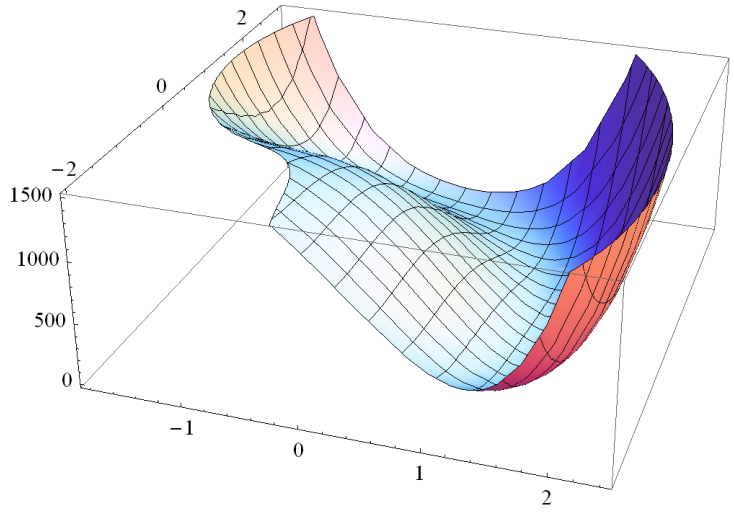
2. Booth Function: $f(x, y) = (x + 2y - 7)^2 + (2x + y - 5)^2$.
   Search region: $[-3, 4] \times [-3, 2]$.
   Search start-points: $[1.5, 2.5], [2, 2], [0, 0], [0, 4]$
   Note: There is a unique place where $\nabla f = \vec{0}$, but it can be tricky to find numerically. Newton's method will, of course, find it since the function is a quadratic.



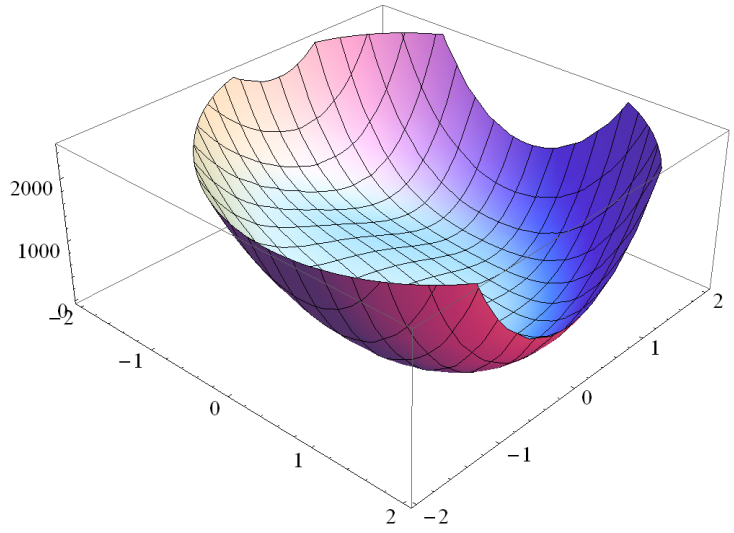
3. Hump Function: $f(x, y) = 4x^2 - 2.1x^4 + \frac{1}{3}x^6 + xy - 4y^2 + 4y^4$.
   Search region: $[-2, 2] \times [-2, 2]$.
   Search start-points: $[0, 0.5], [0, 1], [-1, 2], [2, 2]$
   Note: There are 15(!) locations where $\nabla f = \vec{0}$, although they come in pairs since $f(x, y) = f(-x, -y)$. The true minimum is close to $\pm[0.1, -0.7]^T$.



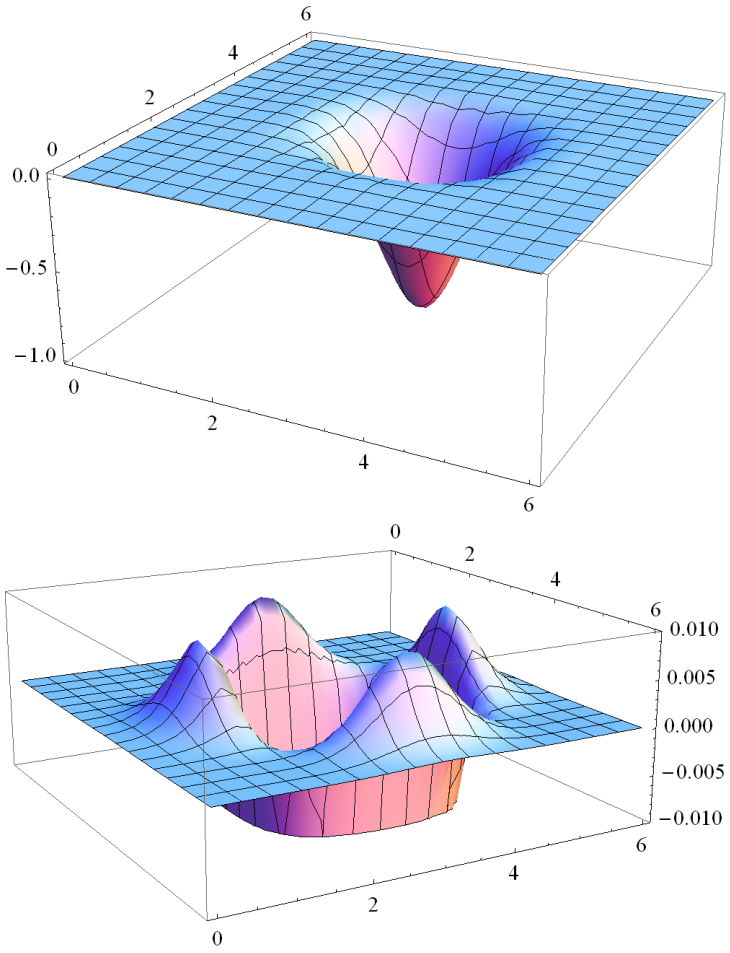
4. 2D Perm Function 1: $f(x,y) = \sum_{k=1}^{2}\sum_{i=1}^{2}(i^k + 10)\Big((\frac{x_i}{i})^k - 1\Big)$.
   Search region: $[0,3] \times [0,3]$.
   Search start-points: $[0.8, 2.2], [0.5, 2.5], [1.5, 1.5], [2,2], [0,0], [3,3]$
   Note: There are three places where $\nabla f = \vec{0}$ and the values of the function at two of these is 0 and at the third is 0.003.



5. 2D Perm Function 2: $f(x,y) = \sum_{k=1}^{2}\sum_{i=1}^{2}(i + 10)\Big(x_i^k - (\frac{1}{i})^k\Big)$.
   Search region: $[-1,2] \times [-1,2]$.
   Search start-points: $[0.8, 0.3], [0.5, 1], [0,0], [0,1], [-1,2]$
   Note: Again, there are three places where $\nabla f = \vec{0}$ and the values of the function at two of these is 0 and at the third is 0.63.



6. Easom Function: $f(x, y) = -\cos x \cos y \exp(-(x-\pi)^2 - (y-\pi)^2)$.
   Search region: $[-1, 5] \times [-1, 5]$.
   Search start-points: $[3, 3], [3.5, 3.5], [3, 4], [1, 2]$
   Notes: The function is very flat away from the hole; furthermore, as we can see in the second plot below, there are "ripples" around the edge of the sink-hole. Close to the minimum, however, it is well-behaved. The true minimum is at $[\pi, \pi]^T$.



7. 3D Sum of Squares: $f(\vec{x}) = \sum_{i=1}^{3} x_i^2$;
   Search region: $[-2, 2] \times [-2, 2] \times [-2, 2]$
   Search start-points: $[1, 1, 1], [4, 3, 5], [8, -8, 6]$
   Note: The minimum is clearly at $[0, 0, 0]^T$.

8. 3D Perm Function: $f(\vec{x}) = \sum_{k=1}^{2} \sum_{i=1}^{2} (i + 10)\left(x_i^k - (\frac{1}{i})^k\right)$;
   Search region: $[-3, 3] \times [-3, 3] \times [-3, 3]$
   Search start-points: $[0.9, 0.6, 0.4], [0.8, 0.7, 0.5], [0.7, 0.8, 0.6]$
   Note: There are 11(!) places where $\nabla f = \vec{0}$, and at many of these, the value of f is very

close to zero. The true minimum is at $[1, 1/2, 1/3]^T$.

9. 4D Rosenbrock function: $f(\vec{x}) = \sum_{i=1}^{3} \Big(100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2 \Big)$;
   Search region $[-5, 10]^4$
   Search start-points: $[1, 2, 3, 4], [-3, -5, 6, 7]$
   Note: The true minimum is at $[1, 1, 1, 1]^T$.

10. 4D Power Sum Function: $f(\vec{x}) = \sum_{k=1}^{4} \Big(\sum_{i=1}^{4} x_i^4 - b_k \Big)^2$, $\vec{b}_k = [8, 18, 44, 114]^T$;
    Search region: $[0, 4]^4$
    Search start-points: $[0.8, 2.2, 2.3, 2.8], [0.5, 1.5, 2.5, 2]$
    Note: The true minimum is at $[1, 2, 2, 3]^T$.

11. No Name Function: $f(\vec{x}) = \sum_{i=1}^{4} \sum_{j=1}^{4} (i^j + 10) \Big(\big(\frac{x_i}{i}\big)^j - 1 \Big)$;
    Search region: $[-3, 3]^4$
    Search start-points: $[-0.5, -1, -1, -1], [0, 0, 0, 0], [-1, -2, -2, -2], [10, 10, 10, 10]$
    Note: This true minimum is close to $[-0.6, -0.94, -0.97, -0.95]^T$.

3.6 Exercises

1. Exercise 3.2, page 61; see Exercise 6 below.
2. Exercise 3.3, page 62.
3. Exercise 3.5, page 65; see Exercise 6 below.
4. Exercise 3.6, page 67.
5. After completing Exercise 3.5, page 65, modify your code to implement the Fibonacci search method; see Exercise 6 below.
6. Let $f(t) = \log t + 6t$, $t > 0$. A plot of f will show that f has a zero between 0 and 1 and that f is increasing there. Thus, by defining $F(t) = \int_a^t f(s)\, ds$ for any a, we see that the zero of f coincides with the minimum of F. Compute $F(t)$ (there is an advantageous choice of a) and implement the following algorithms (by hand) to search for this minimum:
   (a) Perform several iterations of the Fixed Step algorithm (by hand) starting with $t_0 = 0.3$ and an initial *step size* of 0.2.
   Now implement the `FixedStep[ ]` code given on page 59 and compare your answers.

(b) Perform a few iterations of Newton's Method (by hand) starting with $x_0 = 1.0$.

After having completed exercise 3.2, page 61, implement your *Mathematica* code and compare your answers.

(c) Perform the first three iterations of the Golden Section method, starting with the initial interval $[0.1, 0.4]$.

After having completed exercise 3.5, page 65, implement your *Mathematica* code and compare your answers. Implement your code to obtain the best approximation to the minimum using 8 iterations.

(d) Assuming that you are seeking an accuracy of no more than 0.01, perform the first three iterations of the Fibonacci method, starting with $[0.1, 0.4]$.

After completing Exercise 5 above, implement your own code to compare these first three iterations, and to compute further iterations.

7. Define $f(t) = \int_0^t e^{s^3 \sin t}\, ds$.

(a) Compute $f'(t)$ and recognize that it is impossible to analytically solve for $f'(t) = 0$ as a way of finding a minimum.

In fact, $f(t)$ has a unique minimum in $[-4, -1]$. In *Mathematica*, define the pure function

```
f = Function[t, NIntegrate[Exp[Sin[t]s^3], {s, 0, x}]]
```

where we have used `NIntegrate[ ]` for numerical integration since we cannot find a closed form anti-derivative of the integrand. Find the minimum of f to within the given accuracy using each of your algorithms for:

(b) Fixed Step – find t so that $|f'(t)| < 10^{-2}$. Use `NumberForm[results, 8]` to display results with up to 8 digits;

(c) 1D Newton's method – again find t so that $|f'(t)| < 10^{-2}$. Note: you should find that this methods fails unless you give an initial point quite close to the true minimum.;

(d) Golden Section line search – find the minimum point to within 10^{-4};

(e) Fibonacci line search – find the minimum point to within 10^{-4}.

8. For each of the following functions and initial points,

- implement your Golden Section code (see page 65);
- implement your code for the Fibonacci search method.

In each case, find the minimum to within 0.01. You will need to use the suggested algorithm on page 68 to obtain an initial interval.

(a) $f(t) = t^4 - t^2 + t - 1$, $t_0 = 1$.

(b) $g(t) = t^{16} + 3t^{14} - t^7 - 3$, $t_0 = -1$.

(c) $h(t) = \displaystyle\int_{-1}^{t} \frac{se^s + s - 1}{2e^s + 3}\, ds$, $t_0 = -1$. (Note: you will need to use a numerical integration to evaluate h.)

9. Let $f(t) = \dfrac{1}{100}(t^6 - 30t^4 + 192t^2 + 7t^3)$.

   (a) Using Newton's Method with $t_0 = -8$ yields -4.13578, and with $t_0 = -3$ yields -1.86514. Explain!

   (b) Use your Golden Section code to perform 8 iterations, starting with $[-8, -3]$.

   (c) Use your Fibonacci code assuming that you are willing to perform 8 iterations, starting with $[-8, -3]$; compare the accuracy to that of Golden Section. (To determine the accuracy, you will need to know the true location of the minimum – you could try running Golden Section for a very large number of iterations to obtain very high accuracy. You can also use the *Mathematica* function

   `FindRoot[f'[t], {t, t0}]]`

   where `t0` is an initial guess.)

10. In the Fibonacci search method we claimed that the optimal choice of $\rho_1, \rho_2, \ldots \rho_N$ was

$$\rho_1 = 1 - \frac{F_{N+1}}{F_{N+2}}, \ldots, \rho_k = 1 - \frac{F_{N-k+2}}{F_{N-k+3}}, \ldots, \rho_N = 1 - \frac{F_2}{F_3}.$$

Here, $F_{-1} = 0$, $F_0 = 1$ and $F_{k+1} = F_k + F_{k-1}$. More precisely, this sequence minimizes $(1 - \rho_1)(1 - \rho_2) \cdots (1 - \rho_N)$ subject to $\rho_{k+1} = 1 - \dfrac{\rho_k}{1 - \rho_k}$, $k = 1, \ldots, N - 1$ and $0 \le \rho_k \le \dfrac{1}{2}$, $k = 1, \ldots, N$. If $r_k = 1 - \rho_k$ we re-pose this as

$$\begin{aligned} &\text{minimize} && r_1 \cdots r_N \\ &\text{subject to} && r_{k+1} = \frac{1}{r_k} - 1,\ k = 1, \ldots, N - 1 \\ & && \frac{1}{2} \le r_k \le 1,\ k = 1, \ldots, N. \end{aligned}$$

   (a) Show that the constraint $r_k \le 1$ can be dropped (it is implied by the other constraints).

   (b) Prove: Lemma 1. For $k \ge 2$, $r_k = -\dfrac{F_{k-2} - r_1 F_{k-1}}{F_{k-3} - r_1 F_{k-2}}$.

   (c) Prove: Lemma 2. For $k \ge 2$, $(-1)^k (F_{k-2} - r_1 F_{k-1}) > 0$.

   (d) Prove: Lemma 3. For $k \ge 2$, $F_{k-2} F_{k+1} - F_{k-1} F_k = (-1)^k$.

Accept without proof (prove it if you wish, it's not hard, just uninteresting):

Lemma 4. For $k \ge 2$, $(-1)^{k+1} r_1 \ge (-1)^{k+1} \dfrac{F_k}{F_{k+1}}$.

   (e) Prove that $r_1 \cdots r_N \ge \dfrac{1}{F_{N+1}}$.

Thus, since choosing $r_1 = \dfrac{F_{N+1}}{F_{N+2}}$ (and the rest of the r_k accordingly) actually *achieves* this bound, this sequence is optimal.

11. The *bisection method* finds the zero of a function $h(t)$, starting with the interval $[a_0, b_0]$ with $h(a_0)$ and $h(b_0)$ of differing signs. This can be used to find the minimum of a differentiable function starting with the interval $[a_0, b_0]$ with $h'(a_0) > 0 > h'(b_0)$, say. With $c = (a_0 + b_0)/2$, the midpoint of the interval, we compute $h'(c)$ and, depending on the sign of $h'(c)$, set $[a_1, b_1] = [a_0, c]$ or $[a_1, b_2] = [c, b_0]$ so that the zero of $h'(t)$ lies within the new interval. The minimum number of function evaluations which can be used to estimate $h'(c)$ is two; assuming this, show that the bisection method is less efficient than the Golden Section method by finding inequalities analogous to (3.1) and (3.2).

12. Exercise 3.12, page 71.

13. Exercise 3.13, page 71.

14. Exercise 3.14, page 74.

15. Exercise 3.15, page 77 (Newton2D). Test your code on the Beale, Hump, Perm 1, and Perm 2 functions from Section 3.5. Use the suggested start-points. Use a search tolerance of 10^{-2}.

16. Exercise 3.17, page 77 (ModifiedNewton2D). Test your code on the Booth, Hump, Perm 1 and Perm 2 functions from Section 3.5. Use the suggested start-points. Use a search tolerance of 10^{-3} and a line search tolerance of 10^{-3}. Note that the Booth function is quadratic, so Newton's method would give the exact minimum in one step. Compare the performance of modified Newton's method to Newton's method from Exercise 15 in the cases of the Hump, Perm 1 and Perm 2 functions.

17. Exercise 3.22, page 83.

18. Exercise 3.23, page 83.

19. Exercise 3.24, page 84.

20. In the section on quasi-Newton methods, it was mentioned that (page 79) quasi-Newton methods are in fact conjugate gradient methods. What this means is, when applied to a quadratic function, the set of descent directions $\{\vec{d}_k\}$ is a Q-conjugate set, where Q is the Hessian matrix. More precisely,

THEOREM 3.30. *Let* $f(\vec{x}) = \dfrac{1}{2}\vec{x}^T Q\vec{x} - \vec{b}^T\vec{x}$, *where* $Q = Q^T$. *Assume that for* $0 \le k < n-1$, H_k *are symmetric matrices satisfying*

$$H_{k+1}\Delta\vec{g}_j = \Delta\vec{x}_j, \qquad 0 \le j \le k,$$

where $\Delta\vec{g}_j = \nabla f(\vec{x}_{j+1}) - \nabla f(\vec{x}_j)$ *and* $\Delta\vec{x}_j = \vec{x}_{j+1} - \vec{x}_j$. *With* $\vec{d}_0 = -\nabla f(\vec{x}_0)$ *and* $\vec{d}_j = -H_j\nabla f(\vec{x}_j)$ *for* $j > 0$, *and* $\vec{x}_{j+1} = \vec{x}_j + t_j\vec{d}_j$ ($t_j > 0$ *found as usual), the set* $\{\vec{d}_0, \ldots, \vec{d}_{k+1}\}$ *is* Q*-conjugate.*

Prove this theorem. (Hint: use induction; when $k = 0$ it will be useful to show that $\Delta\vec{g}_0 = Q\Delta\vec{x}_0$ and to use $\vec{d}_0 = \Delta\vec{x}_0/t_0$. In all cases you will need to use Lemma 3.25.)

21. In Quasi-Newton methods, one needs a way to update H_k to H_{k+1} so that (3.4), page 78 holds. In this exercise you will derive a rank-one correction. Let $\Delta\vec{g}_i$ and $\Delta\vec{x}_i$ be defined as on page 78 and define $H_{k+1} = H_k + \vec{u}\vec{u}^{\,T}$.

    (a) Show that requiring $H_{k+1}\Delta\vec{g}_k = \Delta\vec{x}_k$ implies $\vec{u} = \dfrac{\Delta\vec{x}_k - H_k\Delta\vec{g}_k}{\vec{u}^{\,T}\Delta\vec{g}_k}$.

    (b) Show that

    $$\vec{u}\vec{u}^{\,T} = \frac{(\Delta\vec{x}_k - H_k\Delta\vec{g}_k)(\Delta\vec{x}_k - H_k\Delta\vec{g}_k)^T}{(\vec{u}^{\,T}\Delta\vec{g}_k)^2}.$$

    (c) Derive $(\vec{u}^{\,T}\Delta\vec{g}_k)^2 = \Delta\vec{g}_k^{\,T}(\Delta\vec{x}_k - H_k\Delta\vec{g}_k)$, and hence the rank-one correction

    $$H_{k+1} = H_k + \frac{(\Delta\vec{x}_k - H_k\Delta\vec{g}_k)(\Delta\vec{x}_k - H_k\Delta\vec{g}_k)^T}{\Delta\vec{g}_k^{\,T}(\Delta\vec{x}_k - H_k\Delta\vec{g}_k)}. \tag{3.9}$$

    (d) Suppose that the function to which we apply the above update formula is the quadratic function $f(\vec{x}) = \frac{1}{2}\vec{x}^{\,T}Q\vec{x} - \vec{b}^{\,T}\vec{x}$, so that $\Delta\vec{g}_i = Q\Delta\vec{x}_i$ for all i. Show that in fact $H_{k+1}\Delta\vec{g}_i = \Delta\vec{x}_i$ for all $0 \le i \le k$ (we have shown it holds when $i = k$ above) by induction, as follows.

        i. Explain why it holds when $k = 0$.

        ii. Assume that $H_k\Delta\vec{g}_i = \Delta\vec{x}_i$, $i < k$, holds for some $k > 0$. In order to show that $H_{k+1}\Delta\vec{g}_i = \Delta\vec{x}_i$, deduce that it suffices to show that

        $$(\Delta\vec{x}_k - H_k\Delta\vec{g}_k)^T\Delta\vec{g}_i = \Delta\vec{x}_k^{\,T}\Delta\vec{g}_i - \Delta\vec{g}_k^{\,T}H_k\Delta\vec{g}_i = 0. \tag{3.10}$$

        iii. Use the induction hypothesis and the fact that f is quadratic to show that

        $$\Delta\vec{g}_k^{\,T}H_k\Delta\vec{g}_i = \Delta\vec{g}_k^{\,T}\Delta\vec{x}_i = \Delta\vec{x}_k^{\,T}\Delta\vec{g}_i \tag{3.11}$$

        thus completing the proof.

22. Let Q be an $n \times n$ real symmetric matrix. Suppose that Q is positive definite. Let $\{\vec{d}_1, \ldots, \vec{d}_n\}$ be a Q-conjugate set which is also orthogonal (i.e. both $\vec{d}_i^{\,T}Q\vec{d}_j = 0$ and $\vec{d}_i^{\,T}\vec{d}_j = 0$, $i \ne j$).

    (a) Show that, if $\vec{d}_i \ne \vec{0}$ (for every $i = 1, \ldots, n$) then $\vec{d}_i$ is an eigenvector of Q.

    (b) What are the corresponding eigenvalues? (It's clearly possible to answer (a) and (b) at the same time, but there are easier ways of answering (a) without needing to find the eigenvalues.)

23. Let $f(x, y) = \dfrac{5}{2}x^2 + \dfrac{1}{2}y^2 + 2xy - 3x - y$.

(a) Express f in the form $f(\vec{x}) = \frac{1}{2}\vec{x}^T Q\vec{x} - \vec{b}^T\vec{x}$.

(b) Find the minimizer of f using the conjugate gradient algorithm with $\vec{x}_0 = (0,0)$. (Note: you know how quickly this should converge!)

(c) Calculate the minimizer of f analytically from Q and $\vec{b}$ and check it against your answer in (b).

24. Let $f(x,y) = 2x^2 - 2xy + y^2 + 2x - 2y$. Perform two iterations of the quasi-Newton method "DFP" with $\vec{x}_0 = [1,0]^T$ and $H_0 = I_{2\times 2}$. At the line search step, use the exact minimizer (since it is a quadratic function).

25. Prove Theorem 3.28, page 86. (Hints: Use induction; the set $\{\vec{d}_0\}$ is trivially Q-conjugate; assume that $\{\vec{d}_0, \ldots, \vec{d}_k\}$ is Q-conjugate, $k < n-1$; prove that $\vec{d}_{k+1}^T Q\vec{d}_j = 0$ for arbitrary $0 \le j \le k$. To do this, first prove that $\vec{d}_{k+1}^T Q\vec{d}_j = -\nabla f(\vec{x}_{k+1})^T\vec{d}_j$, and then use Lemma 3.25.)

26. Write *Mathematica* code to implement the following algorithms in two dimensions: (Note that having written code for one algorithm, the code for others should be obtainable by way of small modifications to the initial code.)

(a) Quasi-Newton using the rank-one correction given in by equation (3.9) in Exercise 21.

(b) DFP;

(c) BFGS;

In each of the algorithms which require a line search, use your code for Golden Section outlined on page 65.

Try these algorithms on the functions of Section 3.5 and compare their performance to modified Newton's method.

27. Extend your codes of Exercises 15, 16 and 26 to handle functions of three variables. Test the algorithms on the 3D Perm and SumOfSquares functions of Section 3.5. Use the suggested start-points, search tolerance of 10^{-3} and line search tolerance of 10^{-3}.

28. Extend your codes of Exercise 26 to handle functions of four variables. Test the algorithms on all three 4D functions in Section 3.5. Use the suggested start-points. First try a search tolerance of 10^{-2} and line search tolerance of 10^{-3}, then try a search tolerance of 10^{-3} and line search tolerance of 10^{-4}.

29. Solve Problem 5a in Chapter 1 for a rectangle of dimensions $\varphi \times 1$ where $\varphi = (1+\sqrt{5})/2$.

30. Solve Problem 5b in Chapter 1 for a box of dimensions $2 \times 3 \times 4$.

31. Solve the first of Problem 6 in Chapter 1:

Given S_2 find c_2 which satisfies (1.1) by using the Golden Section line search to minimize $h(c_2) = (f(c_2) - S_2)^2$ where f is given by the right-hand-side of (1.1). Try to determine

what you should use as an initial bracket. You should define

```
f = NIntegrate[LegendreP[2, Cos[θ]]
      Exp[#LegendreP[2, Cos[θ]]]Sin[θ], {θ, 0, π}]/
    NIntegrate[Exp[#LegendreP[2, Cos[θ]]] Sin[θ],
      {θ, 0, π}]&
```

and you can check your solution `c2` by computing `f[c2]` and comparing it to the S_2 you start with.

Notes: `LegendreP[n, x]` is the n^{th} Legendre polynomial (as a function of `x`); `#` is the place-holder for the variable, since we are defining a pure function; the `&` at the end declares the function as a pure function: `h = #^2&` is shorthand for `h = Function[#^2]`.

32. Solve the second of Problem 6 in Chapter 1: Given S_2 and S_4, find c_2 and c_4 which satisfy both (1.2) and (1.3) by minimizing $(S_2 - f_1(c_2, c_4))^2 + (S_4 - f_2(c_2, c_4))^2$, where $f_1(c_2, c_4)$ and $f_2(c_2, c_4)$ are the right hand sides of (1.2) and (1.3), respectively.

    Notes

    - You will need to write a new version of (for example) `DFP2D[ ]` for this problem. Since the function f will be given in terms of a numerical integral, *Mathematica* cannot compute derivatives (for example). You will need to build specific information about f into the algorithm. It is quite reasonable not to define f as a *pure* function here since we are not passing it as an argument for a module. You should define your functions with *delayed* evaluation.

    - Define two functions (similar to `f` in Exercise 31 above) for (1.2) and (1.3):

      ```
      f1[{c2_, c4_}] :=
        NIntegrate[LegendreP[2, Cos[θ]]
          Exp[c2 LegendreP[2, Cos[θ]]+c4 LegendreP[4, Cos[θ]]]
          Sin[θ], {θ, 0, π},
          Method -> {Automatic, ''SymbolicProcessing'' -> 0}]/
        NIntegrate[
          Exp[c2 LegendreP[2, Cos[θ]]+c4 LegendreP[4, Cos[θ]]]
          Sin[θ], {θ, 0, π},
          Method -> {Automatic, ''SymbolicProcessing'' -> 0}];
      ```

      and similarly for `f2`. Note the use of `Method -> ...`. This forces *Mathematica* to not try to do any symbolic manipulation, which slows the function evaluation down catastrophically.

    - Compute the derivatives and define four functions to be $\frac{\partial f1}{\partial c2}, \frac{\partial f1}{\partial c4}, \frac{\partial f2}{\partial c2}, \frac{\partial f2}{\partial c4}$. You

could name them `d1f1, d2f1, d1f2, d2f2`. For example,

$$\frac{\partial f1}{\partial c2} = \frac{\int_0^\pi P_2(\cos\theta)^2 \exp\left[c_2 P_2(\cos\theta) + c_4 P_4(\cos\theta)\right] \sin\theta \, d\theta}{\int_0^\pi \exp\left[c_2 P_2(\cos\theta) + c_4 P_4(\cos\theta)\right] \sin\theta \, d\theta} - \frac{\left(\int_0^\pi P_2(\cos\theta) \exp\left[c_2 P_2(\cos\theta) + c_4 P_4(\cos\theta)\right] \sin\theta \, d\theta\right)^2}{\left(\int_0^\pi \exp\left[c_2 P_2(\cos\theta) + c_4 P_4(\cos\theta)\right] \sin\theta \, d\theta\right)^2}$$

- When *Mathematica* performs numerical integration, especially when the integrand is oscillatory (sin, cos), it will fail to reach the level of accuracy it desires and so will report "error" messages. Here, we are not too concerned with very high accuracy and so it is convenient to turn these messages off using

```
Off[NIntegrate::ncvb];
Off[NIntegrate::slwcon];
```

  It is good practice to turn them back on when you are done:

```
On[NIntegrate::ncvb];
On[NIntegrate::slwcon];
```

- Given `S2, S4` you will need to define the objective function, its derivatives, and its gradient:

```
f[{c2_, c4_}] := (S2 - f1[{c2, c4}])^2
      + (S4 - f2[{c2, c4}])^2
d1f[{c2_, c4_}] := -2(S2 - f1[{c2, c4}])d1f1[{c2, c4}]
      -2(S4 - f2[{c2, c4}])d1f2[{c2, c4}]
d2f[{c2_, c4_}] := ...
df[{c2_, c4_}] := ...
```

- You will need modify the code for the `InitInt[ ]` algorithm to avoid computing the derivative of the passed function (assume that the direction passed is indeed a descent direction):

```
InitInt[h_, t0_, α_] := Module[{k = 0, L = {t0}, t, αloc = α},
  (* ensure sufficiently small step *)
  While[h[t0+αloc] >= h[t0], αloc = αloc/2];
  While[h[L[[-1]]] > h[t0+2^k αloc ],
    AppendTo[L, t0+2^k αloc ];
    k++ ];
  AppendTo[L, t0+2^k αloc ];
  Sort[{L[[-3]],L[[-1]]}]
]
```

Chapter 4

Variational Problems

4.1 Introduction

Consider some examples first:

- **The Brachistochrone Problem:** A ball is to slide along a frictionless track from the point $(0, Y)$ to $(X, 0)$ affected only by gravity. What shape should the track be so as to minimize the time it takes?

- Let S be a surface of revolution in $\mathbb{R}^3$ and let $\vec{x}$ and $\vec{y}$ be two points on S. What path joining x to y has the shortest distance (within S)? Such a path is called a *geodesic*.

- Given two circles perpendicular to the x-axis at, say $x = a$ and $x = b$ of radii r_a and r_b, what curve $y = f(x)$ is such that when revolved about the x-axis, the resulting surface has minimal surface area, subject to the surface meeting up with the two circles?

- More generally, if we have a "distorted wire ring" what surface, meeting up with the ring, will have the minimal surface area?

- What shape will a cable, hung between two supports, make when the only force acting on it is due to its own weight?

The feature that all these problems have in common is that the minimizer is not a vector in $\mathbb{R}^n$ but rather a function within some restricted class of functions. For the first example, we seek $y = f(x)$ with $f(0) = Y$, $f(X) = 0$, and we might require that $f \in C^k([0, X])$ (for $k = 1$ or 2). For the second example we seek a parameterized curve $\vec{r}(t) = (x(t), y(t), z(t))$ such that $\vec{r}(t) \in S$ for all t, and $\vec{r}(0) = \vec{x}$, $\vec{r}(T) = \vec{y}$.

Before seeking a solution to any of these problems, we formulate each of them as an optimization problem. This will indicate the general form of the type of problem we are interested in.

Straight line

Perhaps the simplest variational problem we can pose is: what path between points (x_0, y_0) and (x_1, y_1) in the plane has the shortest length? If we assume that $x_0 \neq x_1$, and we make the, quite reasonable, assumption that the curve will be the graph of a function, $y = y(x)$, then the length of the path is calculated by integrating $\sqrt{1 + (y')^2}$ and we obtain the problem:

$$\text{minimize } \int_{x_0}^{x_1} \sqrt{1 + (y')^2}\, dx, \quad \text{subject to} \quad (y(0), y(L)) = (y_0, y_1). \tag{4.1}$$

EXERCISE 4.1. Prove that if $\vec{r}(t) = (x(t), y(t))$ is a curve from (x_0, y_0) to (x_1, y_1) ($x_1 > x_2$, say) for which the trace is not the graph of a function (fails the "vertical line test") then $\vec{r}(t)$ cannot minimize arc-length between (x_0, y_0) and (x_1, y_1). This is more difficult than it might appear. First show that if there are $t_1 < t_2$ such that $x(t_1) = x(t_2)$ then the optimal path must include the vertical line from $(x(t_1), y(t_1))$ to $(x(t_2), y(t_2))$. Then show that such a curve cannot be optimal by modifying it to omit the vertical segment and be of shorter length.

Geodesics on a cylinder

A generalization of the problem of finding the shortest path between two points in the plane (or in space) is that of finding the shortest path between two points on a surface, always remaining on the surface. A particular example of this is to consider a cylinder, C, for example $x^2+y^2 = 1$. Without loss of generality, let $P = (1, 0, 0) \in C$ and let Q be another point on C. If we use cylindrical coordinates $(r, \theta, z) = (1, \theta, z)$, $P = (1, 0, 0)$ and $Q = (1, \theta_1, z_1)$ we can assume, without loss of generality, that $0 \leq \theta_1 \leq \pi$, and once again it is reasonable to expect the minimizing curve to be *monotonic* in θ, that is, z can be expressed as a function of θ, $z(\theta)$. (A proof of this result is equivalent to Exercise 4.1.) The problem then becomes identical to (4.1):

$$\text{minimize } \int_{0}^{\theta_1} \sqrt{1 + (z')^2}\, d\theta, \quad \text{subject to} \quad (z(0), z(\theta_1)) = (0, z_1). \tag{4.2}$$

The Brachistochrone problem

In common with the previous examples, we assume that the optimal path will be monotonic in x, that the curve can be given as the graph of a function $y(x)$. When the bead is at position x, let its speed be given by $v(x)$. Then, over an interval Δx the bead will move a

distance of (approximately) Δs in time $\Delta s/v(x)$. On the other hand, the distance it moves is given by $\Delta s \approx \sqrt{1+(y'(x))^2}\Delta x$ and so the time it takes to move Δx is (approximately) $\sqrt{1+(y'(x))^2}/v(x)\ \Delta x$. Summing over all Δx intervals we obtain a Riemann sum, and then, in the limit, the total time the bead takes to travel along the curve $y(x)$ from $(0,Y)$ to $(X,0)$ is

$$\int_0^X \frac{\sqrt{1+(y'(x))^2}}{v(x)}\,dx. \tag{4.3}$$

Now we must determine $v(x)$ based on the fact that the only force acting on the bead is due to gravity. We use conservation of energy, and it is convenient to take "zero" potential energy on the x-axis. At $(0,Y)$ the bead is at rest and so has zero kinetic energy, and has potential energy of mgY (where m is its mass, and g is the constant acceleration due to gravity). At any point $(x,y(x))$ along the curve, its kinetic energy is $\frac{1}{2}mv^2$ and its potential energy is mgy. Conservation then implies

$$\frac{1}{2}mv(x)^2 + mgy = mgY \quad \Rightarrow \quad v = \sqrt{2g(Y-y)}.$$

Plugging this into (4.3) yields

$$\int_0^X \sqrt{\frac{1+(y'(x))^2}{2g(Y-y)}}\,dx.$$

It is clear that the minimizing function y will be the same if we "choose" $2g=1$. We obtain:

$$\text{minimize} \int_0^X \sqrt{\frac{1+(y'(x))^2}{Y-y}}\,dx \quad \text{subject to} \quad (y(0),y(X)) = (Y,0). \tag{4.4}$$

Notice that, unlike equations (4.1) and (4.2), the integrand in (4.4) depends on both $y(x)$ and $y'(x)$. The same is true in (4.5) below.

Minimal surface of revolution

Consider finding a function $u(x)$ with $u(a)=A$, and $u(b)=B$ such that the surface obtained by revolving $u(x)$, $a \le x \le b$, about the x-axis has minimal surface area, among all such functions. From first year calculus, we thus obtain

$$\text{minimize} \int_a^b u(x)\sqrt{1+(u'(x))^2}\,dx \quad \text{subject to} \quad (u(a),u(b)) = (A,B). \tag{4.5}$$

Hanging cable

Another feature of the examples thus far is that the constraints have all been in the form of *boundary conditions*. Suppose we have a uniform density cable of length L which is to be hung

between equal-height supports distance $2D$ apart. Physically, the cable takes the form which minimizes the total potential energy. Let the resulting form be given by $u(x)$, $-D \leq x \leq D$. If the density of the cable is $\rho\ kg/m$, then the potential energy of a segment corresponding to Δx is $mgu(x) \approx \rho\sqrt{1+(u'(x))^2}\Delta x\, g\, u(x)$; the total potential energy of the cable is then given by

$$\int_{-D}^{D} \rho\, g\, u(x)\sqrt{1+(u'(x))^2}\, dx.$$

Apart from the boundary conditions $u(-D) = u(D) = 0$, say, we also require the *length* to be L. Since the minimal function will be independent of ρ or g, we obtain

$$\text{minimize } \int_{-D}^{D} u(x)\sqrt{1+(u'(x))^2}\, dx \qquad \text{subject to} \quad u(-D) = u(D) = 0, \text{ and} \tag{4.6}$$

$$\int_{-D}^{D} \sqrt{1+(u'(x))^2}\, dx = L. \tag{4.7}$$

4.2 Formulation and the Euler-Lagrange equation

Motivated by the form of the resulting optimization problems (equations (4.1, 4.2, 4.4, 4.5, 4.6)), we introduce the general formulation of a variational problem, and we will now see why these problems are called *variational*. In all the examples presented, the optimal function was a function of *one* variable. We present the theory for functions of several variables, though we will restrict our examples to single variable cases.

We seek the minimum of

$$I(u) = \int_{\Omega} F(\vec{x}, u(\vec{x}), \nabla u(\vec{x}))\, d\vec{x}$$

where $\Omega \subset \mathbb{R}^n$, $u : \Omega \to \mathbb{R}$ is differentiable, and there is some constraint imposed on the class of admissible u.

The *integrand* is called the Lagrangian,

$$F : \Omega \times \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}.$$

When the problem is one-dimensional we have $\Omega = (a, b) \subset \mathbb{R}$,

$$I(u) = \int_{a}^{b} F(x, u(x), u'(x))\, dx.$$

Each problem comes with its constraints, the form of which vary from problem to problem. We have seen examples with fixed, one-dimensional, boundary conditions, and an integral constraint.

We use the variables $(\vec{x}, \lambda, \vec{\xi}) \in \Omega \times \mathbb{R} \times \mathbb{R}^n$ for the Lagrangian: $F(\vec{x}, \lambda, \vec{\xi})$. (That is: λ is the (scalar) variable corresponding to $u(\vec{x}) \in \mathbb{R}$; and $\vec{\xi}$ is the (vector valued) variable corresponding to $\nabla u(x) \in \mathbb{R}^n$). We assume F is differentiable with respect to $\vec{x}$ and twice differentiable with respect to both λ and $\vec{\xi}$.

Examples: The Lagrangians corresponding to the examples presented in Section 4.1 are:

Equations (4.1) and (4.2):	$F(x, \lambda, \xi) = F(\xi) = \sqrt{1 + \xi^2}$
Equation (4.4):	$F(x, \lambda, \xi) = F(\lambda, \xi) = \sqrt{\dfrac{1 + \xi^2}{Y - \lambda}}$
Equation (4.5) and (4.6):	$F(x, \lambda, \xi) = F(\lambda, \xi) = \lambda\sqrt{1 + \xi^2}$

The Euler-Lagrange equation: (boundary condition case)

THEOREM 4.2. *Let $\Omega \in \mathbb{R}^n$ be a bounded open set with piece-wise smooth boundary $\partial\Omega$. Consider the variational problem: minimize $I(u) = \displaystyle\int_\Omega F(\vec{x}, u(\vec{x}), \nabla u(\vec{x}))\, d\vec{x}$, with $u = u_0$ on $\partial\Omega$.*

1. *If u is an optimal solution, then u is also a solution to (E-L):*

$$\begin{aligned} \operatorname{div}\big(F_{\vec{\xi}}(\vec{x}, u(\vec{x}), \nabla u(\vec{x}))\big) &= F_\lambda(\vec{x}, u(\vec{x}), \nabla u(\vec{x})) && \text{in } \Omega \\ u &= u_0 && \text{on } \partial\Omega. \end{aligned}$$

2. *(Converse in the convex case.) Suppose that for each fixed $\vec{x} \in \Omega$, F is a convex function of λ and $\vec{\xi}$. If u satisfies (E-L), then u is an optimal solution to the variational problem.*

3. *(Converse in the strictly convex case.) Suppose that for each fixed $\vec{x} \in \Omega$, F is a strictly convex function of λ and $\vec{\xi}$. If u satisfies (E-L), then u is the unique optimal solution to the variational problem.*

Before proving Theorem 4.2, we explain the equation (E-L): $\operatorname{div}\big(F_{\vec{\xi}}(\vec{x}, u(\vec{x}), \nabla u(\vec{x}))\big)$ means first take the gradient of $F(\vec{x}, \lambda, \vec{\xi})$ with respect to $\vec{\xi}$ and then evaluate at $\lambda = u(\vec{x})$, $\vec{\xi} = \nabla u(\vec{x})$ – the result is a vector-valued function of $\vec{x}$. Now take the divergence of this, with respect to $\vec{x}$.

For example, suppose $\vec{x} = [x, y]^T$, $u(\vec{x}) = x^2 + y^2$ and $F(\vec{x}, \lambda, \vec{\xi}) = \lambda x \xi_1^2 - \xi_2^3$. Then $\nabla u(\vec{x}) = [2x, 2y]^T$ and

$$\begin{aligned} F_{\vec{\xi}}(\vec{x}, \lambda, \vec{\xi}) &= \big[2\lambda x \xi_1, -3\xi_2^2\big]^T; \\ F_{\vec{\xi}}(\vec{x}, u(\vec{x}), \nabla u(\vec{x})) &= \big[2(x^2 + y^2)x(2x), -3(2y)^2\big]^T; \\ \operatorname{div}\big(F_{\vec{\xi}}(\vec{x}, u(\vec{x}), \nabla u(\vec{x}))\big) &= \big(8x(2x^2 + y^2)\big) + (-24y) \end{aligned}$$

The right hand side of (E-L) in this example is simply

$$F_\lambda(\vec{x}, \lambda, \vec{\xi}) = x\xi_1^2$$
$$F_\lambda(\vec{x}, u(\vec{x}), \nabla u(\vec{x})) = x(2x)^2.$$

In the one-dimensional case, (E-L) is simply a second order ODE in the variable x, with boundary conditions:

$$\frac{d}{dx}\big(F_\xi(x, u(x), u'(x)\big) = F_\lambda(x, u(x), u'(x)) \qquad x \in (a, b)$$
$$u(a) = A,\ u(b) = B.$$

Since (E-L) is a necessary condition, if an optimal solution exists, it must be among the solutions to this ODE.

There is a slightly stronger statement of the Euler-Legrange theorem, which we won't prove here. It states that, provided there is a unique solution to (E-L), then convexity in just the $\vec{\xi}$ variable is sufficient to conclude the solution is optimal.

THEOREM 4.3. *If u is the unique solution to (E-L) and F is convex with respect to the variable $\vec{\xi}$ for each fixed $\vec{x} \in \Omega$ and $\lambda \in \mathbb{R}$, then u is an optimal solution of the variational problem.*

Proof of Theorem 4.2: one-dimensional case. Suppose that $u(x)$ minimizes $I(u)$ among all functions satisfying $u(a) = A$, $u(b) = B$. If $\varphi(x)$ is any differentiable function $\varphi : (a, b) \to \mathbb{R}$ with $\varphi(a) = \varphi(b) = 0$, then for $-\varepsilon < t < \varepsilon$, $v_t(x) = u(x) + t\varphi(x)$ is again an admissible function (certainly $v_t(a) = u(a) = A$, $v_t(b) = u(b) = B$ and v_t is differentiable in $x \in (a, b)$). See Figure 4.1. We call $v_t(x)$ a *variation* of $u(x) = v_0(x)$. Since u is optimal for I, we must have

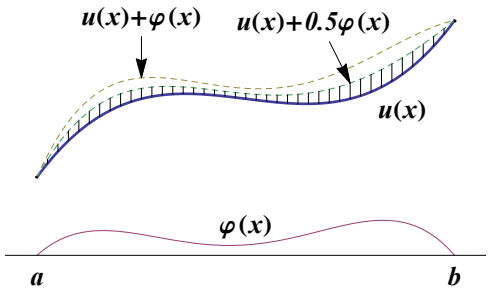


Figure 4.1: A variation of $u(x)$.

$I(u) = I(v_0) \le I(v_t)$ for all t.

Now define the function of one variable

$$g(t) = I(v_t) = \int_a^b F(x, u(x) + t\varphi(x), u'(x) + t\varphi'(x))\, dx.$$

The statement that $u = v_0$ is *minimal* means that $g'(0)$ must equal zero. So we differentiate (the requirements of differentiability on F justify moving any derivatives in and out of the integral):

$$0 = \frac{d}{dt}\Big|_{t=0} I(v_t) = \int_a^b \big(F_\lambda(x, u(x), u'(x))\varphi(x) + F_\xi(x, u(x), u'(x))\varphi'(x)\big)\, dx.$$

We integrate by parts, in the second term only, to move the derivative off φ and onto F_ξ, making use of the fact that the boundary terms involve $\varphi(a) = \varphi(b) = 0$. We obtain

$$0 = \int_a^b \Big(F_\lambda(x, u(x), u'(x)) - \frac{d}{dx}F_\xi(x, u(x), u'(x))\Big)\varphi(x)\, dx. \tag{4.8}$$

Now since this must hold for every φ (with $\varphi(a) = \varphi(b) = 0$), this implies the (E-L) equation, by Lemma 4.4 below. This proves part 1.

LEMMA 4.4. *If $f : (a, b) \to \mathbb{R}$ is continuous and $\displaystyle\int_a^b f(x)\varphi(x)\, dx = 0$ for all differentiable φ with $\varphi(a) = \varphi(b) = 0$, then $f(x) = 0$ for all $x \in (a, b)$.*

Proof of Lemma 4.4. Recall the definition of continuity at x_0: given any $\varepsilon > 0$ there exists $\delta > 0$ such that $|f(x) - f(x_0)| < \varepsilon$ for all $|x - x_0| < \delta$. Suppose now that there exists x_0 such that $f(x_0) \neq 0$; without loss of generality, suppose that $f(x_0) > 0$. Let $\varepsilon = \frac{1}{2}f(x_0)$ and let $\delta > 0$ be that guaranteed by the continuity of f at x_0. See Figure 4.2. Thus, on $(x_0 - \delta, x_0 + \delta)$, $|f(x) - f(x_0)| < \varepsilon = \frac{1}{2}f(x_0)$, which implies $f(x) > f(x_0) - \frac{1}{2}f(x_0) = \frac{1}{2}f(x_0)$. Let φ be a differentiable function which is 0 outside the interval $(x_0 - \delta, x_0 + \delta)$, and is strictly greater than zero inside the interval, say $\varphi(x_0) = 1$. See Figure 4.2.

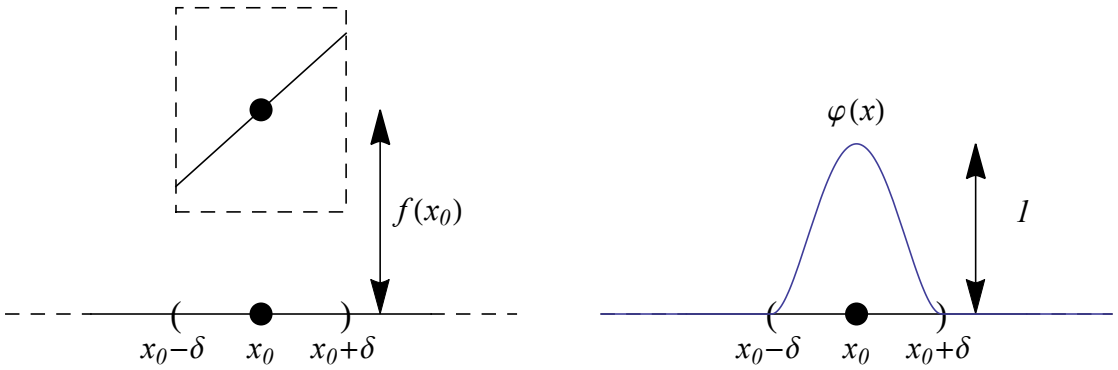


Figure 4.2: If $f(x_0) > 0$ then $f(x) > 0$ near x_0.

Now, with this φ,

$$0 = \int_a^b f(x)\varphi(x)\, dx = \int_{x_0-\delta}^{x_0+\delta} f(x)\varphi(x)\, dx \geq \frac{f(x_0)}{2}\int_{x_0-\delta}^{x_0+\delta} \varphi(x)\, dx > 0$$

which is clearly a contradiction. Thus, no such x_0 can exist and $f(x) = 0$ for all x. □

We continue with the proof of Theorem 4.2, parts 2 and 3. Suppose now that F is convex as a function of λ and ξ for each fixed $x \in (a,b)$, and suppose that u satisfies (E-L). Convexity gives (see (2.18) in Chapter 2)

$$F(x,\tilde{\lambda},\tilde{\xi}) \geq F(x,\lambda,\xi) + F_{\lambda,\xi}(x,\lambda,\xi)\big((\tilde{\lambda},\tilde{\xi}) - (\lambda,\xi)\big)$$

so for any admissible v,

$$\begin{aligned} F(x,v(x),v'(x)) \geq F(x,u(x),u'(x)) &+ F_\lambda(x,u(x),u'(x))(v(x)-u(x)) \\ &+ F_\xi(x,u(x),u'(x))(v'(x)-u'(x)). \end{aligned}$$

Thus

$$\begin{aligned} I(v) - I(u) &= \int_a^b \big(F(x,v(x),v'(x)) - F(x,u(x),u'(x))\big)\,dx \\ &\geq \int_a^b \big(F_\lambda(x,u(x),u'(x))(v(x)-u(x)) + F_\xi(x,u(x),u'(x))(v'(x)-u'(x))\big)\,dx \end{aligned}$$

Integrating the second term by parts, as above, we obtain the (E-L) equation for u (times $(v(x)-u(x))$), giving 0. Thus $I(v) \geq I(u)$ and u is an optimal solution.

For strict convexity, we can simply replace the $\geq$ by $>$ in the above argument. □

Proof of Theorem 4.2: general case. The proof proceeds in the same manner as for the one-dimensional case. We consider any function $\varphi : \Omega \subset \mathbb{R}^n \to \mathbb{R}$ where we now require $\varphi = 0$ on all of $\partial\Omega$. The variation $v_t(\vec{x}) = u(\vec{x}) + t\varphi(\vec{x})$ is as before, and we again define a function of one variable

$$g(t) = I(v_t) = \int_\Omega F(\vec{x}, u(\vec{x}) + t\varphi(\vec{x}), \nabla u(\vec{x}) + t\nabla\varphi(\vec{x}))\,d\vec{x};$$

since $u = v_0$ is minimal, $g'(0) = 0$, and so

$$0 = \frac{d}{dt}\Big|_{t=0} I(v_t) = \int_a^b \big(F_\lambda(\vec{x},u(\vec{x}),\nabla u(\vec{x}))\varphi(\vec{x}) + \nabla_{\vec{\xi}} F(\vec{x},u(\vec{x}),\nabla u(\vec{x}))\cdot\nabla\varphi(\vec{x})\big)\,d\vec{x}. \quad (4.9)$$

Applying the Divergence Theorem (B.7.10) to the vector field $\varphi(\vec{x})\nabla_{\vec{\xi}} F(\vec{x},u(\vec{x}),\nabla u(\vec{x}))$ (see Exercise 4.5),

$$\int_\Omega \nabla_{\vec{\xi}} F\cdot\nabla\varphi\,d\vec{x} = -\int_\Omega \varphi\,\mathrm{div}(\nabla_{\vec{\xi}} F)\,d\vec{x} + \int_{\partial\Omega}\varphi\nabla_{\vec{\xi}} F\cdot d\vec{A}.$$

Using the fact that $\varphi = 0$ on $\partial\Omega$ and inserting the result into (4.9) we obtain

$$0 = \int_\Omega \Big(F_\lambda(\vec{x},u(\vec{x}),\nabla u(\vec{x})) - \mathrm{div}(F_{\vec{\xi}}(\vec{x},u(\vec{x}),\nabla u(\vec{x}))\Big)\varphi(\vec{x})\,d\vec{x}.$$

By Exercise 4.6, this implies (E-L). □

EXERCISE 4.5. If φ is a scalar valued function, and $\vec{G}$ is a vector field, apply the Divergence Theorem (B.7.10) to the vector field $\varphi\vec{G}$ to prove

$$\int_\Omega \big(\vec{G}\cdot\nabla\varphi + \varphi(\operatorname{div}\vec{G})\big)\,dV = \int_{\partial\Omega}\varphi\vec{G}\cdot\,d\vec{A}.$$

EXERCISE 4.6. Mimicking the proof of Lemma 4.4, prove:

If $F:\Omega\subset\mathbb{R}^n\to\mathbb{R}$ is continuous, and

$$\int_\Omega F(\vec{x})\varphi(\vec{x})\,d\vec{x} = 0$$

for all continuous $\varphi:\Omega\to\mathbb{R}$ with $\varphi = 0$ on $\partial\Omega$, then $F(\vec{x}) = 0$ for all $\vec{x}\in\Omega$.

From this point on, unless stated otherwise, we will deal only with *one-dimensional* problems, $F:(a,b)\times\mathbb{R}\times\mathbb{R}\to\mathbb{R}$. In many applications, the Lagrangian is independent of some of the variables (we will see this case by case).

Special Case I: When $F(x,\lambda,\xi) = F(\xi)$ only (is independent of x and λ), (E-L) becomes $\dfrac{d}{dx}\big(F_\xi(u'(x))\big) = 0$. If we expand this derivative

$$F_{\xi\xi}(u'(x))u''(x) = 0 \tag{4.10}$$

From this we obtain: if $u(x)$ is an optimal solution, either,

1. there exists x_0 such that $F_{\xi\xi}(u'(x_0))\neq 0$: in this case (Exercise 4.7) it can be shown that $u''(x) = 0$ for all $x\in(a,b)$. Then $u(x)$ is a linear function,

   $$u = \alpha x + \beta,$$

   and satisfying the boundary conditions, u is simply the equation of the straight line passing through (a,A) and (b,B). Or,

2. $F_{\xi\xi}$ is identically zero on the interval J with endpoints $\min_{(a,b)} u'(x)$ and $\max_{(a,b)} u'(x)$. In this case $F(\xi) = \alpha\xi+\beta$, and then

   $$I(u) = \int_a^b (\alpha u'(x)+\beta)\,dx = \alpha(B-A)+\beta(b-a).$$

   But *any* admissible u gives this result, and so the functional I is actually constant over all admissible functions.

EXERCISE 4.7. Let $u\in C^2(a,b)$ and suppose that there exists x_0 such that $F_{\xi\xi}(u'(x_0))\neq 0$. Prove that (4.10) implies that $u''(x) = 0$ for all $x\in(a,b)$. (Hint: first argue that there is an interval $(a_0,b_0)\subset(a,b)$ containing x_0 on which $F_{\xi\xi}(u'(x))\neq 0$, and take it to be the largest such interval, so either $F(u'(a_0)) = 0$ or $a_0 = a$. Now prove that $F(u'(a_0)) = F(u'(x_0))$, so that $a_0 = a$. Complete the argument to prove the result.)

Special Case II: Beltrami's identity. Suppose that $F(x, \lambda, \xi) = F(\lambda, \xi)$ is independent of x. We compute the derivative of $F(x, u(x), u'(x))$ with respect to x:

$$\frac{dF}{dx} = \frac{\partial F}{\partial x} + \frac{\partial F}{\partial \lambda}u' + \frac{\partial F}{\partial \xi}u'' = \frac{\partial F}{\partial \lambda}u' + \frac{\partial F}{\partial \xi}u''$$

when $F(x, \lambda, \xi) = F(\lambda, \xi)$. The Euler-Lagrange equation says

$$\frac{d}{dx}\Big(\frac{\partial F}{\partial \xi}\Big) = \frac{\partial F}{\partial \lambda} \quad \Rightarrow \quad u'\frac{d}{dx}\Big(\frac{\partial F}{\partial \xi}\Big) = u'\frac{\partial F}{\partial \lambda} = \frac{dF}{dx} - \frac{\partial F}{\partial \xi}u''$$

from the above. We rearrange this and notice that (by the product rule) we have the following:

$$0 = \frac{dF}{dx} - u'\frac{d}{dx}\Big(\frac{\partial F}{\partial \xi}\Big) - u''\frac{\partial F}{\partial \xi} = \frac{d}{dx}\Big(F - u'\frac{\partial F}{\partial \xi}\Big).$$

Thus (this is Beltrami's identity)

$$F - u'\frac{\partial F}{\partial \xi} = C = \text{ constant.} \tag{4.11}$$

REMARK. What we have shown is that if u is a solution to (E-L), then it is also a solution to Beltrami's Identity. The converse is *not* necessarily true. One might find solutions to (4.11) which do not solve (E-L). However, among all solutions to (4.11) must be the solutions to (E-L).

4.3 Examples

Geodesic on a cylinder

See (4.2), page 102. The path of minimal length joining $P = (1, 0, 0)$ to $Q = (1, \theta_0, z_0)$ (in cylindrical coordinates) can be parameterized as

$$\sigma(\theta) = (\cos\theta, \sin\theta, z(\theta)), \qquad 0 < \theta < \theta_0.$$

So, we seek $z(\theta)$ satisfying (4.2). The Lagrangian is $F(\xi) = \sqrt{1 + \xi^2}$, which is independent of θ and λ. Since $F''(\xi) = 1/(1 + \xi^2)^{3/2} > 0$, if we find a solution to (E-L) then it has to be the unique minimizer. According to what we computed earlier for $F = F(\xi)$ only, $z(\theta) = \alpha\theta + \beta$ and the boundary conditions imply $z(\theta) = \dfrac{z_0}{\theta_0}\theta$.

Solution to the Brachistochrone Problem

See (4.4), page 103. The time-minimizing path is given by $y(x)$ satisfying (4.4). The problem is completely scalable, and so we will assume here that $Y = 1$; the bead starts at $(0, 1)$. The

Lagrangian is

$$F(x,\lambda,\xi) = \sqrt{\frac{1+\xi^2}{1-\lambda}}$$

which, we see, depends only on λ and ξ. Thus we may use Beltrami's identity, (4.11). Applying this to our problem,

$$\begin{aligned} C = F - y'\frac{\partial F}{\partial \xi}(x,y,y') &= \frac{\sqrt{1+(y')^2}}{\sqrt{1-y}} - y'\frac{y'}{\sqrt{(1-y)(1+(y')^2)}} \\ &= \frac{1}{\sqrt{(1-y)(1+(y')^2)}} \end{aligned}$$

so, with a new constant C,

$$(1+(y')^2)(1-y) = C \qquad \Rightarrow \qquad \frac{dy}{dx} = \pm\sqrt{\frac{C-(1-y)}{1-y}} = \pm\sqrt{\frac{C}{1-y}-1}. \tag{4.12}$$

(Notice that since the bead is dropping down from $y=1$, $C>0$.) This equation is separable:

$$\pm\sqrt{\frac{1-y}{C-1+y}}\,dy = dx.$$

The left hand side encourages the use of a trigonometric substitution; we set $1-y = C\sin^2(\varphi/2)$. (The choice of $\varphi/2$ is unnecessary, but results in a simpler expression for the solution.) Computing $\dfrac{dy}{d\varphi} = -C\sin(\varphi/2)\cos(\varphi/2)$, we obtain

$$\pm\sqrt{\frac{C\sin^2(\varphi/2)}{C(1-\sin^2(\varphi/2))}}\,C\sin(\varphi/2)\cos(\varphi/2)\,d\varphi = dx$$

which simplifies to

$$\frac{dx}{d\varphi} = \pm C\sin^2(\varphi/2) = \pm\frac{C}{2}(1-\cos(\varphi)) \quad \Rightarrow \quad x = \pm\frac{C}{2}(\varphi-\sin\varphi)+\tilde{C}.$$

Now $x(0) = \tilde{C} = 0$, and since $x(\varphi)$ must be an *increasing* function (for the bead to move to the right) $x'(\varphi)$ must be positive (although $x'(0)=0$); thus we see that we should take the positive sign. We obtain

$$x(\varphi) = \frac{C}{2}(\varphi-\sin\varphi), \qquad y(\varphi) = \frac{C}{2}(\cos\varphi-1)+1. \tag{4.13}$$

These are the parametric equations of a cycloid – the result of rolling a wheel of radius C "upside down" along the bottom side of the line $y=1$; the center of the wheel is always along $y = 1-C/2$, and the wheel rolls counter-clockwise. The path $(x(\varphi), y(\varphi))$ is that traced out by the point on the rim of the wheel starting at $(0,1)$. We depict this in Figures 4.4–4.7 below. First,

however, we must solve for the two degrees of freedom in these equations: we must determine a C and a final value φ_0 of φ such that $x(\varphi_0) = X$ and $y(\varphi_0) = 0$.

Understanding the solution: First, notice that $x'(\varphi) = 0$ and $y'(0) < 0$, so the optimal strategy is always to start falling straight down, which makes sense. To obtain the path which meets $(X, 0)$ we need to find the right C, but also the right final φ value. We have

$$y = 0 \qquad \Rightarrow \qquad \cos\varphi = 1 - \frac{2}{C};$$

since $-1 \le \cos\varphi \le 1$, we find that $C \ge 1$. We can solve for $\varphi = \arccos(1-2/C)$, but this only gives us one of the solutions (from the definition of $\arccos$); the other is $2\pi - \arccos(1-2/C)$. Plugging these into x, and simplifying, we find

$$X = \frac{C}{2}\arccos\Big(1 - \frac{2}{C}\Big) - \sqrt{C-1} = f_1(C), \qquad \text{and}$$
$$X = \frac{C}{2}\Big(2\pi - \arccos\Big(1 - \frac{2}{C}\Big)\Big) + \sqrt{C-1} = f_2(C).$$

One finds

- $f_1(1) = \dfrac{\pi}{2}$, and $f_1(C) \to 0$ as $C \to \infty$, with $f_1'(C) < 0$. Thus $0 < f_1(C) \le \pi/2$ and so if $0 < X \le \dfrac{\pi}{2}$, C is determined by solving $X = f_1(C)$.
- On the other hand, $f_2(1) = \dfrac{\pi}{2}$, $f_2(C) \to \infty$ as $C \to \infty$, and $f_2'(C) > 0$. Thus if $X > \dfrac{\pi}{2}$, C is determined by solving $X = f_2(C)$.

Figure 4.3 shows these functions.

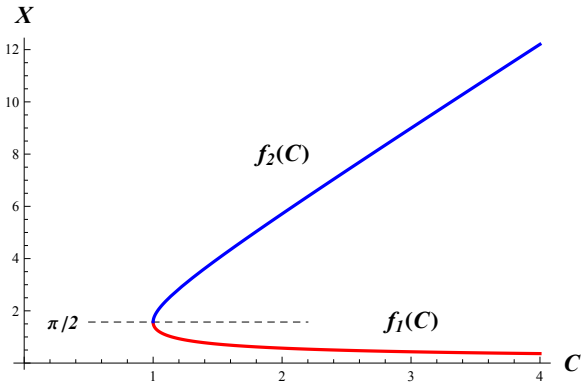


Figure 4.3: Determination of C given X.


In order to find specific solutions, it is necessary to find C given X, which we do numerically with *Mathematica*. The function we use is `FindRoot[equation, {var, init}]`,

where `equation` is that which you are trying to solve, `var` is the variable to solve for, and `init` is the initial guess from where to search. What should be taken as an initial guess is usually best determined on a case by case basis. In this application, it turns out using `C = 2` is successful for all `X`.

One remark: *Mathematica* reserves the symbol `C` for the list of undetermined coefficients in solving equations, so we must use another symbol; here we use `c`. This code returns both the C and the corresponding maximal φ_0 in the parameterization for $(x(\varphi), y(\varphi))$, $0 \le \varphi \le \varphi_0$.

```
f[X_] := Module[ {c, ϕ},
  If[0 < X <= π/2,
    (* use f1 *)
    c = c/.FindRoot[c/2(ArcCos[1-2/c]-2Sqrt[1/c-1/c^2])
        == X, {c, 2}];
    ϕ = ArcCos[1-2/c],
    (* use f2 *)
    c = c/.FindRoot[c/2(2π-ArcCos[1-2/c]+2Sqrt[1/c-1/c^2])
        == X, {c, 2}];
    ϕ = 2π - ArcCos[1-2/c]
    ];
  Chop[{c, ϕ}]
  ]
```

The command `FindRoot[ ]` produces a (potentially empty) list of replacement rules,

```
{{c -> sol1}, {c -> sol2}, ...}
```

where `sol1, soln2` are the set of roots found. In the present example, there is only one possible root, so we don't need to worry about using the correct root. The use of `c/.FindRoot[ ]` causes the replacement to happen and `c` takes on the solution found. Since roots are being found *numerically*, computational errors can lead to very small complex parts (of the order of 10^{-18}). The `Chop[ ]` command throws away such small components.

Minimal surface of revolution:

The surface of revolution about the x-axis, $a \le x \le b$, with radius A at a and radius B at b is formed by the generating function $u(x)$ which solves (4.5). The Lagrangian is thus

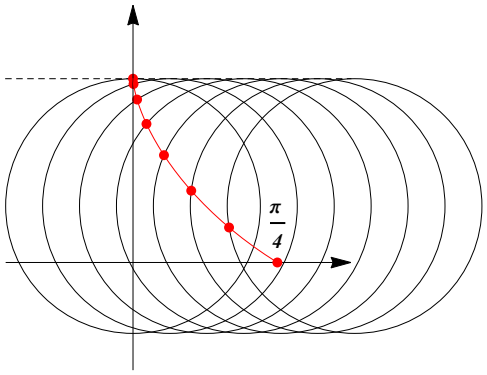


Figure 4.4: Brachistochrone solution: $X = \dfrac{\pi}{4}$, $Y = 1$.

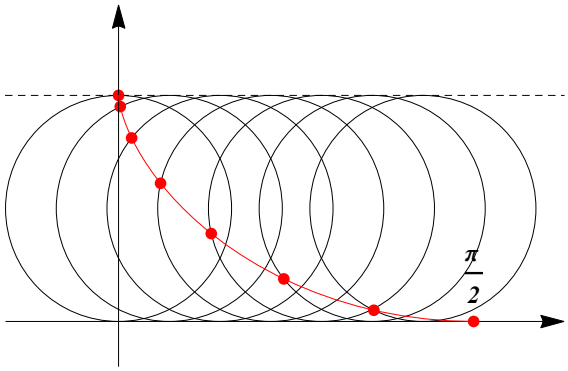


Figure 4.5: Brachistochrone solution: $X = \dfrac{\pi}{2}$, $Y = 1$.

$F(\lambda, \xi) = \lambda\sqrt{1+\xi^2}$. This function is not jointly convex in λ and ξ for

$$D^2F(\lambda,\xi) = \begin{bmatrix} 0 & \dfrac{\xi}{\sqrt{1+\xi^2}} \\ \dfrac{\xi}{\sqrt{1+\xi^2}} & \dfrac{\lambda}{(1+\xi^2)^{3/2}} \end{bmatrix}$$

which has determinant $-(y')^2/(1+(y')^2) \leq 0$. However, for fixed $\lambda > 0$, it is strictly convex in ξ. (We may assume that $u > 0$ in order to actually have a surface.) Thus, if we find there is uniqueness of solution in (E-L) we can apply Theorem 4.3 to conclude the solution is optimal.

We again make use of Beltrami's identity (4.11) to obtain

$$u\sqrt{1+(u')^2} - u'\frac{uu'}{\sqrt{1+(u')^2}} = C.$$

If $C = 0$ then we find $u = 0$ which we will not allow. Provided $u^2 \neq C^2$, the above can be

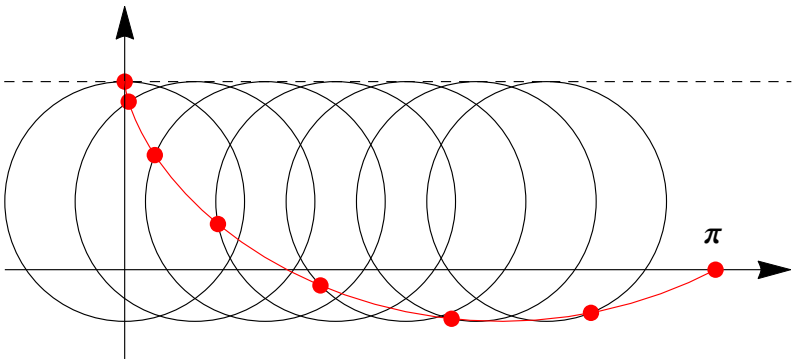


Figure 4.6: Brachistochrone solution: $X = \pi$, $Y = 1$.

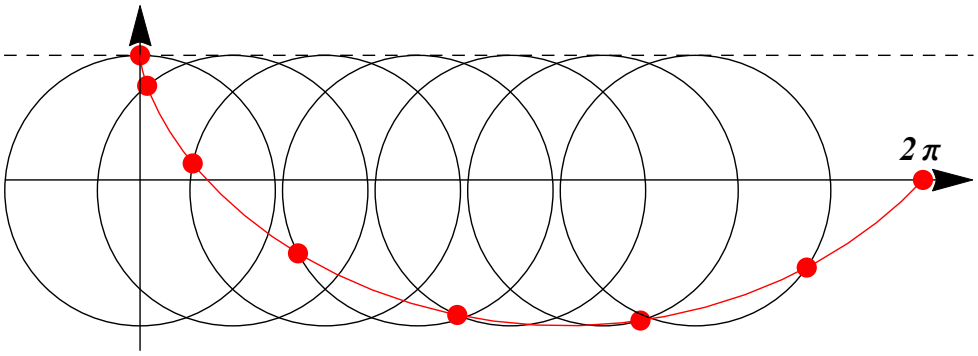


Figure 4.7: Brachistochrone solution: $X = 2\pi$, $Y = 1$.

re-written $u^2 - C^2 = C^2(u')^2$, and then,

$$\frac{u'}{\sqrt{u^2 - C^2}} = \pm\frac{1}{C} \qquad \Rightarrow \qquad \frac{du}{\sqrt{u^2 - C^2}} = \pm\frac{dx}{C}. \tag{4.14}$$

Since $\cosh^2\theta - 1 = \sinh^2\theta$, we are motivated to substitute $u = C\cosh\theta$; then $du = C\sinh\theta\, d\theta$, and we get $d\theta = \pm dx/C$, and so

$$\theta = \pm\frac{x}{C} + d \qquad \Rightarrow \qquad \operatorname{arccosh}\frac{u}{C} = \pm\frac{x}{C} + d \qquad \Rightarrow \qquad u = C\cosh\Big(\pm\frac{x}{C} + d\Big).$$

Note that we may take the positive sign (say) since $\cosh$ is an even function; taking the negative sign will simply result in d being replaced by $-d$, another constant to be determined anyway.

If, in fact, $u^2 = C^2$, then $u = C$ is constant. This is an example of a where there is a solution to Beltrami's identity which is not a solution to (E-L). To see this, we must compute (E-L) for this Lagrangian:

$$0 = \frac{d}{dx}\big(F_\xi(x,u,u')\big) - F_\lambda(x,u,u') = \frac{u(x)u''(x) - u'(x)^4 - 2u'(x) - 1}{(1+u'(x)^2)^{3/2}}$$

and it is evident that $u(x) = C$, constant, does not satisfy this.

REMARK. There is another, useful and clever, way to approach the solution to the non-linear ordinary differential equation $u^2 - C^2 = C^2(u')^2$ we solved above. We demonstrate it here, even though we have successfully solved the equation above. Starting with $C^2(u')^2 - u^2 = -C^2$, we differentiate once more to obtain

$$2C^2u'u'' - 2uu' = 0 \quad \Rightarrow \quad u' = 0, \quad \text{or} \quad C^2u'' - u = 0.$$

If $u' = 0$ then $u(x) = c$ is constant, which we discussed above. To solve $C^2u'' - u = 0$, we can consider the auxiliary polynomial $C^2r^2 - 1$, which has roots $r = \pm 1/C$, and the general solution is (see Appendix A)

$$u(x) = c_1e^{x/C} + c_2e^{-x/C}$$

This can be manipulated algebraically to look like the $\cosh$ solution above, but it takes some work. This alternative approach isn't advantageous in this case because by differentiating once more, we are introducing yet another unknown constant (notice that there are three constants c_1, c_2, and C now) and it takes extra work to determine these. We will see, later, that sometimes this approach *is advantageous* – what makes it so is when the constant C disappears from the equation. See Example 4.9 below. □

Now for the boundary conditions. We need $u(a) = A$ and $u(b) = B$. Without loss of generality, we can fix one of these to be $a = 0$, $A = 1$. Thus we must have

$$C\cosh d = 1, \qquad \frac{B}{C} = \cosh\Big(\frac{b}{C} + d\Big). \tag{4.15}$$

Thinking of these as functions of C and d, we need the level-0 sets of $C\cosh d - 1$ and $\cosh(b/C + d) - B/C$ to have intersection.

Before investigating when such intersection can occur, it is worth thinking physically: the ring at $a = 0$ has radius 1; the ring at b has radius B. If b is very large, it is clear that the surface of smallest area which has these circles as boundary is actually a disconnected surface comprised of two disks, with area $\pi(1+B^2)$; indeed, it can be thought of as the limit of a connected surface which has a very thin tube joining the two disks together. See Figure 4.8. When b is small, it is believable that we can form a surface joining the rings with area less than $\pi(1 + B^2)$. By continuity, there must be some value of b for which these two possibilities have the same area.

Equation (4.15) cannot be solved analytically. It is also more complicated that it might appear. To begin with, let's consider the case when the two rings are of the same size, say $A = B = 1$, and are situated at $a = -\alpha$, $b = \alpha$ (this enables easier use of symmetry). Instead of (4.15), we then have

$$u(-\alpha) = 1 = C\cosh\Big(\frac{-\alpha}{C} + d\Big) = C\cosh\Big(\frac{\alpha}{C} + d\Big) = u(\alpha). \tag{4.16}$$

With $C \neq 0$, taking $\cosh^{-1}$ we obtain

$$\frac{\alpha}{C} + d = \pm\Big(\frac{-\alpha}{C} + d\Big) \qquad \Rightarrow \qquad \text{either } \frac{\alpha}{C} = 0, \text{ or } d = 0. \tag{4.17}$$

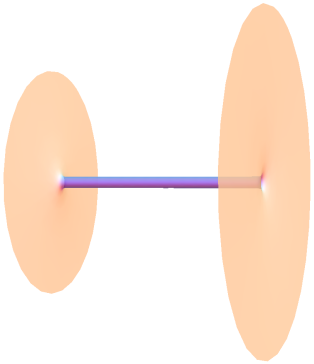


Figure 4.8: Approaching a minimal surface when the boundary rings are far apart.

The first is not relevant (the rings are both at $x = 0$), so $d = 0$ and

$$u(x) = C \cosh \frac{x}{C}. \tag{4.18}$$

Of course we still need $u(\alpha) = 1 = C \cosh \frac{\alpha}{C}$. Figure 4.9 shows a plot of this equation's solution set.

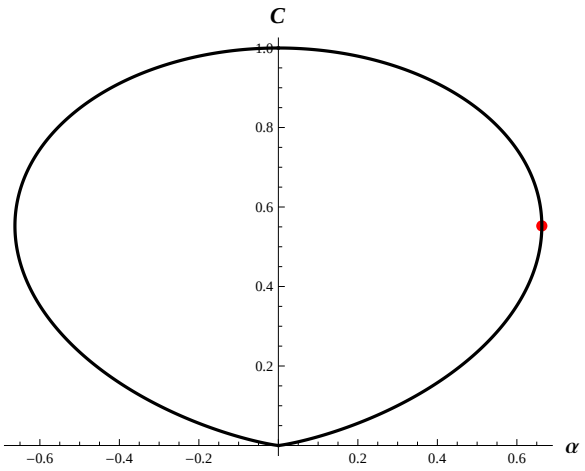


Figure 4.9: The solution set to $C \cosh \frac{\alpha}{C} = 1$.

The maximum value of α for which there exists a solution C is (approximately) $\alpha \approx 0.6627$, with $C \approx 0.5524$. If $\alpha \lesssim 0.6627$ we see that there are *two* solutions to (E-L) of the form (4.18). There is also the limiting solution, not found by (E-L), consisting of the two disjoint disks. Which is optimal, for a given separation constant α, is not immediately clear. For each α we can compute the corresponding surface areas of the various possible solutions:

1. The two-disk (non)-solution has surface area 2π;

2. When $\alpha \lesssim 0.6627$ there are C_1 and C_2, giving solutions $u_j(x) = C_j \cosh \dfrac{x}{C_j}$ from which we compute the surface areas

$$\int_{-\alpha}^{\alpha} 2\pi u_j(x)\sqrt{1 + (u_j'(x))^2}\, dx.$$

When there are two C_j we obtain surfaces of revolution, one of which is "close to" a cylinder, the other of which is "pinched". For example, when $\alpha = 0.5$, the two surfaces are shown in Figure 4.10. Which has smaller surface area? The surface areas of all potential solutions are plotted in Figure 4.11.

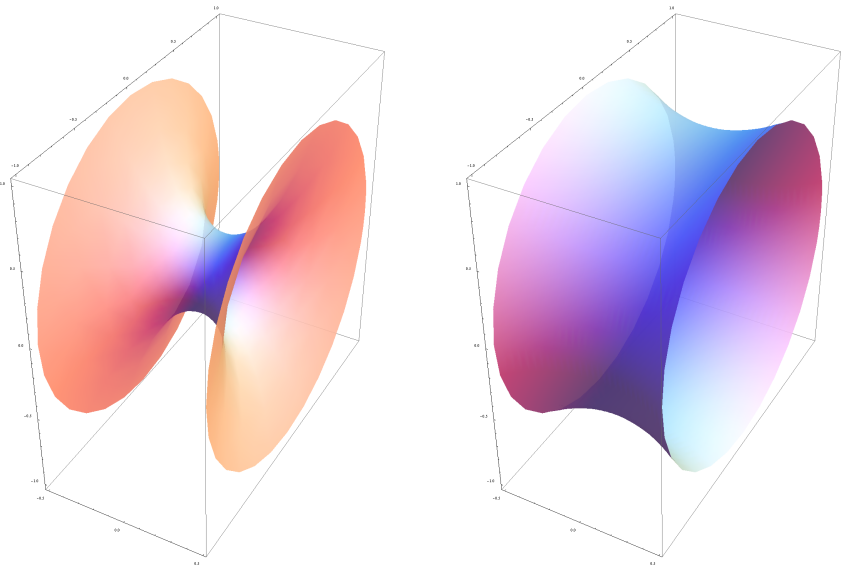


Figure 4.10: The two surfaces of revolution corresponding to the two solutions to (E-L) when $\alpha = 0.5$.

Conclusion: Despite the multiple possibilities, we see that the optimal solution is:

1. when $\alpha \lesssim 0.5277$, the approximate cylinder whose generating function is given by (4.18) with C being the larger of the two solutions found from Figure 4.9;
2. when $\alpha \gtrsim 0.5277$, the disconnected surface consisting of two disks;
3. when $\alpha \approx 0.5277$, the approximate cylinder ((4.18) with $C \approx 0.8255$) and the two-disk solution are *both* optimal, with the same surface area.

The "pinched" solutions are, of course, places where the functional $I(u)$ has zero derivative, $I'(u) = 0$. Without further analysis, we can't be sure whether this is a local minimum, a local maximum, or neither. Figure 4.12 shows the optimal approximate cylinder and the two-disk solution for $\alpha \approx 0.5277$, when the surface areas are equal.

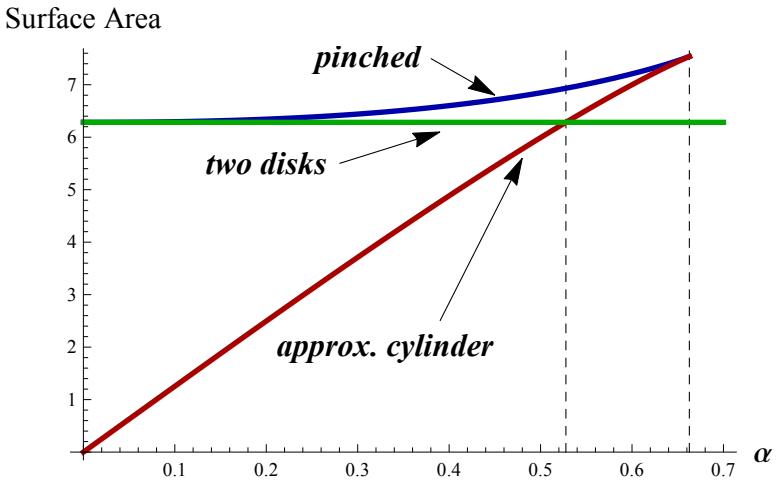


Figure 4.11: Surface areas of (potential) solutions to the minimal surface of revolution problem.

The more general case requires seeing whether or not (4.15) has solutions C and d, and in general when it does, there will be two solutions, and so one must test to see which is minimal. These still need to be tested against the two disk solution for it may turn out to be smaller still.

In the case that b is too large for there to be solutions to (E-L), our formulation of the problem cannot find the optimal two-disk solution. We can approximate the optimal solution as closely as we wish by taking a very narrow tube joining two "almost-disks," but (E-L) will not find these functions since they are not ever optimal.

4.4 Natural Boundary Conditions

We consider now the case where there is only one fixed boundary condition; the other end is "free." Consider minimizing

$$I(u) = \int_a^b F(x, u(x), u'(x))\, dx, \qquad u(a) = A.$$

Since we have an extra degree of freedom, it's likely that (E-L) plus $u(a) = A$ are not enough conditions to pin the optimal solution down. This is indeed the case as we now see.

In the derivation of the (E-L) equation, we used the boundary conditions at the step of integration by parts – we "varied" the optimal solution $u(x)$ by a function $\varphi(x)$ which was *zero* at $x = a$ and $x = b$. Now we have no reason to impose $\varphi(b) = 0$ and we obtain

$$\begin{aligned} 0 = \int_a^b \Big(F_\lambda(x, u(x), u'(x)) - \frac{d}{dx} F_\xi(x, u(x), u'(x))\Big)\varphi(x)\, dx \\ + F_\xi(b, u(b), u'(b))\varphi(b). \end{aligned} \tag{4.19}$$

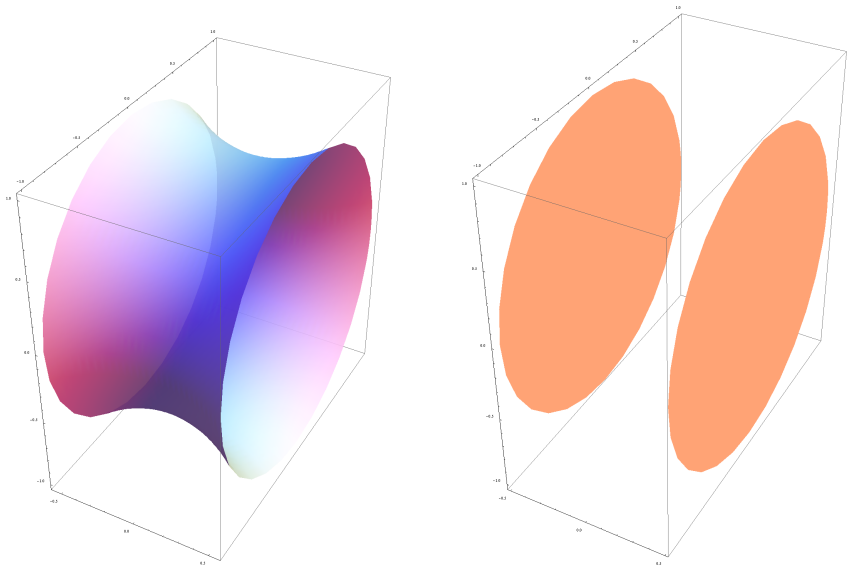


Figure 4.12: The optimal approximate cylinder and the two-disk solution when $\alpha \approx 0.5277$. The surface areas of the two solutions are equal.

This must hold for all differentiable φ with $\varphi(a) = 0$. Among all such allowable φ are the ones which *do have* $\varphi(b) = 0$, that is it must also be true that

$$0 = \int_a^b \Big(F_\lambda(x, u(x), u'(x)) - \frac{d}{dx} F_\xi(x, u(x), u'(x)) \Big) \varphi(x)\, dx \tag{4.20}$$

for all φ which are zero at both $x = a$ and $x = b$. But just as in the proof of (E-L), (4.20) implies that the original (E-L) equation still holds. Knowing this, we return to (4.19) but with $F_\lambda(x, u(x), u'(x)) - \frac{d}{dx} F_\xi(x, u(x), u'(x)) = 0$. Thus $F_\xi(b, u(b), u'(b))\varphi(b) = 0$; since $\varphi(b)$ is arbitrary, we obtain the extra condition which must be satisfied

$$F_\xi(b, u(b), u'(b)) = 0.$$

This is the "natural" boundary condition at b. The same could be applied to the endpoint $x = a$.

EXAMPLE 4.8. Consider a variation on the Brachistochrone problem: starting at $(0, Y)$, what is the path a frictionless bead should follow, influenced only by gravity, so as to reach $x = X$ in the shortest amount of time? There is no restriction on what y should equal at $x = X$.

Again, we will take $Y = 1$. The set-up of the problem is just as in 4.3, but we now have the new natural boundary condition $F_\xi(X, y(X), y'(X)) = 0$. That is, at $x = X$,

$$\frac{y'}{\sqrt{2g(1-y)(1+(y')^2)}} = 0 \qquad \Rightarrow \qquad y'(X) = 0.$$

This means the bead reaches X traveling horizontally. By (4.12), at $x = X$, we have

$$\frac{dy}{dx} = -\sqrt{\frac{C}{1-y} - 1} = 0 \qquad \Rightarrow \qquad y(X) = 1 - C.$$

This time, we can find the unknown constants, namely C and the final value φ_0 of φ, more easily. At $\varphi = \varphi_0$ it must hold that $y'(\varphi_0) = 0$. By (4.13), we have

$$0 = y'(\varphi_0) = -\frac{C}{2}\sin\varphi_0 \qquad \Rightarrow \qquad \varphi_0 = \pi,$$

and then

$$X = x(\varphi_0) = x(\pi) = \frac{C}{2}(\pi - \sin\pi) = \frac{C}{2}\pi \qquad \Rightarrow \qquad C = \frac{2X}{\pi}.$$

Thus, $x(\varphi) = \dfrac{X}{\pi}(\varphi - \sin\varphi)$ and $y(\varphi) = \dfrac{X}{\pi}(\cos\varphi - 1) + 1$. □

There is no reason why *both* boundary conditions couldn't be free as in the following example. In this case we would obtain the natural boundary conditions on the Euler-Lagrange equation:

$$F_\xi(a, u(a), u'(a)) = 0, \qquad F_\xi(b, u(b), u'(b)) = 0.$$

EXAMPLE 4.9. Find the function u which minimizes

$$I(u) = \frac{1}{2}\int_0^{\log 2} \left((u'(x) - 1)^2 + u(x)^2\right) dx.$$

Intuitively, we are looking for a function which is of slope as close to 1 as possible, while simultaneously having value as close to zero as possible. Both boundary conditions are free. We have $F(x, \lambda, \xi) = (\xi - 1)^2 + \lambda^2$ and we must solve

$$\begin{aligned} F(x, u, u') - u'F_\xi(x, u, u') = C, \qquad & \text{i.e.} \qquad (u' - 1)^2 + u^2 - u'2(u' - 1) = C \\ & \text{or,} \qquad 1 - (u')^2 + u^2 = C. \end{aligned}$$

This time it is advantageous to differentiate the equation once more. The reason is that the constant C will vanish; the resulting equation is *second order* and so still has two degrees of freedom, as necessary. Differentiating, we obtain $2u'u'' - 2uu' = 0$ so, either $u = c$ or $u'' - u = 0$. Consider the latter. The auxiliary polynomial $r^2 - 1 = 0$ has roots at $r = \pm 1$ so the general solution is (see Appendix A)

$$u(x) = Ae^x + Be^{-x}.$$

The natural boundary conditions are

$$\begin{aligned} F_\xi(0, u(0), u'(0)) = 2(u'(0) - 1) = 0, \qquad & \text{and} \\ F_\xi(\log 2, u(\log 2), u'(\log 2)) = 2(u'(\log 2) - 1) = 0. & \end{aligned}$$

That is

$$1 = u'(0) = A - B, \qquad \text{and} \qquad 1 = u'(\log 2) = 2A - \frac{B}{2}$$

so

$$u(x) = \frac{1}{3}e^x - \frac{2}{3}e^{-x}.$$

If $u(x) = c$ then we are unable to satisfy the natural boundary conditions. A plot of this optimal solution shows it to be barely distinguishable from the straight line with slope 1, $y = x - \frac{1}{2}\log 2$. However,

$$I\Big(\frac{1}{3}e^x - \frac{2}{3}e^{-x}\Big) \approx 0.2648, \text{ while } I\Big(x - \frac{1}{2}\log 2\Big) \approx 0.2775.$$

□

EXERCISE 4.10. Re-solve the above example by solving

$$1 - (u')^2 + u^2 = C \qquad \Rightarrow \qquad \frac{1}{\sqrt{1 - C + u^2}}\, du = dx.$$

(Hint: make the substitution $u = \sqrt{1 - C}\sinh\varphi$ for the left hand side.)

One can also consider *inequalities* for the boundary conditions (e.g. $B_1 \le u(b) \le B_2$), and in practice such restrictions may be very important. These are treated simply by considering the boundary condition as free; we solve the problem and then look at the (potentially) optimal solution at b, $u(b)$. If indeed $B_1 \le u(b) \le B_2$ then this is our optimal solution; if not, however, the optimal solution must have either $u(b) = B_1$ or $u(b) = B_2$. One option now would be to solve the two variational problems with these fixed boundary conditions and simply check which was optimal.

4.5 Variational problems with integral constraints

Note: As in the last paragraph of the previous section, one can consider both equality and inequality constraints, but we will restrict our attention to *equality* constraints.

Now we consider variational problems which, in addition to having boundary conditions (one or both of which may be free) we have *integral* constraints: the optimal solution u must satisfy

$$J(u) := \int_a^b \vec{H}(x, u(x), u'(x))\, dx = \vec{\beta}. \tag{4.21}$$

Here, the functions comprising $\vec{H}$ are given, and $\vec{\beta}$ is given; it is u we are looking for.

A good example of an integral constraint will be seen when we look for the shape taken by a hanging cable strung between two points. The constraint is the *length* of the cable, which is, of course, given by an integral.

REMARK. Including fixed endpoints as integral constraints: sometimes it can be convenient to replace $u(a) = A,\ u(b) = B$ by

$$u(a) = A, \qquad \text{and} \qquad \int_a^b u'(x)\,dx = B - A$$

so that the second condition can be incorporated with the other integral constraint(s). In practice, re-writing the constraint this way can make no difference to the solutions. It might be helpful from a book-keeping point of view.

Boundary value constraints are constraints on the *value(s)* of the feasible function; integral constraints, on the other hand, are constraints on the feasible functions themselves. This situation is therefore more like the case of constrained optimization over $\mathbb{R}^n$, but this time over a vector space of functions. We approach the problem from the point of view of Lagrange's theorem. The proof of Lagrange's theorem is very much dependent on the finite-dimensionality of $\mathbb{R}^n$, and the inner product (the dot-product) of $\mathbb{R}^n$. To extend this to a proof in an infinite dimensional vector space of functions, we have to work with what is called the L^2 inner product which goes beyond the prerequisites of this material.

The idea, however, is the following. We consider the case where the boundary conditions are fixed, $u(a) = A$ and $u(b) = B$. Let u be an optimal function for $I(u)$ subject to the integral constraint (4.21). Let $\varphi \in C^1([a,b])$ be such that $\varphi(a) = \varphi(b) = 0$ and consider the admissible variation $v_t = u + t\varphi$. Once again (E-L) must hold since u is optimal. But further, to be admissible, it must also hold that $J(v_t) = \beta$ for all sufficiently small t. Differentiating this with respect to t and integrating by parts (just as in the proof of the Euler-Lagrange theorem), and combining with (E-L) for u, we obtain

$$\int_a^b \big(F_\lambda(x,u,u') - \frac{d}{dx}F_\xi(x,u,u')\big)\varphi\,dx = 0$$

for all φ such that

$$\int_a^b \big(H_\lambda(x,u,u') - \frac{d}{dx}H_\xi(x,u,u')\big)\varphi\,dx = 0$$

(plus, of course, $\varphi(a) = \varphi(b) = 0$). What this means, though we don't prove it here, is that the first integrand must be a multiple of the second,

$$F_\lambda(x,u,u') - \frac{d}{dx}F_\xi(x,u,u') + z\big(H_\lambda(x,u,u') - \frac{d}{dx}H_\xi(x,u,u')\big) = 0$$

for some scalar multiplier z.

We can pose his condition more succinctly by defining the *augmented Lagrangian*. Returning to possibly more than one integral constraint, we define

$$\tilde{F}(x,\lambda,\xi) := F(x,\lambda,\xi) + \vec{z}^{\,T}\vec{H}(x,\lambda,\xi).$$

THEOREM 4.11. *Consider the optimization problem: minimize*

$$I(u) = \int_a^b F(x, u(x), u'(x))\, dx \tag{4.22}$$

subject to $u(a) = A,\ u(b) = B$ *and* $\int_a^b \vec{H}(x, u(x), u'(x))\, dx = \vec{\beta}.$ (4.23)

Suppose there exists a vector $\vec{z}$ of multipliers such that the augmented Lagrangian $\tilde{F} = F + \vec{z}^T\vec{H}$ is convex as a function of ξ for each fixed $x \in (a, b)$, $\lambda \in \mathbb{R}$. If u is the unique solution to the (E-L) equation for $\tilde{F}$:

$$\frac{d}{dx}\left[\tilde{F}_\xi(x, u(x), u'(x))\right] = \tilde{F}_\lambda(x, u(x), u'(x))$$

and u satisfies (4.23), then u is an optimal solution of the problem (4.22, 4.23).

We should remark that the process of solving such a problem differs somewhat from the case where there are no integral constraints, and hence no vector of multipliers. When we solve the (E-L) equation for $\tilde{F}$, we will have parameters $z_1, \ldots, z_k$ floating around which are not (yet) determined; the potentially optimal solution u will be given in terms of these parameters. To determine what these z_j actually are, we then impose the conditions

$$\int_a^b \vec{H}(x, u(x), u'(x))\, dx = \vec{\beta}$$

using the actual u we have found. We will see this by example.

EXAMPLE 4.12. Minimize $I(u) = \int_0^1 (u'(x))^2\, dx$ under the restrictions $u(0) = u(1) = 0$, $\int_0^1 u(x)\, dx = 1$. Geometrically, we seek the function whose graph goes through $(0, 0)$ and $(1, 0)$, has minimal (absolute) slope, but has (potentially *signed*) area one.

The *augmented* Lagrangian is

$$\tilde{F}(x, \lambda, \xi) = F(x, \lambda, \xi) + z\lambda = \xi^2 + z\lambda.$$

Here, z is the undetermined multiplier. Since $\tilde{F}$ is independent of x, the (E-L) equation is now (by Beltrami's identity)

$$C = \tilde{F} - u'\tilde{F}_\xi = (u')^2 + zu - u'(2u') = zu - (u')^2.$$

Differentiating once more we obtain

$$\begin{aligned} 0 = u'(z - 2u'') \quad &\Rightarrow \quad u = c_0, \text{ or } u'' = \frac{z}{2} \\ &\Rightarrow \quad u = c_0, \text{ or } u = \frac{z}{4}x^2 + c_1 x + c_2. \end{aligned}$$

Since $u(0) = 0$, $c_0 = c_2 = 0$. Since $u(1) = 0$, $\frac{z}{4} + c_1 = 0$. Now we impose the integral constraint (we quickly see that $u = c_0$ is not a solution):

$$1 = \int_0^1 u(x)\,dx = \int_0^1 \Big(\frac{z}{4}x^2 + c_1 x\Big)\,dx = \frac{z}{12} + \frac{c_1}{2}.$$

These imply $z = -24$ and $c_1 = 6$. The optimal solution is thus $u(x) = -6x^2 + 6x$. □

4.5.1 A Hanging Cable example.

The potential energy of a cable hung between $(-D, 0)$ and $(D, 0)$ is proportional to (see (4.6))

$$I(u) = \int_{-D}^{D} u(x)\sqrt{1 + (u'(x))^2}\,dx.$$

We require $u(-D) = 0$ and $u(D) = 0$; the last constraint is that its length be L:

$$\int_{-D}^{D} \sqrt{1 + (u'(x))^2}\,dx = L.$$

Although it is not necessarily more efficient to do so, to illustrate the approach we absorb one boundary condition as an integral constraint: instead of $u(D) = 0$, we require $\displaystyle\int_{-D}^{D} u'(x)\,dx = 0$. Consider, then, the augmented Lagrangian $\tilde{F}(x, \lambda, \xi) = \lambda\sqrt{1 + \xi^2} + z_1\xi + z_2\sqrt{1 + \xi^2}$. The (E-L) equation gives, via the Beltrami identity,

$$\begin{aligned} C &= \tilde{F} - u'\tilde{F}_\xi \\ &= u\sqrt{1 + (u')^2} + z_1 u' + z_2\sqrt{1 + (u')^2} - u'\Big(\frac{uu'}{\sqrt{1 + (u')^2}} + z_1 + \frac{z_2 u'}{\sqrt{1 + (u')^2}}\Big) \end{aligned}$$

from which we obtain, provided $(u + z_2)^2 \neq C^2$,

$$u + z_2 = C\sqrt{1 + (u')^2} \quad \Rightarrow \quad u' = \pm\frac{\sqrt{(u + z_2)^2 - C^2}}{C} \quad \Rightarrow \quad \frac{du}{\sqrt{(u + z_2)^2 - C^2}} = \pm\frac{dx}{C}.$$

This is the same equation we solved in seeking the minimal area surface of revolution (see (4.14)) with u replaced by $u + z_2$. Its solution is thus

$$u + z_2 = C\cosh\Big(\pm\frac{x}{C} + d\Big), \quad \Rightarrow \quad u = C\cosh\Big(\frac{x}{C} + d\Big) - z_2$$

where we can take the positive sign since $\cosh$ is an even function. Imposing $u(-D) = 0$ and $0 = \displaystyle\int_{-D}^{D} u'(x)\,dx = u(D) - u(-D) = u(D)$, we have

$$C\cosh\Big(-\frac{D}{C} + d\Big) = C\cosh\Big(\frac{D}{C} + d\Big)$$

and, as in (4.16) and (4.17), we deduce $d = 0$. Next

$$u(-D) = 0 \quad \Rightarrow \quad z_2 = C\cosh\frac{D}{C}, \quad \text{so} \quad u = C\Big(\cosh\frac{x}{C} - \cosh\frac{D}{C}\Big).$$

Finally,

$$L = \int_{-D}^{D} \sqrt{1+(u')^2}\,dx = \int_{-D}^{D} \frac{u+z_2}{C}\,dx = \int_{-D}^{D} \cosh\frac{x}{C}\,dx = 2C\sinh\frac{D}{C}.$$

Thus, C is determined from $L = 2C\sinh\dfrac{D}{C}$. Note that we must have $L \geq 2D$ for the problem to make sense, and indeed this equation has no solution for C is $L \leq 2D$. If $L = 2D$, the solution is the constant solution $u = 0$. See Exercise 4.13.

EXERCISE 4.13. In solving the hanging cable problem above, we assumed that $(u+z_2)^2 \neq C^2$. Show that if $(u+z_2)^2 = C^2$ then there is a solution only if $L = 2D$, and show that the solution is $u = 0$.

4.6 Exercises

1. Find the solutions to

   (a) minimize $\displaystyle\int_0^1 \big(u'(x)^2 + 2u(x)^2\big)e^x\,dx$ subject to $u(0) = 1,\ u(1) = 0$, and

   (b) minimize $\displaystyle\int_0^\pi \big(u'(x)^2 - \frac{1}{2}u(x)^2\big)e^x\,dx$ subject to $u(0) = 1,\ u(\pi) = 2$.

2. Exercise 4.1, page 102.

3. Exercise 4.5, page 109.

4. Exercise 4.6, page 109.

5. Exercise 4.7, page 109.

6. Exercise 4.10, page 122.

7. Show that, assuming sufficient smoothness of all functions involved, if u is a minimizer of the *second order* problem

   $$\begin{aligned} &\text{minimize} \qquad \int_a^b F(x, u(x), u'(x), u''(x))\,dx \\ &\text{subject to } (u(a), u(b), u'(a), u'(b)) = (A, B, C, D) \end{aligned}$$

   then u satisfies the second order Euler-Lagrange equation

   $$F_\lambda(x, u, u', u'') - \frac{d}{dx}\big(F_\xi(x, u, u', u'')\big) + \frac{d^2}{dx^2}\big(F_\mu(x, u, u', u'')\big) = 0 \tag{4.24}$$

   where $F : (a, b) \times \mathbb{R} \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ and we use the variables $F(x, \lambda, \xi, \mu)$.

8. We wish to drive a rocket propelled car from $u = 0$ to $u = 1$ in time 1, minimizing the use of fuel. The only way to slow the car down is to fire the rocket in the opposite direction, so deceleration costs fuel in the same way as acceleration does. We require that the car begins and ends with zero velocity.

   (a) If we model fuel consumption as being due to acceleration *and* to velocity, solve the optimization problem

   $$\begin{aligned} &\text{minimize} \qquad \int_0^1 \left((u'(t))^2 + (u''(t))^2\right) dt \\ &\text{subject to} \quad u(0) = 0,\ u(1) = 1,\ u'(0) = u'(1) = 0. \end{aligned}$$

   You will need to use Exercise 7. Use *Mathematica* to `Plot[ ]` your solution.

   (b) If we model fuel consumption as being due *only* to acceleration, solve the optimization problem

   $$\text{minimize} \quad \int_0^1 (u''(t))^2\, dt \quad \text{subject to} \quad u(0) = 0,\ u(1) = 1,\ u'(0) = u'(1) = 0.$$

   Use *Mathematica* to `Plot[ ]` your solution and to compute $\displaystyle\int_0^1 (u''(t))^2\, dt$. Show that this is (slightly) smaller for this problem than for the u of the previous problem.

9. Consider the following variation on the Brachistochrone problem: Starting at $x = 0,\ y = 1$, what path should the bead follow so as to reach $x = 3$ in the shortest amount of time, with the requirement that, at $x = 3$, $-\dfrac{1}{2} \le y \le \dfrac{1}{2}$?

   Recall that to answer this problem you need to first solve the free boundary value problem; if the solution to this problem satisfies $-\dfrac{1}{2} \le y \le \dfrac{1}{2}$ at $x = 3$, then this will be the optimal solution. If not, then you must solve the two boundary value problems with $y = \pm\dfrac{1}{2}$ at $x = 3$, and compare the values of the objective functional for each of these solutions. This last step introduces a new problem: given the parameterized curve $x(\varphi)$ and $y(\varphi)$, how *long* does the bead take to reach $x = 3$? If the curve is given as the graph of a function $u(x)$, say, then we know this time is

   $$\int_0^3 \sqrt{\frac{1 + u'(x)^2}{1 - u(x)}}\, dx.$$

   However, we don't know this function u. We must make the change of variables $x = x(\varphi)$ in the integral. We achieve this using

   $$u'(x) = \frac{dy}{dx} = \frac{dy}{d\varphi} \Big/ \frac{dx}{d\varphi}\,, \qquad u(x(\varphi)) = y(\varphi), \qquad \text{and} \qquad dx = x'(\varphi)\, d\varphi$$

   together with the fact that $x = 3$ when $\varphi = \varphi_0$, which we must determine. As in the solution to the Brachistochrone problem, you will need to use *Mathematica*. You will need to modify the module given on page 113 which is formulated only for the case that $y = 0$ at $x = X$.

10. Solve the following case of Problem 8 in Chapter 1: find the function $y(x) \leq 0$ defined on $-1 \leq x \leq 1$ which minimizes the arc-length ($-1 \leq x \leq 1$) subject to $y(-1) = y(1) = 0$, and the area trapped between the curve $y(x)$ and $y = 0$ is A, where $0 < A \leq \pi/2$.

    If, with everything else the same, A is changed to be $A > \pi/2$, there is no solution to the corresponding Euler-Lagrange equation which gives $y(x)$ as the graph of a function. What do you think will be the optimal design of the channel nonetheless?

11. In Bayesian probability theory, the principle of maximum entropy seeks the function $u(x)$, defined on $(0, \infty)$, which maximizes

$$I(u) = -\int_0^\infty u(x) \log u(x)\, dx.$$

    If we normalize the resulting function to have mass 1 and "first moment" $1/\alpha$, then we obtain the constraints

$$\int_0^\infty u(x)\, dx = 1, \qquad \int_0^\infty x u(x)\, dx = \frac{1}{\alpha}.$$

    Find the optimal function.

12. In this exercise, we demonstrate that the natural boundary condition leads to the optimal solution among all possible boundary conditions.

    (a) Solve

$$\text{minimize} \quad \int_0^{\log 2} \left(u_\beta'(x)^2 + u_\beta(x)^2\right) dx \quad \text{subject to} \quad u_\beta(0) = 1,\ u_\beta(\log 2) = \beta$$

    where β is an arbitrary boundary condition. Define

$$f(\beta) = \int_0^{\log 2} \left(u_\beta'(x)^2 + u_\beta(x)^2\right) dx$$

    to be the optimal value of the integral for given β. Solve $f'(\beta) = 0$ for the β^* which results in the minimal integral among all the possible boundary conditions. The resulting $u_{\beta^*}(x)$ is thus the solution to the free boundary condition problem

$$\text{minimize} \quad \int_0^{\log 2} \left(u_\beta'(x)^2 + u_\beta(x)^2\right) dx \quad \text{subject to} \quad u_\beta(0) = 1.$$

    (b) Solve the above free boundary condition problem using the natural boundary condition. Your two solutions should agree.

13. Exercise 4.13, page 126.

14. When a cylindrical can containing a volume V of fluid is spun around its axis at an angular velocity of ω_0 (radians per second), the fluid is subject to a centripetal force toward the axis (exerted by the wall of the cylinder) and to the gravitational force. The profile

of the fluid will be such that the total potential energy of the fluid is minimized. Suppose that the cylinder is of radius 1; by symmetry, the profile can be given in cylindrical coordinates with $z = z(r)$, $0 \le r \le 1$. The potential energy associated with this profile is (proportional to)

$$U(z) = \int_0^1 \big(z(r)^2 - 2kz(r) - \omega_0^2 r^2 z(r)\big) r \, dr$$

where k is a constant. Minimize this functional, subject to the constraint

$$V = 2\pi \int_0^1 z(r) r \, dr,$$

that is, the volume of the fluid is V.

15. Show that, assuming sufficient smoothness of all functions involved, if u is a minimizer of the *second order* problem with integral constraint

$$\begin{aligned} \text{minimize} \quad & \int_a^b F(x, u(x), u'(x), u''(x)) \, dx \\ \text{subject to} \quad & (u(a), u(b), u'(a), u'(b)) = (A, B, C, D), \\ \text{and} \quad & \int_a^b H((x, u(x), u'(x), u''(x)) \, dx = k \end{aligned}$$

then u satisfies the second order Euler-Lagrange equation (4.24) for the *augmented Lagrangian*

$$\tilde{F}(x, \lambda, \xi, \mu) = F(x, \lambda, \xi, \mu) + zH(x, \lambda, \xi, \mu)$$

where z is a multiplier to be determined.

16. A variant of Exercise 8 is the following: the car is actually a ride at an amusement park, and to make it more exciting, it is desired that the car travel quickly throughout the ride, but still requiring that the ride starts and ends with zero velocity, and that we minimize fuel costs. We model this as follows:

$$\text{minimize} \quad \int_0^1 (u''(t))^2 \, dt \qquad \text{subject to} \quad u(0) = 0, \ u(1) = 1, \ u'(0) = u'(1) = 0,$$
$$\text{and} \ \int_0^1 (u'(t))^2 \, dt = k.$$

The last constraint quantifies the *total velocity* experienced by the riders. You will need to use Exercise 15. Solve the problem for $k = 1.1$ and $k = 1.5$. Notes:

- The resulting equation is a fourth order, constant coefficient, linear ODE. Using the (fourth degree) auxiliary polynomial you will find the roots are $r = 0$ (repeated) and $r^2 = z$. You will need to consider the three cases $z < 0$, $z = 0$ and $z > 0$. The case $z = 0$ is actually just Exercise 8b, so you may disregard this case. The repeated root $r = 0$ results in a solution $c_1 t + c_2$. When $z < 0$ the other part of the solution is trigonometric (let $w = \sqrt{-z}$ and obtain $c_3 \sin wt + c_4 \cos wt$); when $z > 0$ it is exponential (let $w = \sqrt{z}$).

- Use *Mathematica* to solve the system of equations for c_1, c_2, c_3, c_4 in terms of w (and `Simplify[ ]`). Obtain $u(t, w)$.
- Have *Mathematica* `Integrate[ ]` $(u'(t, w))^2$ over $0 \leq t \leq 1$ to obtain a function $f(w)$.
- `Plot[ ]` $f(w)$ to get an idea of what you are after in order to try to solve $f(w) = k$. You will have to play with the range of w over which to plot to get a reasonable graph.
- Use `FindRoot[f[w] == k, {w, w0}]` with an appropriate `w0` to find (numerically) the solution to $f(w) = k$.
- Plot the resulting solution $u(t)$ over $0 \leq t \leq 1$, and compute $\int_0^1 (u''(t))^2 \, dt$ to see how much the solution *costs*.
- What value of k does the solution to Exercise 8b correspond to?
- Try plotting the solutions with $k = 1.1$, with $k = 1.5$, and with no integral constraint (Exercise 8b) all on the same graphics. You should see that the solution with $k = 1.5$ is *steeper* than that with no constraint, which is steeper than the solution with $k = 1.1$. Is this consistent with the answer to the previous bullet?

Chapter 5

Optimal Control

5.1 Introduction

EXAMPLE 5.1. Suppose that we have a remote control car which can accelerate and decelerate according to $-a \leq u \leq b$. We wish to get the car from $x = 0$ to $x = \alpha$, starting and finishing at rest, in the minimal amount of time. What is the best strategy?

State and Controls: The state of a given system is described by a number of parameters $\vec{x} = (x_1, x_2, \ldots, x_n)$ and the system evolves according to a given

$$\textbf{state equation:} \qquad \vec{x}\,'(t) = f\big(t, \vec{x}(t), \vec{u}(t)\big)$$

where $\vec{u}(t) = (u_1, u_2, \ldots, u_m)$ is a given vector of *controls* on the system.

The control $\vec{u}$ will be constrained by, for example, $\vec{u} \in K \subset \mathbb{R}^m$.

The state equation also comes with initial and/or final conditions: $\vec{x}(0) = \vec{x}_0$, $\vec{x}(T) = \vec{x}_T$.

In our example, the *state* of the system at time t consists of the cars position together with its velocity: $\vec{x}(t) = (x_1(t), x_2(t))$ where $x_1(t)$ is the position and $x_2(t) = x_1'(t)$ is its velocity. The *control* is the acceleration $u(t)$; the state equation is thus

$$\vec{x}\,'(t) = (x_1'(t), x_2'(t)) = (x_2(t), u(t)).$$

Notice that we have re-written the second order equation for acceleration as a system of first order equations. Indeed, the state equation is, by definition, a first order ordinary differential equation. The boundary conditions are $\vec{x}(0) = (0, 0)$ and $\vec{x}(T) = (\alpha, 0)$, where (the unknown) T is when the process ends.

Objective functional: Of course, since we are seeking an *optimal* control, we must have a cost which we seek to minimize. This is the objective functional

$$I(\vec{x}, \vec{u}) = \int_0^T F(t, \vec{x}(t), \vec{u}(t))\, dt, \qquad F : (0, T) \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}.$$

In the example,

$$I(\vec{x}, u) = \int_0^T 1\, dt$$

which gives the time taken to complete the process.

Feasible or **admissible** vectors: The pair $\vec{x}(t)$ and $\vec{u}(t)$ are admissible if

- $\vec{u}(t) \in K$ for $0 \leq t \leq T$;
- $\vec{x}\,'(t) = f(t, \vec{x}(t), \vec{u}(t))$ for $0 \leq t \leq T$;
- $\vec{x}(0) = \vec{x}_0$, $\vec{x}(T) = \vec{x}_T$.

Notice an optimal control problems, as described, is a generalization of the variational problems of the previous chapter. A variational problem can be posed as an optimal control problem simply by setting the state equation to be $x'(t) = u(t)$.

5.2 The Hamiltonian

Assume that $K = \mathbb{R}^m$ for now. The optimization problem: minimize

$$I(\vec{x}, \vec{u}) = \int_0^T F(t, \vec{x}(t), \vec{u}(t))\, dt$$

subject to

$$\vec{x}\,'(t) = f(t, \vec{x}(t), \vec{u}(t)), \text{ or } f(t, \vec{x}(t), \vec{u}(t)) - \vec{x}\,'(t) = 0$$

is similar to the variational problem with an integral constraint (we can re-write $\vec{x}\,'(t) = f(t, \vec{x}(t), \vec{u}(t))$ as $\displaystyle\int_0^T f(t, \vec{x}(t), \vec{u}(t)) = \vec{x}(T) - \vec{x}(0)$). Here, however, we now have a constraint at each value of $0 \leq t \leq T$ (it must also hold that $\displaystyle\int_0^t f(s, \vec{x}(s), \vec{u}(s)) = \vec{x}(t) - \vec{x}(0)$ for each t). Consequently, we introduce a multiplier $\vec{p}$, just like we did for the variational problem,

but now we must have a *continuum* $\vec{p}(t)$ of multipliers, one for every t. Thus we consider the augmented functional

$$\begin{aligned} I^*(\vec{x},\vec{u},\vec{p},\vec{x}\,') &= \int_0^T \left[F(t,\vec{x}(t),\vec{u}(t)) + \vec{p}(t)^T\big(f(t,\vec{x}(t),\vec{u}(t)) - \vec{x}\,'(t)\big)\right] dt \\ &= \int_0^T G(t,\vec{u},\vec{p},\vec{x},\vec{u}\,',\vec{p}\,',\vec{x}\,')\,dt, \qquad \text{say.} \end{aligned}$$

Note that we include possible dependence of G on $\vec{u}\,'$ and $\vec{p}\,'$ for generalization later.

The optimal solution will again satisfy (E-L) for G, where we must think of $\vec{\lambda} = (\vec{u},\vec{p},\vec{x})$ and $\vec{\xi} = (\vec{u}\,',\vec{p}\,',\vec{x}\,')$; we must extend (E-L) to this case where $\vec{\lambda}$ is vector-valued. In this context, (E-L) says $\dfrac{d}{dt}G_{\vec{\xi}} = G_{\vec{\lambda}}$. If we separate out components of this equation, we obtain

$$\frac{d}{dt}(G_{\vec{x}\,'}) = G_{\vec{x}}, \qquad \frac{d}{dt}(G_{\vec{u}\,'}) = G_{\vec{u}}, \qquad \frac{d}{dt}(G_{\vec{p}\,'}) = G_{\vec{p}}.$$

Now $G = F + \vec{p}^T(f - \vec{x}\,')$, so we can re-write these in terms of F and f. The first equation says $\dfrac{d}{dt}\big(-\vec{p}\big) = F_{\vec{x}} + \vec{p}^T f_{\vec{x}}$, so $F_{\vec{x}} + \vec{p}^T f_{\vec{x}} + \vec{p}\,' = \vec{0}$; for the second and third we observe that G is independent of $\vec{u}\,'$ and $\vec{p}\,'$. Thus

$$\begin{aligned} \vec{0} &= F_{\vec{x}}(t,x,u) + \vec{p}^T f_{\vec{x}}(t,x,u) + \vec{p}\,' \\ \vec{0} &= F_{\vec{u}}(t,x,u) + \vec{p}^T f_{\vec{u}}(t,x,u), \\ \vec{0} &= \vec{x}\,' - f(t,x,u). \end{aligned} \tag{5.1}$$

The Hamiltonian: Define $H = F + \vec{p}^T f$. In terms of H, then, (E-L) becomes

$$\vec{p}\,' = -\frac{\partial H}{\partial \vec{x}}, \qquad \frac{\partial H}{\partial \vec{u}} = \vec{0}, \qquad \vec{x}\,' = f(t,\vec{x},\vec{u}).$$

Since we have derivatives of $\vec{x}$ and $\vec{p}$, we must have sufficiently many additional constraints to determine the solution. These constraints may come in the form of boundary conditions. However, if at an end-point a boundary condition for x_j is *free* then we impose the **transversality condition** $p_j = 0$ at this end-point. (One would have to go through the derivation of (E-L) for this more complicated problem to see why this is the correct condition.)

EXAMPLE 5.2. With $I(x,u) = \displaystyle\int_0^1 u(t)^2\,dt$, state equation $x'(t) = f(t,x,u) = u(t) + ax(t)$ and boundary condition $x(0) = 1$, find the optimal control u. Here a is a constant, and the boundary condition at $t = 1$ is free.

We compute

$$H = F + pf = u^2 + p(u + ax).$$

The (E-L) equations yield

$$p' = -\frac{\partial H}{\partial x} = -ap, \qquad 0 = \frac{\partial H}{\partial u} = 2u + p, \qquad x' = u + ax.$$

The second equation implies $u = -\frac{1}{2}p$, so the third can be re-written $x' = -\frac{1}{2}p + ax$. We also have $x(0) = 1$, and there is no boundary condition at $x = 1$ so we impose the transversality condition $p(1) = 0$.

The equation $p' = ap$ has solution $p(t) = ce^{-at}$ and $p(1) = 0$ gives $p(t) = 0$ for all t. Then $x(t) = e^{at}$ and $u(t) = 0$ for all t; the optimal control is to do nothing! Looking back at the objective function, we might have seen that $u = 0$ will minimize $I(u)$, and $x' = u + ax = ax$ can satisfy the given boundary condition.

Now let's change the boundary conditions to get something more interesting: we consider $x(0) = 1$, $x(1) = 0$. Again we have $p(t) = ce^{-at}$, but no longer require $p(1) = 0$. Plugging this into $x' = -\frac{1}{2}p + ax$ and solving the ODE we obtain

$$x(t) = \frac{c}{4a}e^{-at} + de^{at}.$$

Imposing the boundary conditions we find

$$c = \frac{2ae^a}{\sinh a}, \qquad d = -\frac{e^{-a}}{2\sinh a}, \qquad x(t) = \frac{e^a}{2\sinh a}e^{-at} - \frac{e^{-a}}{2\sinh a}e^{at}.$$

This time we obtain the optimal control via $u = -\frac{1}{2}p$ to obtain $u(t) = -\frac{ae^{a(1-t)}}{\sinh a}$.

EXAMPLE 5.3. This is a vector-valued example. Let the cost be the same as the previous example, but consider $x''(t) = u(t)$ (like the introductory example) with $x(0) = x'(0) = 1$ and $x(1) = 0$. We can think of the control as acceleration and the cost fuel consumption. Re-writing as a system of first order equations, we obtain

$$\begin{aligned} x_1' &= x_2, & x_2' &= u \\ x_1(0) &= 1, & x_1(1) &= 0, & x_2(0) &= 1. \end{aligned}$$

With $\vec{x} = [x_1, x_2]^T$, $\vec{x}\,' = f(t, \vec{x}, u) = [x_2, u]^T$. Now $H = F + \vec{p}^T f = u^2 + p_1 x_2 + p_2 u$. Since x_2 is free at $t = 1$, we set $p_2(1) = 0$. The full (E-L) set, together with the endpoint conditions is now

$$\begin{aligned} 0 &= \frac{\partial H}{\partial u} = 2u + p_2 \\ p_1' &= -\frac{\partial H}{\partial x_1} = 0, & p_2' &= -\frac{\partial H}{\partial x_2} = -p_1 \\ x_1' &= f_1 = x_2, & x_2' &= f_2 = u \\ x_1(0) &= 1, & x_1(1) &= 0 \\ x_2(0) &= 1, & p_2(0) &= 0. \end{aligned}$$

Again, from the first equation we have $u = -\frac{1}{2}p_2$. Starting with $p_1' = 0$ we obtain (easily)

$$\begin{aligned} p_1 &= c_1 \\ p_2 &= -\int p_1\, dt = -c_1 t + c_2 \\ u &= -\frac{1}{2}p_2 = \frac{c_1}{2}t - \frac{c_2}{2} \\ x_2 &= \int u\, dt = \frac{c_1}{4}t^2 - \frac{c_2}{2}t + c_3 \\ x_1 &= \int x_2\, dt = \frac{c_1}{12}t^3 - \frac{c_2}{4}t^2 + c_3 t + c_4. \end{aligned}$$

Now it's simply a matter of imposing the boundary conditions to find the c_j. It can be efficient to carefully choose which equation to use. The second with $p_2(1) = 0$ gives $c_1 = c_2$; the fourth with $x_2(0) = 1$ and the fifth with $x_1(0) = 1$ give $c_3 = c_4 = 1$. Thus $x_1 = \frac{c_1}{12}t^3 - \frac{c_1}{4}t^2 + t + 1$, and $x_1(1) = 0$ gives $c_1 = 12$. We have as our solution

$$u(t) = 6t - 6 \qquad\qquad x(t) = x_1(t) = \frac{1}{2}t^3 - \frac{3}{2}t^2 + t + 1.$$

In the above examples, we have been minimizing *cost* over a given time interval. Another important and interesting problem is to minimize the *time* given a fixed cost.

EXAMPLE 5.4. Suppose we have a process for which $x(0) = 1$ and we want to obtain $x(T) = 0$ in the least time possible; the process must satisfy the state equation

$$x'(t) = ax(t) + u(t)$$

for a given constant a, and we are constrained by

$$\int_0^T u(t)^2\, dt = K$$

for another given constant K. The functional we wish to minimize is

$$I(x, u) = \int_0^T 1\, dt$$

for this gives the total time taken. Recall that integral constraints for variational problems were handled by introducing a multiplier into the Lagrangian. We do the same here, which results in the Hamiltonian

$$H = 1 + zu^2 + p(ax + u).$$

We must solve

$$p' = ap \qquad\qquad 0 = 2zu + p \qquad\qquad x' = ax + u$$

plus the constraints. We obtain $p(t) = c_1 e^{-at}$ and $u(t) = -\dfrac{c_1}{2z} e^{-at}$. Then

$$x' = ax - \frac{c_1}{2z} e^{-at} \qquad \Rightarrow \qquad x(t) = \frac{c_1}{4az} e^{-at} + c_2 e^{at}.$$

Setting $x(0) = 1$ and solving for c_2, $c_2 = 1 - \dfrac{c_1}{4az}$. This gives

$$x(t) = e^{at}\Big(1 - \frac{c_1}{4az}\Big) + \frac{c_1}{4az} e^{-at}.$$

Now

$$K = \int_0^T u(t)^2 \, dt = \int_0^T \frac{c_1^2}{4z^2} e^{-2at} \, dt = \frac{c_1^2}{8az^2}\big(1 - e^{-2aT}\big)$$

and

$$0 = x(T) = e^{aT}\Big(1 - \frac{c_1}{4az}\Big) + \frac{c_1}{4az} e^{-aT}$$

together imply $c_1 = 2Kz$ and $T = -\dfrac{1}{2a} \ln\Big(1 - \dfrac{2a}{K}\Big)$. The solution make sense only if $K > 2a$. The optimal control is then

$$u(t) = -Ke^{-at}.$$

Our original problem of accelerating and then decelerating a model car is still not solvable by these methods. We don't have a mechanism by which we can impose the bounds on u. Without bounds on u, we can clearly make the time as small as possible by accelerating arbitrarily quickly and decelerating again arbitrarily quickly. We need *Pontryagin's Principle*.

5.2.1 Pontryagin's Principle

We now consider optimal control problems where we put constraints on the control, $\vec{u}(t) \in K \subset \mathbb{R}^m$. Our goal is to evolve the system $\vec{x}$ from some given initial state $\vec{x}(0)$ to (if not free) some final state $\vec{x}(T)$, always satisfying the state equation $\vec{x}\,'(t) = f(t, \vec{x}(t), \vec{u}(t))$, and to do so in such a way as to minimize $\displaystyle\int_0^T F(t, \vec{x}, \vec{u}) \, dt$.

We'll derive some necessary conditions which must be satisfied, so suppose that $\vec{u}(t)$ is optimal, and $\vec{x}(t)$ is the corresponding state function. Suppose that at some (fixed, for the moment) time $0 \le t \le T$, the system is in state $\vec{x}(t)$. Consider the problem of minimizing the objective functional *from this time on*, i.e. finding the optimal control $\vec{u}(s)$ and corresponding state function

$\vec{y}(s)$ for $t \le s \le T$ with initial condition now $\vec{y}(t) = \vec{x}(t)$. We define the function $S(t, \vec{x}(t))$ to be the value of the functional attained by this optimal control:

$$S(t,\vec{x}(t)) = \min_{\vec{u}}\Big\{\int_t^T F(s,\vec{y}(s),\vec{u}(s))\,ds \;:\; \vec{y}\,'(s) = f(s,\vec{y}(s),\vec{u}(s)),\ \vec{u}(s)\in K,\ t<s<T,\ \vec{y}(t)=\vec{x}(t)\Big\}.$$

This function S satisfies the following somewhat complicated equality which we will explain immediately following: for any $t' > t$,

$$S(t,\vec{x}(t)) = \min_{\vec{y}}\Big\{\min_{\vec{u}}\Big\{\int_t^{t'} F(s,\vec{z}(s),\vec{u}(s))\,ds \;:\; \vec{z}\,'(s) = f(s,\vec{z}(s),\vec{u}(s)),\ \vec{u}(s)\in K, \\ t<s<t',\ \vec{z}(t)=\vec{x}(t),\ \vec{z}(t')=\vec{x}(t)+y(t')(t'-t)\Big\} \\ + S(t',\vec{x}(t)+y(t')(t'-t))\Big\}.$$

The inner minimization yields, for a given $\vec{y}$, what results from the optimal way of getting from state $\vec{x}(t)$ at time t to state $\vec{x}(t) + \vec{y}(t')(t'-t)$ at time t'. Now, again with this given $\vec{y}$, we add $S(t', \vec{x}(t) + \vec{y}(t')(t'-t))$, which is the minimal possible result in getting from state $\vec{x}(t) + \vec{y}(t')(t'-t)$ at time t' to the final state at time T. In other words, for a given $\vec{y}$ we have found the optimal control which "links" the state $\vec{x}(t)$ at time t with the state $\vec{x}(t) + \vec{y}(t')(t'-t)$ at time t'. Now, by definition, $S(t, \vec{x}(t))$ is the minimal result for getting from $\vec{x}(t)$ at time t to the final state at time T, so it must equal the minimum of what we just described over all possible $\vec{y}$.

We can re-write this, with $\Delta t = t' - t$ as

$$0 = \min_{\vec{y}}\Big\{\min_{\vec{u}}\Big\{\frac{1}{\Delta t}\int_t^{t+\Delta t} F(s,\vec{z}(s),\vec{u}(s))\,ds \;:\; \vec{z}\,'(s) = f(s,\vec{z}(s),\vec{u}(s)),\ \vec{u}(s)\in K, \\ t<s<t+\Delta t,\ \vec{z}(t)=\vec{x}(t),\ \vec{z}(t+\Delta t)=\vec{x}(t)+y(t+\Delta t)\Delta t\Big\} \\ + \frac{S(t+\Delta t,\vec{x}(t)+\vec{y}(t+\Delta t)\Delta t) - S(t,\vec{x}(t))}{\Delta t})\Big\}.$$

Notice that the conditions within the inner minimization can be combined as

$$\frac{\vec{z}(t+\Delta t) - z(t)}{\Delta t} = \vec{y}(t+\Delta t)$$

and taking the limit as $\Delta t \to 0$, this becomes $\vec{z}\,'(t) = f(t, \vec{x}(t), \vec{u}(t)) = \vec{y}(t)$. By the fundamental theorem of calculus, together with the definition of the derivative, we obtain

$$0 = \min_{\vec{y}}\Big\{\min_{\vec{u}\in K}\big\{F(t,\vec{x},\vec{u}(t)) \;:\; \vec{y}(t) = f(t,\vec{x}(t),\vec{u}(t))\big\} + \frac{\partial S}{\partial t}(t,\vec{x}(t)) + \vec{y}(t)\frac{\partial S}{\partial x}(t,\vec{x}(t))\Big\}.$$

We can re-write this as

$$\frac{\partial S}{\partial t}(t,\vec{x}(t)) = -\min_{\vec{u}\in K}\Big\{F(t,\vec{x}(t),\vec{u}(t)) + f(t,\vec{x}(t),\vec{u}(t))\frac{\partial S}{\partial x}(t,\vec{x}(t))\Big\}.$$

Now, from the definition of S, since $(\vec{x}(t), \vec{u}(t))$ is optimal, for each t it must hold that

$$S(t, \vec{x}(t)) = \int_t^T F(s, \vec{x}(s), \vec{u}(s))\, ds, \qquad \vec{x}\,'(s) = f(s, \vec{x}(s), \vec{u}(s)),\ t < s < T.$$

If we differentiate this with respect to t and combine it with what we just derived, we obtain (after some work)

$$\frac{d}{dt}\frac{\partial S}{\partial x} = -\frac{\partial F}{\partial x}(t, \vec{x}(t), \vec{u}(t)) - \frac{\partial S}{\partial x}\frac{\partial f}{\partial x}(t, \vec{x}(t), \vec{u}(t)).$$

This is of the same form as the first equation (5.1) satisfied by the Hamiltonian. This motivates us to define

$$\tilde{p}(t, \vec{x}) = \frac{\partial S}{\partial x}(t, \vec{x})$$

and, as usual, consider the Hamiltonian

$$H(t, \vec{x}, \vec{u}, \vec{p}) = F(t, \vec{x}, \vec{u}) + \vec{p}^T f(t, \vec{x}, \vec{u})$$

where, when $\vec{x}(t)$ is optimal, $\vec{p}(t) = \tilde{p}(t, x(t)) = \dfrac{\partial S}{\partial x}(t, \vec{x}(t))$.

THEOREM 5.5 (Pontryagin's principle). *If $(\vec{x}(t), \vec{u}(t))$ is optimal for the control problem: minimize*

$$\int_0^T F(t, \vec{x}, \vec{u})\, dt,\ \vec{x}\,'(t) = f(t, \vec{x}(t), \vec{u}(t)),\ \vec{u}(t) \in K$$

with $\vec{x}(0) = \vec{x}_0$, $\vec{x}(T) = \vec{x}_T$ given, then there must exist a function $\vec{p}(t)$ such that

$$\begin{aligned} \vec{p}\,'(t) &= -\frac{\partial H}{\partial x}(t, \vec{x}(t), \vec{u}(t), \vec{p}(t)), \\ H(t, \vec{x}(t), \vec{u}(t), \vec{p}(t)) &= \min_{\vec{v} \in K} H(t, \vec{x}(t), \vec{v}, \vec{p}(t)), \\ \vec{x}\,'(t) &= f(t, \vec{x}(t), \vec{u}(t)),\ \vec{x}(0) = \vec{x}_0,\ \vec{x}(T) = \vec{x}_T. \end{aligned}$$

Note: if, say, the endpoint $t = T$ is free, then the condition $x(T) = x_T$ gets replaced by $\vec{p}(T) = 0$.

Applying this theorem can be non-trivial. We'll look at a couple of examples.

EXAMPLE 5.6. Recall the original problem of controlling a remote control car. Here $K = [-a, b]$ and the cost functional is

$$I = \int_0^T 1\, dt$$

with the conditions $x''(t) = u(t),\ u \in K$, $x(0) = x'(0) = 0$, $x(T) = \alpha$, $x'(T) = 0$. Re-writing as a system of first order equations, as earlier,

$$\vec{x}\,' = [x_1', x_2']^T = [x_2, u]^T,\ u \in K$$
$$\vec{x}(0) = [0,0]^T,\ \vec{x}(T) = [\alpha, 0]^T.$$

The Hamiltonian is $H = 1 + p_1 x_1 + p_2 x_2$, and at an optimal solution, Pontryagin's principle imposes

$$p_1' = 0,\ p_2' = -p_1 \qquad \Rightarrow \qquad p_1 = c_1,\ p_2 = -c_1 t + c_2.$$

The second condition in Theorem 5.5 is now, for each $0 < t < T$,

$$1 + c_1 x_1(t) + (-c_1 t + c_2)u(t) = \min_{v \in [-a,b]} \big\{1 + c_1 x_1(t) + (-c_1 t + c_2)v\big\},$$

or, equivalently,

$$(-c_1 t + c_2)u(t) = \min_{v \in [-a,b]} (-c_1 t + c_2)v.$$

Recall, we consider t fixed in this problem. Clearly, the minimum is $v = -a$ if $-c_1 t + c_2 > 0$ and $v = b$ if $-c_1 t + c_2 < 0$. If $-c_1 t + c = 0$ then v can take any value in $[-a, b]$. So,

$$u(t) = \begin{cases} -a, & -c_1 t + c_2 > 0, \\ b, & -c_1 t + c_2 < 0, \\ \text{anything}, & -c_1 t + c_2 = 0. \end{cases}$$

At $t = 0$ we must have $u(0) = b$ (in order for the car to move forward), so in fact $u(t) = b$ until $-c_1 t + c_2 = 0$, that is until $t = t_0 = c_2/c_1$. For $t > t_0$, $-c_1 t + c_2 > 0$, so then $u(t) = a$. Bringing this information back to $\vec{x}$, we have $x_2' = b$ and $x_1' = x_2$, $0 \le t \le t_0$, with $x_1(0) = x_2(0) = 0$. Thus $x_1(t) = \dfrac{1}{2} b t^2$. Then, for $t > t_0$, we have $x_2' = -a$ and $x_1' = x_2$, $t_0 \le t \le T$, with $x_1(t_0) = \dfrac{1}{2} b t_0^2$, $x_2(t_0) = b t_0$. From this we deduce:

$$\begin{aligned} x_2 &= -at + \beta \\ x_1' = x_2 \quad &\Rightarrow \quad x_1 = -\frac{a}{2} t^2 + \beta t + \gamma \\ \frac{b}{2} t_0 t^2 = x_1(t_0) &= -\frac{a}{2} t_0^2 + \beta t_0 + \gamma \\ b t_0 = x_2(t_0) &= -a t_0 + \beta. \end{aligned}$$

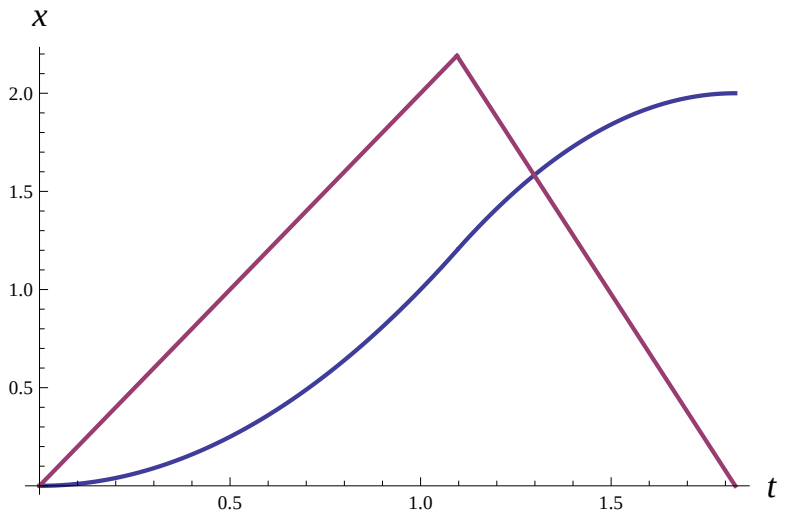
These last two equations easily give $\beta = t_0(a + b)$, $\gamma = -\dfrac{1}{2} t_0^2 (a + b)$. Thus

$$x_1(t) = -\frac{a}{2} t^2 + t_0(a + b)t - \frac{t_0^2}{2}(a + b).$$

We still don't know t_0, nor $c_1,\ c_2$. We have the condition $x'(T) = 0$ which gives $T = t_0(a + b)/a$, and putting this into $x(T) = \alpha$,

$$-\frac{a}{2}t_0^2\frac{(a+b)^2}{a^2} + t_0^2\frac{(a+b)^2}{a} - t_0^2\frac{a+b}{2} = \alpha \quad \Rightarrow \quad t_0 = \sqrt{\frac{2a\alpha}{b(a+b)}} \quad \Rightarrow \quad T = \sqrt{\frac{2(a+b)\alpha}{ab}}.$$

With $a = 3$ and $b = 2$ (so we can brake more powerfully than we can accelerate), and $\alpha = 2$ we obtain the following results for $x(t)$ and $x'(t)$ (we find $t_0 \approx 1.1$, $T = 1.83$):



Appendix A

Constant coefficient, linear, ordinary differential equations

We briefly outline how to solve constant coefficient, linear, ordinary differential equations. Such an equation, of order n, is of the form

$$a_n y^{(n)}(t) + a_{n-1} y^{(n-1)}(t) + \cdots + a_1 y'(t) + a_0 y(t) = f(t), \tag{A.1}$$

where $a_0, \ldots, a_n$ are constants, and $a_n \neq 0$. When such equations are of second order and homogeneous (zero "right-hand-side"), this is a review of what is covered in Math 331, though perhaps from a slightly different point of view. When such equations are non-homogeneous, this is sometimes covered in Math 331, and sometimes covered in Math 435. We will only need the basic structure of how to obtain solutions.

An ordinary differential equation (ODE) is *linear* if, and only if, whenever $y_1(t)$ and $y_2(t)$ are solutions, so too is $y(t) = \alpha y_1(t) + \beta y_2(t)$ for any scalars α, β.

A.1 Homogeneous ODE

An ODE is *homogeneous* if, and only if, the function which is identically zero is a solution. Of course, this is not likely to be the *only* solution. Looking at (A.1) we see that this is equivalent to $f(t) = 0$.

A.1.1 Second order, homogeneous

Consider $ay'' + by' + cy = 0$, a, b, c constant, $a \neq 0$. Since the equation is of second order, the general solution must be able to be written as a linear combination of *two linearly independent*

solutions. We might guess that solutions should be made out of exponentials since we seek solution y which when differentiated is *comparable* to the solution y itself. If we "try" $y = e^{rt}$ we find

$$ar^2e^{rt} + bre^{rt} + ce^{rt} = 0, \qquad \text{so} \qquad ar^2 + br + c = 0.$$

The polynomial left-hand-side is called the **auxiliary polynomial** (the equation is often called the characteristic equation) for the ODE. Of course this equation has solutions

$$r = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}.$$

We will see that solutions vary significantly depending on the sign of $b^2 - 4ac$.

Distinct real roots: The general solution is $y = c_1e^{r_1t} + c_2e^{r_2t}$.

EXAMPLE A.1. The ODE $y'' + 5y' + 6y = 0$ has characteristic equation $r^2 + 5r + 6 = (r + 3)(r + 2) = 0$. So $y_1(t) = e^{-2t}$ and $y_2(t) = e^{-3t}$ are both solutions and the general solution is $y(t) = c_1e^{-2t} + c_1e^{-3t}$.

One repeated real root: We have one solution e^{rt}. A second independent solution is found to be te^{rt}.

EXAMPLE A.2. The ODE $y'' - y' + \frac{1}{4}y = 0$ has auxiliary equation with the repeated root $r = \frac{1}{2}$. Thus $y_1(t) = e^{t/2}$ and $y_2(t) = te^{t/2}$ are independent solutions.

Complex (conjugate) roots: As before we obtain exponential solutions, but this time they are "complex-exponential" solutions. The roots are of course

$$r = \frac{-b \pm \sqrt{4ac - b^2}i}{2a} = -\frac{b}{2a} \pm \frac{\sqrt{4ac - b^2}}{2a}i = \alpha \pm \beta i, \text{ say,}$$

which shows that they (always) occur in *complex conjugate pairs*, i.e. $r_1 = \alpha + \beta i$, $r_2 = \alpha - \beta i$. This means that we have the *complex valued* solutions

$$z_1(t) = e^{(\alpha+i\beta)t} = e^{\alpha t}e^{i\beta t} = e^{\alpha t}\big(\cos(\beta t) + i\sin(\beta t)\big)$$
$$z_2(t) = e^{(\alpha-i\beta)t} = e^{\alpha t}e^{-i\beta t} = e^{\alpha t}\big(\cos(\beta t) - i\sin(\beta t)\big).$$

It is a (readily checked) fact that if $z(t) = p(t) + iq(t)$ solves $az'' + bz' + c = 0$, then each of $p(t)$ and $q(t)$ solve the same equation. Thus we have solutions

$$y_1(t) = e^{\alpha t}\cos(\beta t) \qquad y_2(t) = e^{\alpha t}\sin(\beta t).$$

Thus the general solution is

$$y(t) = c_1y_1(t) + c_2y_2(t) = e^{\alpha t}\big(c_1\cos(\beta t) + c_2\sin(\beta t)\big).$$

EXAMPLE A.3. Consider $y'' + y' + y = 0$ with $y(0) = 2,\ y'(0) = -3$. We find $r = -\frac{1}{2} \pm i\frac{\sqrt{3}}{2}$ and so

$$y(t) = e^{-\frac{1}{2}t}\big(c_1 \cos(\sqrt{3}t/2) + c_2 \sin(\sqrt{3}t/2)\big).$$

At $t = 0$, $y(0) = 2 = c_1$. Next,

$$\begin{aligned} y'(t) = &-\frac{1}{2}e^{-\frac{1}{2}t}\big(c_1 \cos(\sqrt{3}t/2) + c_2 \sin(\sqrt{3}t/2)\big) \\ &+ (\sqrt{3}/2)e^{-\frac{1}{2}t}\big(-c_1 \sin(\sqrt{3}t/2) + c_2 \cos(\sqrt{3}t/2)\big) \end{aligned}$$

so $y'(0) = -3 = -c_1/2 + \sqrt{3}c_2/2 = -1 + \sqrt{3}c_2/2$ gives $c_2 = -1/\sqrt{3}$. □

EXAMPLE A.4. The ODE $y'' + 9y = 0$ gives $r^2 + 9 = 0$ so $r = \pm 3i$ and the general solution is $y(t) = c_1 \cos(3t) + c_2 \sin(3t)$.

REMARK. Notice that we came up with two complex valued solutions; the real and imaginary parts of each solution yields a real valued solution. One might think that this gives *four* solutions – but there are only *two independent* solutions (which is all we need).

A.1.2 Higher order, homogeneous

Now consider

$$a_n y^{(n)}(t) + a_{n-1} y^{(n-1)}(t) + \cdots + a_1 y'(t) + a_0 y(t) = 0. \tag{A.2}$$

Once again, we are lead to consider exponential solutions $y = e^{rt}$; upon substitution into (A.2) we obtain the auxiliary polynomial

$$a_n r^n + a_{n-1} r^{n-1} + \cdots + a_1 r + a_0.$$

Allowing complex and repeated roots, this polynomial has roots $r_1, \ldots, r_n$, and the (possibly complex valued) functions $e^{r_j t}$ are solutions to the ODE (A.2). From each pair of complex-conjugate roots we obtain *two* linearly independent solutions, as above. From each *distinct* real root, we obtain one solution. If a real root r has multiplicity k (is repeated k times), then we obtain k linearly independent solutions

$$e^{rt}, \quad te^{rt}, \quad \cdots, \quad t^{k-1}e^{rt}.$$

Combining these, we thus obtain n linearly independent solutions, and hence the general solution as an arbitrary linear combination of these.

EXAMPLE A.5. Consider $y^{(5)}(t) - 2y^{(4)}(t) + y^{(3)}(t) - 2y''(t) = 0$. This has auxiliary polynomial which factors as $r^2(r^2 + 1)(r - 2)$. Thus $y(t) = e^{0t} = 1$, $y(t) = te^{0t} = t$, $y(t) = e^{it} = \cos t + i \sin t$ and $y(t) = e^{2t}$ are all solutions. From the complex valued solution we obtain the two real-valued solutions $y(t) = \cos t$ and $y(t) = \sin t$. Finally, the general solution is

$$y(t) = c_1 + c_2 t + c_3 \cos t + c_4 \sin t + c_5 e^{2t}.$$

A.2 Non-homogeneous ODE

When the right-hand-side is non-zero, it is no longer necessarily true that a linear combination of solutions is again a solution. However the following important result holds.

Suppose that $y_p(t)$ is *any* "particular" solution to

$$a_n y_p^{(n)}(t) + a_{n-1} y_p^{(n-1)}(t) + \cdots + a_1 y_p'(t) + a_0 y_p(t) = f(t),$$

and that $y_h(t)$ is the *general solution* to the corresponding *homogeneous* ODE

$$a_n y_h^{(n)}(t) + a_{n-1} y_h^{(n-1)}(t) + \cdots + a_1 y_h'(t) + a_0 y_h(t) = 0.$$

Then the general solution to the non-homogeneous ODE is

$$y(t) = y_h(t) + y_p(t).$$

So, finding the general solution to a non-homogeneous (linear, constant coefficient) ODE comes down to (1) finding the general solution to the corresponding homogeneous ODE (which we know how to do from the above), and (2) finding just one solution to the non-homogeneous ODE. It is (2) which we briefly address here. The method of finding a particular solution is known as the method of undetermined coefficients. Its applicability is restricted to a small set of right-hand-side (RHS) functions $f(t)$, and the approach is, essentially, to guess a solution and then determine appropriate constants to make the guess work. We tend to try solutions which are (undetermined) multiples of the RHS, adjusting by multiplying by $t, t^2, \ldots$ if we happen to coincide with one of the solutions to the homogeneous problem.

In what follows, when we refer to *roots*, we mean the roots of the auxiliary polynomial for the *homogeneous* problem. We present the solution for roots with multiplicity less than or equal to 2; these extend for higher multiplicity roots in the obvious way.

1. If the RHS is an exponential:

$$f(t) = Ae^{\beta t} \qquad \Rightarrow \qquad y_p(t) = \begin{cases} Ce^{\beta t}, & \text{if } \beta \text{ is not a real root,} \\ Cte^{\beta t}, & \text{if } \beta \text{ is a distinct real root,} \\ Ct^2 e^{\beta t}, & \text{if } \beta \text{ is a repeated real root,} \\ \cdots, & \text{if } \beta \text{ is a root of higher multiplicity.} \end{cases}$$

   To find the constant C, we substitute $y_p(t)$ into the non-homogeneous ODE.

   EXAMPLE A.6. Consider $y'' + 5y' + 6y = f(t)$. The homogeneous problem has auxiliary polynomial $r^2 + 5r + 6 = (r+3)(r+2)$, so $y_1(t) = e^{-2t}$ and $y_2(t) = e^{-3t}$ are independent solutions to the homogeneous problem.

(a) if $f(t) = e^{-t}$, we try $y_p(t) = Ce^{-t}$ for unknown C. Plugging in we find that $2Ce^{-t} = e^{-t}$ and so we set $C = 1/2$. The general solution is then

$$y(t) = y_h(t) + y_p(t) = c_1e^{-2t} + c_2e^{-3t} + \frac{1}{2}e^{-t};$$

(b) if $f(t) = e^{-2t}$, we cannot try $y_p(t) = Ce^{-2t}$ since plugging this into the ODE will give 0, not $f(t)$. Instead, we try $y_p(t) = te^{-2t}$, and upon substitution and some algebra, $C = 1$.

2. If the RHS is a polynomial $p_n(t)$, let n be the *order* of the polynomial; then

$$y_p(t) = \begin{cases} q_n(t), \text{ a polynomial of the same order,} & \text{if } 0 \text{ is not a real root,} \\ tq_n(t), \text{ a polynomial of the order } n+1, & \text{if } 0 \text{ is a distinct real root,} \\ t^2q_n(t), \text{ a polynomial of the order } n+2, & \text{if } 0 \text{ is a repeated real root,} \\ \cdots, & \text{if } 0 \text{ is a root of higher multiplicity.} \end{cases}$$

where $q_n(t) = a_0 + a_1t + \cdots + a_nt^n$, the coefficients a_j to be determined.

EXAMPLE A.7. Consider again $y'' + 5y' + 6y = f(t)$. The homogeneous problem has auxiliary polynomial $r^2+5r+6 = (r+3)(r+2) = 0$. If $f(t) = 1+t$, we try $y_p = a+bt$ and find that $6a + 5b + 6bt = 1 + t$ which determines a and b.

EXAMPLE A.8. Consider $y'' - 2y' = 1 + t$. Here $y_1 = 1$, $y_2 = e^{2t}$. If we try $y_p = a + bt$ again, we find $-2b = 1 + t$ which clearly cannot hold (for all t). We need to try $y_p = at + bt^2$ which yields $-2a + 2b - 4bt = 1 + t$ which this time determines a and b.

3. If the RHS is (exponential times) trigonometric,

$$f(t) = e^{\alpha t}(A\cos\beta t + B\sin\beta t)$$

(some of α, A, B may be zero, but $\beta \neq 0$),

$$y_p(t) = \begin{cases} e^{\alpha t}(C_1\cos\beta t + C_2\sin\beta t), & \text{if } \alpha+\beta i \text{ is not a root,} \\ te^{\alpha t}(C_1\cos\beta t + C_2\sin\beta t), & \text{if } \alpha+\beta i \text{ is a distinct root,} \\ t^2e^{\alpha t}(C_1\cos\beta t + C_2\sin\beta t), & \text{if } \alpha+\beta i \text{ is a repeated root,} \\ \cdots, & \text{if } \alpha+\beta i \text{ is a root of higher multiplicity.} \end{cases}$$

Appendix B

Mathematical Background

Here is a collection of material we will need throughout the course. Some of it is material you have seen (I have indicated the classes where you might have seen it) and some of it is new. We will address the new material as we need it.

B.1 Vector Space Notions

B.1.1. (204) Linear combination of vectors.

B.1.2. (204) Linear independence of vectors.

B.1.3. (204) Subspace.

B.1.4. (204) Span of a set of vectors.

B.1.5. (204) Basis for a vector space or subspace.

B.1.6. (204) Dimension of a vector space or subspace.

B.1.7. (204) $\mathbb{R}^n$ and $\mathbb{C}^n$.

B.2 Matrix Notions

B.2.1. (204) Rank of an $m \times n$ matrix.

B.2.2. (204) Determinant of a square matrix.

B.2.3. A pth order minor – rank equals the highest order of the nonzero minor.

B.2.4. (204) Invertibility of square matrices (non-singular).

B.2.5. (204) Eigenvalues and eigenvectors of square matrices.

B.2.6. (204) The characteristic polynomial of a square matrix.

B.2.7. (304) Eigenvectors corresponding to distinct eigenvalues always independent, and are mutually orthogonal for symmetric matrices.

B.2.8. (304) Diagonalization of square matrices with a full linearly independent set of eigenvectors.

B.2.9. (204) Transpose A^T of a matrix A.

B.2.10. (304) Real symmetric matrices are diagonalizable.

B.3 Inner Products and Norms

B.3.1. (224/204/304) Euclidean inner product on $\mathbb{R}^n$, $\vec{x} \cdot \vec{y}$. This can be re-expressed as $\vec{x}^T \vec{y}$. The more general notation for an inner product is $\langle \vec{x}, \vec{y} \rangle$.

B.3.2. (304) Properties of an inner product on $\mathbb{R}^n$:

1. $\langle \vec{x}, \vec{x} \rangle \geq 0$, $\langle \vec{x}, \vec{x} \rangle = 0$ iff $\vec{x} = \vec{0}$.
2. $\langle \vec{x}, \vec{y} \rangle = \langle \vec{y}, \vec{x} \rangle$.
3. $\langle \vec{x} + \vec{y}, \vec{z} \rangle = \langle \vec{x}, \vec{z} \rangle + \langle \vec{y}, \vec{z} \rangle$.
4. $\langle \alpha \vec{x}, \vec{y} \rangle = \alpha \langle \vec{x}, \vec{y} \rangle$ for all $\alpha \in \mathbb{R}$.

B.3.3. (224/304) Orthogonality of vectors: $\langle \vec{x}, \vec{y} \rangle = 0$.

B.3.4. (224/304) Norm of vectors: $\|\vec{x}\| = \sqrt{\langle \vec{x}, \vec{x} \rangle}$. We shall use $|\vec{x}|$ for $\sqrt{\vec{x} \cdot \vec{x}}$ and $\|\vec{x}\|$ for (perhaps) other general norms.

B.3.5. (304) Cauchy-Schwartz Inequality: $\|\langle \vec{x}, \vec{y} \rangle\| \leq \|\vec{x}\| \|\vec{y}\|$.

B.3.6. (304) The norm satisfies:

1. $\|\vec{x}\| \geq 0$, $\|\vec{x}\| = 0$ iff $\vec{x} = \vec{0}$.
2. $\|\alpha \vec{x}\| = |\alpha| \|\vec{x}\|$, for all $\alpha \in \mathbb{R}$.
3. $\|\vec{x} + \vec{y}\| \leq \|\vec{x}\| + \|\vec{y}\|$ (the triangle inequality), with equality iff $\vec{x}$ and $\vec{y}$ are linearly dependent.

B.3.7. (304) If $\vec{x}$ and $\vec{y}$ are orthogonal then $\|\vec{x} + \vec{y}\|^2 = \|\vec{x}\|^2 + \|\vec{y}\|^2$.

B.3.8. Other norms exist for $\mathbb{R}^n$, for example:

$$\|\vec{x}\|_p = \begin{cases} \big(|x_1|^p + \cdots + |x_n|^p\big)^{1/p}, & 1 \le p < \infty \\ \max\big\{|x_1|, \ldots, |x_n|\big\}, & p = \infty \end{cases}$$

B.3.9. (304) Orthogonal complement $S^\perp$ of a subspace S.

B.3.10. The notion of a norm can be applied also to matrices. An example (equivalent to the Euclidean norm on $\mathbb{R}^{m\,n}$) is

$$\|A\|_2 = \Big(\sum_{i=1}^{m}\sum_{j=1}^{n} |a_{ij}|^2\Big)^{1/2}.$$

B.3.11. In addition to the conditions in B.3.6, we require $\|AB\| \le \|A\|\|B\|$.

B.4 Linear Transformations

We use the term "transformation", or "map", for functions between vector spaces (the term "function" is often reserved for maps when the *range* is contained in $\mathbb{R}$).

B.4.1. (304) $\mathcal{L} : \mathbb{R}^n \to \mathbb{R}^m$ is a linear transformation if

1. $\mathcal{L}(\alpha\vec{x}) = \alpha\mathcal{L}(\vec{x})$ for all $\vec{x} \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$.
2. $\mathcal{L}(\vec{x} + \vec{y}) = \mathcal{L}(\vec{x}) + \mathcal{L}(\vec{y})$.

B.4.2. (304) Once we specify bases for $\mathbb{R}^n$ and $\mathbb{R}^m$, a linear transformation is represented by an $m \times n$ matrix: if $\hat{x}$ is the coordinate vector of a vector $\vec{x} \in \mathbb{R}^n$ in terms of the chosen basis, and $\vec{y} = \mathcal{L}(\vec{x})$, then $\hat{y}$, the coordinate vector $\vec{y}$ in the chosen basis for $\mathbb{R}^m$, is given by $\hat{y} = A\hat{x}$.

B.4.3. (304) Given bases $\{\vec{e}_1, \ldots, \vec{e}_n\}$ and $\{\vec{u}_1, \ldots \vec{u}_n\}$ for $\mathbb{R}^n$, the matrix

$$T = \big[\vec{u}_1, \ldots \vec{u}_n\big]^{-1}\big[\vec{e}_1, \ldots, \vec{e}_n\big]$$

is the *transformation matrix* from $\{\vec{e}_1, \ldots, \vec{e}_n\}$ to $\{\vec{u}_1, \ldots \vec{u}_n\}$. Then, if $\vec{x}$ is a coordinate vector w.r.t. $\{\vec{e}_1, \ldots, \vec{e}_n\}$ and $\hat{x}$ is the coordinate vector w.r.t. $\{\vec{u}_1, \ldots \vec{u}_n\}$, it holds that $\hat{x} = T\vec{x}$.

B.4.4. (304) Orthogonal projection onto a subspace.

B.4.5. (304) Range (or image) $\mathcal{R}(\mathcal{L})$, and nullspace (or kernel) $\mathcal{N}(\mathcal{L})$.

B.4.6. (304) For a matrix A, $\mathcal{R}(A)^\perp = \mathcal{N}(A^T)$ and $\mathcal{N}(A)^\perp = \mathcal{R}(A^T)$.

B.4.7. If we have norms $\|\cdot\|_{(n)}$ and $\|\cdot\|_{(m)}$ on $\mathbb{R}^n$ and $\mathbb{R}^m$ respectively, then we define an *induced* (operator) norm on $m \times n$ matrices A to be

$$\|A\| = \max_{\|\vec{x}\|_{(n)}=1} \|A\vec{x}\|_{(m)}.$$

Thus we are seeking the "largest" vector in the image of the unit sphere under multiplication by A.

For example, if we use the Euclidean norm on $\mathbb{R}^n$, then the induced norm for $n \times n$ matrices A is

$$\|A\| = \sqrt{\lambda_1}$$

where λ_1 is the largest eigenvalue of the matrix $A^T A$.

B.5 Quadratic Forms

A *quadratic form* $f : \mathbb{R}^n \to \mathbb{R}$ is a function which can be expressed as

$$f(\vec{x}) = \vec{x}^T Q \vec{x}$$

with Q an $n \times n$ real matrix. WLOG, Q is symmetric.

B.5.1. A quadratic form is:

1. positive definite if $\vec{x}^T Q \vec{x} > 0$ for all $\vec{x} \neq 0$,
2. positive semidefinite if $\vec{x}^T Q \vec{x} \geq 0$ for all $\vec{x} \neq 0$, and
3. similarly for negative definite and negative semidefinite.

B.5.2. The "leading principal minors" of Q are the determinants of the sub-matrices comprising the top-left $j \times j$ corner of Q. **Sylvester's Criterion:** A quadratic form $\vec{x}^T Q \vec{x}$ (Q symmetric) is positive definite iff the leading principal minors of Q are positive.

B.5.3. There is another condition which gives positive *semi*-definiteness. A general "principal minor" is a determinant of the result of deleting any number $1 \leq j \leq n$ rows and columns – if row j is removed, then so too is column j. A quadratic form $\vec{x}^T Q \vec{x}$ (Q symmetric) is positive *semi*-definite iff *all* the principal minors are non-negative (greater than or equal to zero).

B.5.4. We carry the notions of (semi)definiteness of a quadratic form to matrices Q by considering the corresponding quadratic form. A matrix Q is said to be indefinite if it is neither positive- nor negative-semidefinite.

B.5.5. A symmetric matrix Q is positive (semi)definite iff all the eigenvalues of Q are positive (non-negative).

B.5.6. **Rayleigh's Inequality:** If Q is an $n \times n$ real symmetric positive definite matrix, then

$$\lambda_{\text{min}}|\vec{x}|^2 \leq \vec{x}^T Q\vec{x} \leq \lambda_{\text{max}}|\vec{x}|^2$$

where $\lambda_{\text{min}},\ \lambda_{\text{max}}$ are the minimal and maximal eigenvalues of Q.

B.6 Notions from Geometry

B.6.1. (224) The line segment between points $\vec{p}$ and $\vec{q}$ can be parameterized by $\alpha\vec{p} + (1-\alpha)\vec{q}$, $0 \leq \alpha \leq 1$.

B.6.2. (224) If $\vec{u} \neq 0$ then the set of $\vec{x}$ satisfying $\vec{x}^T\vec{u} = c$ is a "hyperplane" in $\mathbb{R}^n$. It is a subspace iff $c = 0$. The vector $\vec{u}$ is orthogonal to the hyperplane.

B.6.3. A hyperplane $H = \{\vec{x} \,:\, \vec{x}^T\vec{u} = c\}$ divides $\mathbb{R}^n$ into two "half-spaces" $H_+ = \{\vec{x} \,:\, \vec{x}^T\vec{u} \geq c\}$ and $H_- = \{\vec{x} \,:\, \vec{x}^T\vec{u} \leq c\}$.

B.6.4. The intersection of finitely many subspaces is again a subspace; the intersection of finitely many m hyperplanes is called a "linear variety". It can be understood as a "translation" of a subspace. Any linear variety can be written in the form

$$\{\vec{x} \in \mathbb{R}^n \,:\, A\vec{x} = \vec{b}\}$$

for some $m \times n$ matrix A and vector $\vec{b} \in \mathbb{R}^m$ (indeed, the columns of A are the normal vectors of the hyperplanes and the components of $\vec{b}$ are the RHS constants). If $\dim \mathcal{N}(A) = r$ then the linear variety is said to have dimension r (it is the translation of an r-dimensional subspace).

B.6.5. A set $K \subset \mathbb{R}^n$ is *convex* if for all $\vec{x}, \vec{y} \in K$ it holds that the line-segment joining $\vec{x}$ to $\vec{y}$ is entirely contained with K also: i.e.

$$\alpha\vec{x} + (1-\alpha)\vec{y} \in K \quad \text{for all } 0 \leq \alpha \leq 1 \text{ and for all } \vec{x}, \vec{y} \in K.$$

B.6.6. (226) An ε-ball around $\vec{x}$ is the set

$$B_\varepsilon(\vec{x}) = \{\vec{y} \,:\, |\vec{y} - \vec{x}| < \varepsilon\},$$

where $\varepsilon > 0$. It does not contain the "boundary".

B.6.7. (312) A set U is *open* if for every point $\vec{x} \in U$ it is possible to find $\varepsilon_{\vec{x}} > 0$ such that $B_{\varepsilon_{\vec{x}}}(\vec{x}) \subset U$. A set V is *closed* if its complement $\mathbb{R}^n \setminus V$ is open.

B.6.8. (312) A set U is *bounded* if there exists R such that $|\vec{x}| < R$ for all $\vec{x} \in U$.

B.6.9. A set $K \subset \mathbb{R}^n$ is *compact* if it is both closed and bounded.

B.6.10. **Theorem of Weierstrass:** If K is a compact set, and $f : K \to \mathbb{R}$ is continuous, then f attains its maximum and its minimum on K.

B.7 Notions of Calculus

B.7.1. (224) A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is said to be differentiable (at $\vec{x}_0$) if there is a linear transformation $\mathcal{L} : \mathbb{R}^n \to \mathbb{R}^m$ which "approximates f to first order at x_0" (we shall not bother being more precise for the moment). See B.7.2

B.7.2. A function $f : \mathbb{R}^n \to \mathbb{R}^m$ can be represented as $[f_1, \ldots, f_m]^T$ where each of $f_i : \mathbb{R}^n \to \mathbb{R}$. If f is differentiable, each f_j may be differentiated w.r.t. any of its n variables x_j. With respect to the standard bases (for $\mathbb{R}^n$ and $\mathbb{R}^m$), the linear operator in B.7.1 has the $m \times n$ matrix representation

$$Df(\vec{x}_0) = \left[\frac{\partial f_i}{\partial x_j}(\vec{x}_0)\right]_{1\le i\le m, 1\le j\le n} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix} (\vec{x}_0).$$

B.7.3. The matrix in B.7.2 is called the *Jacobian matrix* of f at $\vec{x}_0$ and is denoted $Df(\vec{x}_0)$.

B.7.4. (224) The linear (or affine) approximation to f near $\vec{x}_0$ is

$$\mathcal{A}(\vec{x}) = f(\vec{x}_0) + Df(x_0)(\vec{x} - \vec{x}_0)$$

B.7.5. (224) If $f : \mathbb{R}^n \to \mathbb{R}$ is differentiable then the *gradient* vector of f at $\vec{x}_0$ is

$$\nabla f(\vec{x}_0) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(\vec{x}_0) \\ \vdots \\ \frac{\partial f}{\partial x_n}(\vec{x}_0) \end{bmatrix} = Df(\vec{x}_0)^T.$$

B.7.6. If $f : \mathbb{R}^n \to \mathbb{R}$ is twice differentiable then the *Hessian* matrix of f is

$$D^2 f = \begin{bmatrix} f_{x_1x_1} & f_{x_2x_1} & \cdots & f_{x_nx_1} \\ f_{x_1x_2} & f_{x_2x_2} & \cdots & f_{x_nx_2} \\ \vdots & \vdots & \ddots & \vdots \\ f_{x_1x_n} & f_{x_2x_n} & \cdots & f_{x_nx_n} \end{bmatrix}.$$

B.7.7. If f is k times differentiable, and every derivative is continuous, then we say $f \in \mathcal{C}^k$.

B.7.8. (224) The *chain rule* for functions of several variables:

1. if $g : U \subset \mathbb{R}^n \to \mathbb{R}$ and $f : (a, b) \subset \mathbb{R} \to U$ are differentiable, then $F : (a, b) \to \mathbb{R}$, $F(t) = g(f(t))$ is differentiable and $F'(t) = Dg(f(t))Df(t) = \nabla g(f(t))^T Df(t)$.
2. If $g : U \subset \mathbb{R}^n_{\vec{x}} \to \mathbb{R}$ and $f : W \subset \mathbb{R}^m_{\vec{y}} \to U$ are differentiable, then $F : W \to \mathbb{R}$, $F(\vec{y}) = g(f(y))$ is differentiable and $DF(\vec{y}) = Dg(f(\vec{y}))Df(\vec{y})$.

B.7.9. (224) Gradient vectors and level sets: let $f : \mathbb{R}^n \to \mathbb{R}$ be differentiable at $\vec{x}_0$. Then

1. $\nabla f(\vec{x}_0)$ is perpendicular to (the tangent hyperplane to) the level set through $\vec{x}_0$ at x_0, and points in the direction of maximum increase of f from $\vec{x}_0$.
2. The *graph* of f is the set $\{[\vec{x}^T, f(\vec{x})]^T\} \subset \mathbb{R}^{n+1}$; if we use coordinates $[\vec{x}^T, z]^T$ for $\mathbb{R}^{n+1}$ then the tangent hyperplane to the graph of f at $\vec{x}_0$ has equation

$$z - z_0 = Df(\vec{x}_0)(\vec{x} - \vec{x}_0) = (\vec{x} - \vec{x}_0)^T \nabla f(\vec{x}_0).$$

B.7.10. (225) The divergence theorem: suppose that $\Omega \subset \mathbb{R}^n$ is compact and has a piecewise smooth boundary $\partial\Omega$. If $\vec{F}$ is a $C^1(\Omega)$ vector field (in a neighborhood of Ω), then

$$\int_\Omega \operatorname{div} \vec{F}(\vec{x})\, dV(\vec{x}) = \int_{\partial\Omega} \vec{F} \cdot d\vec{A}(\vec{x}).$$

B.7.11. Big-O and Little-o terminology: Let $f : U \subset \mathbb{R}^n \to \mathbb{R}^m$ with $\vec{0} \in U$. Then

1. f is Big-Oh of $|\vec{x}|^r$, $f(\vec{x}) = O(|\vec{x}|^r)$ means that $\dfrac{|f(\vec{x})|}{|\vec{x}|^r}$ is *bounded* near $\vec{0}$ (i.e. f must be "at least as small as" $|\vec{x}|^r$ near $\vec{0}$), and
2. f is Little-o of $|\vec{x}|^r$, $f(\vec{x}) = o(|\vec{x}|^r)$ means that $\lim\limits_{\vec{x}\to\vec{0}} \dfrac{|f(\vec{x})|}{|\vec{x}|^r} = 0$ (i.e. f goes to zero faster than $|\vec{x}|^r$ as $\vec{x} \to \vec{0}$).

B.7.12. (224) Taylor series: let $f : \mathbb{R}^n \to \mathbb{R}$;

1. If $f \in \mathcal{C}^2$ near $\vec{x}_0$ then

$$f(\vec{x}) = f(\vec{x}_0) + Df(\vec{x}_0)(\vec{x} - \vec{x}_0) + \frac{1}{2}(\vec{x} - \vec{x}_0)^T Df(\vec{x}_0)(\vec{x} - \vec{x}_0) + o(|\vec{x} - \vec{x}_0|^2).$$

2. If $f \in \mathcal{C}^3$ near $\vec{x}_0$ then

$$f(\vec{x}) = f(\vec{x}_0) + Df(\vec{x}_0)(\vec{x} - \vec{x}_0) + \frac{1}{2}(\vec{x} - \vec{x}_0)^T Df(\vec{x}_0)(\vec{x} - \vec{x}_0) + O(|\vec{x} - \vec{x}_0|^2).$$

Index