Chapter 4
The Gradient Method

4.1 ▪ Descent Directions Methods

In this chapter we consider the unconstrained minimization problem

$$\min\{f(\mathbf{x}) : \mathbf{x} \in \mathbb{R}^n\}.$$

We assume that the objective function is continuously differentiable over $\mathbb{R}^n$. We have already seen in Chapter 2 that a first order necessary optimality condition is that the gradient vanishes at optimal points, so in principle the optimal solution of the problem can be obtained by finding among all the stationary points of f the one with the minimal function value. In Chapter 2 several examples were presented in which such an approach can lead to the detection of the unconstrained global minimum of f, but unfortunately these were exceptional examples. In the majority of problems such an approach is not implementable for the following reasons: (i) it might be a very difficult task to solve the set of (usually nonlinear) equations $\nabla f(\mathbf{x}) = \mathbf{0}$; (ii) even if it is possible to find all the stationary points, it might be that there are infinite number of stationary points and the task of finding the one corresponding to the minimal function value is an optimization problem which by itself might be as difficult as the original problem. For these reasons, instead of trying to find an analytic solution to the stationarity condition, we will consider an iterative algorithm for finding stationary points.

The iterative algorithms that we will consider in this chapter take the form

$$\mathbf{x}_{k+1} = \mathbf{x}_k + t_k \mathbf{d}_k, \quad k = 0, 1, 2, \ldots,$$

where $\mathbf{d}_k$ is the so-called *direction* and t_k is the *stepsize*. We will limit ourselves to "descent directions," whose definition is now given.

Definition 4.1 (descent direction). *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function over $\mathbb{R}^n$. A vector $\mathbf{0} \neq \mathbf{d} \in \mathbb{R}^n$ is called* **a descent direction** *of f at $\mathbf{x}$ if the directional derivative $f'(\mathbf{x}; \mathbf{d})$ is negative, meaning that*

$$f'(\mathbf{x}; \mathbf{d}) = \nabla f(\mathbf{x})^T \mathbf{d} < 0.$$

The most important property of descent directions is that taking small enough steps along these directions lead to a decrease of the objective function.

Lemma 4.2 (descent property of descent directions). *Let f be a continuously differentiable function over an open set U, and let $\mathbf{x} \in U$. Suppose that $\mathbf{d}$ is a descent direction of f at $\mathbf{x}$. Then there exists $\varepsilon > 0$ such that*

$$f(\mathbf{x} + t\mathbf{d}) < f(\mathbf{x})$$

for any $t \in (0, \varepsilon]$.

Proof. Since $f'(\mathbf{x}; \mathbf{d}) < 0$, it follows from the definition of the directional derivative that

$$\lim_{t \to 0^+} \frac{f(\mathbf{x} + t\mathbf{d}) - f(\mathbf{x})}{t} = f'(\mathbf{x}; \mathbf{d}) < 0.$$

Therefore, there exists an $\varepsilon > 0$ such that

$$\frac{f(\mathbf{x} + t\mathbf{d}) - f(\mathbf{x})}{t} < 0$$

for any $t \in (0, \varepsilon]$, which readily implies the desired result. □

We are now ready to write in a schematic way a general descent directions method.

Schematic Descent Directions Method

Initialization: Pick $\mathbf{x}_0 \in \mathbb{R}^n$ arbitrarily.
General step: For any $k = 0, 1, 2, \ldots$ set

(a) Pick a descent direction $\mathbf{d}_k$.

(b) Find a stepsize t_k satisfying $f(\mathbf{x}_k + t_k \mathbf{d}_k) < f(\mathbf{x}_k)$.

(c) Set $\mathbf{x}_{k+1} = \mathbf{x}_k + t_k \mathbf{d}_k$.

(d) If a stopping criterion is satisfied, then STOP and $\mathbf{x}_{k+1}$ is the output.

Of course, many details are missing in the above description of the schematic algorithm:

- What is the starting point?
- How to choose the descent direction?
- What stepsize should be taken?
- What is the stopping criteria?

Without specification of these missing details, the descent direction method remains "conceptual" and cannot be implemented. The initial starting point can be chosen arbitrarily (in the absence of an educated guess for the optimal solution). The main difference between different methods is the choice of the descent direction, and in this chapter we will elaborate on one of these choices. An example of a popular stopping criteria is $\|\nabla f(\mathbf{x}_{k+1})\| \leq \varepsilon$. We will assume that the stepsize is chosen in such a way such that $f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k)$. This means that the method is assumed to be a *descent method*, that is, a method in which the function values decrease from iteration to iteration. The process of finding the stepsize t_k is called *line search*, since it is essentially a minimization

procedure on the one-dimensional function $g(t) = f(\mathbf{x}_k + t\mathbf{d}_k)$. There are many choices for stepsize selection rules. We describe here three popular choices:

- **constant stepsize** $t_k = \bar{t}$ for any k.
- **exact line search** t_k is a minimizer of f along the ray $\mathbf{x}_k + t\mathbf{d}_k$:
$$t_k \in \operatorname{argmin}_{t\geq 0} f(\mathbf{x}_k + t\mathbf{d}_k).$$
- **backtracking** The method requires three parameters: $s > 0, \alpha \in (0,1), \beta \in (0,1)$. The choice of t_k is done by the following procedure. First, t_k is set to be equal to the initial guess s. Then, while
$$f(\mathbf{x}_k) - f(\mathbf{x}_k + t_k\mathbf{d}_k) < -\alpha t_k \nabla f(\mathbf{x}_k)^T \mathbf{d}_k,$$
we set $t_k \leftarrow \beta t_k$. In other words, the stepsize is chosen as $t_k = s\beta^{i_k}$, where i_k is the smallest nonnegative integer for which the condition
$$f(\mathbf{x}_k) - f(\mathbf{x}_k + s\beta^{i_k}\mathbf{d}_k) \geq -\alpha s\beta^{i_k} \nabla f(\mathbf{x}_k)^T \mathbf{d}_k$$
is satisfied.

The main advantage of the constant stepsize strategy is of course its simplicity, but at this point it is unclear how to choose the constant. A large constant might cause the algorithm to be nondecreasing, and a small constant can cause slow convergence of the method. The option of exact line search seems more attractive from a first glance, but it is not always possible to actually find the exact minimizer. The third option is in a sense a compromise between the latter two approaches. It does not perform an exact line search procedure, but it does find a good enough stepsize, where the meaning of "good enough" is that it satisfies the following sufficient decrease condition:

$$f(\mathbf{x}_k) - f(\mathbf{x}_k + t_k\mathbf{d}_k) \geq -\alpha t_k \nabla f(\mathbf{x}_k)^T \mathbf{d}_k. \tag{4.1}$$

The next result shows that the sufficient decrease condition (4.1) is always satisfied for small enough t_k.

Lemma 4.3 (validity of the sufficient decrease condition). *Let f be a continuously differentiable function over $\mathbb{R}^n$, and let $\mathbf{x} \in \mathbb{R}^n$. Suppose that $0 \neq \mathbf{d} \in \mathbb{R}^n$ is a descent direction of f at $\mathbf{x}$ and let $\alpha \in (0,1)$. Then there exists $\varepsilon > 0$ such that the inequality*

$$f(\mathbf{x}) - f(\mathbf{x} + t\mathbf{d}) \geq -\alpha t \nabla f(\mathbf{x})^T \mathbf{d}$$

holds for all $t \in [0, \varepsilon]$.

Proof. Since f is continuously differentiable it follows that (see Proposition 1.23)

$$f(\mathbf{x} + t\mathbf{d}) = f(\mathbf{x}) + t\nabla f(\mathbf{x})^T \mathbf{d} + o(t\|\mathbf{d}\|),$$

and hence

$$f(\mathbf{x}) - f(\mathbf{x} + t\mathbf{d}) = -\alpha t \nabla f(\mathbf{x})^T \mathbf{d} - (1-\alpha)t\nabla f(\mathbf{x})^T \mathbf{d} - o(t\|\mathbf{d}\|). \tag{4.2}$$

Since $\mathbf{d}$ is a descent direction of f at $\mathbf{x}$ we have

$$\lim_{t\to 0^+} \frac{(1-\alpha)t\nabla f(\mathbf{x})^T \mathbf{d} + o(t\|\mathbf{d}\|)}{t} = (1-\alpha)\nabla f(\mathbf{x})^T \mathbf{d} < 0.$$

Hence, there exists $\varepsilon > 0$ such that for all $t \in (0, \varepsilon]$ the inequality

$$(1-\alpha)t\nabla f(\mathbf{x})^T\mathbf{d} + o(t\|\mathbf{d}\|) < 0$$

holds, which combined with (4.2) implies the desired result. ∎

We will now show that for a quadratic function, the exact line stepsize can be easily computed.

Example 4.4 (exact line search for quadratic functions). Let $f(\mathbf{x}) = \mathbf{x}^T\mathbf{A}\mathbf{x} + 2\mathbf{b}^T\mathbf{x} + c$, where $\mathbf{A}$ is an $n \times n$ positive definite matrix, $\mathbf{b} \in \mathbb{R}^n$, and $c \in \mathbb{R}$. Let $\mathbf{x} \in \mathbb{R}^n$ and let $\mathbf{d} \in \mathbb{R}^n$ be a descent direction of f at $\mathbf{x}$. We will find an explicit formula for the stepsize generated by exact line search, that is, an expression for the solution of

$$\min_{t \geq 0} f(\mathbf{x} + t\mathbf{d}).$$

We have

$$\begin{aligned} g(t) = f(\mathbf{x} + t\mathbf{d}) &= (\mathbf{x} + t\mathbf{d})^T\mathbf{A}(\mathbf{x} + t\mathbf{d}) + 2\mathbf{b}^T(\mathbf{x} + t\mathbf{d}) + c \\ &= (\mathbf{d}^T\mathbf{A}\mathbf{d})t^2 + 2(\mathbf{d}^T\mathbf{A}\mathbf{x} + \mathbf{d}^T\mathbf{b})t + \mathbf{x}^T\mathbf{A}\mathbf{x} + 2\mathbf{b}^T\mathbf{x} + c \\ &= (\mathbf{d}^T\mathbf{A}\mathbf{d})t^2 + 2(\mathbf{d}^T\mathbf{A}\mathbf{x} + \mathbf{d}^T\mathbf{b})t + f(\mathbf{x}). \end{aligned}$$

Since $g'(t) = 2(\mathbf{d}^T\mathbf{A}\mathbf{d})t + 2\mathbf{d}^T(\mathbf{A}\mathbf{x} + \mathbf{b})$ and $\nabla f(\mathbf{x}) = 2(\mathbf{A}\mathbf{x} + \mathbf{b})$, it follows that $g'(t) = 0$ if and only if

$$t = \bar{t} \equiv -\frac{\mathbf{d}^T\nabla f(\mathbf{x})}{2\mathbf{d}^T\mathbf{A}\mathbf{d}}.$$

Since $\mathbf{d}$ is a descent direction of f at $\mathbf{x}$, it follows that $\mathbf{d}^T\nabla f(\mathbf{x}) < 0$ and hence $\bar{t} > 0$, which implies that the stepsize dictated by the exact line search rule is

$$\bar{t} = -\frac{\mathbf{d}^T\nabla f(\mathbf{x})}{2\mathbf{d}^T\mathbf{A}\mathbf{d}}. \tag{4.3}$$

Note that we have implicitly used in the above analysis the fact that the second order derivative of g is always positive. ∎

4.2 ▪ The Gradient Method

In the gradient method the descent direction is chosen to be the minus of the gradient at the current point: $\mathbf{d}_k = -\nabla f(\mathbf{x}_k)$. The fact that this is a descent direction whenever $\nabla f(\mathbf{x}_k) \neq 0$ can be easily shown—just note that

$$f'(\mathbf{x}_k; -\nabla f(\mathbf{x}_k)) = -\nabla f(\mathbf{x}_k)^T\nabla f(\mathbf{x}_k) = -\|\nabla f(\mathbf{x}_k)\|^2 < 0.$$

In addition for being a descent direction, minus the gradient is also the steepest direction method. This means that the normalized direction $-\nabla f(\mathbf{x}_k)/\|\nabla f(\mathbf{x}_k)\|$ corresponds to the minimal directional derivative among all normalized directions. A formal statement and proof is given in the following result.

Lemma 4.5. *Let f be a continuously differentiable function, and let $\mathbf{x} \in \mathbb{R}^n$ be a nonstationary point ($\nabla f(\mathbf{x}) \neq 0$). Then an optimal solution of*

$$\min_{\mathbf{d} \in \mathbb{R}^n}\{f'(\mathbf{x}; \mathbf{d}) : \|\mathbf{d}\| = 1\} \tag{4.4}$$

is $\mathbf{d} = -\frac{\nabla f(\mathbf{x})}{\|\nabla f(\mathbf{x})\|}$.

Proof. Since $f'(\mathbf{x};\mathbf{d}) = \nabla f(\mathbf{x})^T\mathbf{d}$, problem (4.4) is the same as

$$\min_{\mathbf{d}\in\mathbb{R}^n}\{\nabla f(\mathbf{x})^T\mathbf{d} : ||\mathbf{d}|| = 1\}.$$

By the Cauchy–Schwarz inequality we have

$$\nabla f(\mathbf{x})^T\mathbf{d} \geq -||\nabla f(\mathbf{x})|| \cdot ||\mathbf{d}|| = -||\nabla f(\mathbf{x})||.$$

Thus, $-||\nabla f(\mathbf{x})||$ is a lower bound on the optimal value of (4.4). On the other hand, plugging $\mathbf{d} = -\frac{\nabla f(\mathbf{x})}{||\nabla f(\mathbf{x})||}$ in the objective function of (4.4) we obtain that

$$f'\left(\mathbf{x}, -\frac{\nabla f(\mathbf{x})}{||\nabla f(\mathbf{x})||}\right) = -\nabla f(\mathbf{x})^T\left(\frac{\nabla f(\mathbf{x})}{||\nabla f(\mathbf{x})||}\right) = -||\nabla f(\mathbf{x})||,$$

and we thus come to the conclusion that the lower bound $-||\nabla f(\mathbf{x})||$ is attained at $\mathbf{d} = -\frac{\nabla f(\mathbf{x})}{||\nabla f(\mathbf{x})||}$, which readily implies that this is an optimal solution of (4.4). □

We will now present the gradient method with the standard stopping criteria, which is the condition $||\nabla f(\mathbf{x}_{k+1})|| \leq \varepsilon$.

The Gradient Method

Input: $\varepsilon > 0$ - tolerance parameter.

Initialization: Pick $\mathbf{x}_0 \in \mathbb{R}^n$ arbitrarily.
General step: For any $k = 0, 1, 2, \ldots$ execute the following steps:

(a) Pick a stepsize t_k by a line search procedure on the function

$$g(t) = f(\mathbf{x}_k - t\nabla f(\mathbf{x}_k)).$$

(b) Set $\mathbf{x}_{k+1} = \mathbf{x}_k - t_k\nabla f(\mathbf{x}_k)$.

(c) If $||\nabla f(\mathbf{x}_{k+1})|| \leq \varepsilon$, then STOP and $\mathbf{x}_{k+1}$ is the output.

Example 4.6 (exact line search). Consider the two-dimensional minimization problem

$$\min_{x,y}\ x^2 + 2y^2 \tag{4.5}$$

whose optimal solution is $(x, y) = (0, 0)$ with corresponding optimal value 0. To invoke the gradient method with stepsize chosen by exact line search, we constructed a MATLAB function, called `gradient_method_quadratic`, that finds (up to a tolerance) the optimal solution of a quadratic problem of the form

$$\min_{\mathbf{x}\in\mathbb{R}^n}\left\{\mathbf{x}^T\mathbf{A}\mathbf{x} + 2\mathbf{b}^T\mathbf{x}\right\},$$

where $\mathbf{A} \in \mathbb{R}^{n\times n}$ positive definite and $\mathbf{b} \in \mathbb{R}^n$. The function invokes the gradient method with exact line search, which by Example 4.4 corresponds to the stepsize $t_k = \frac{||\nabla f(\mathbf{x}_k)||^2}{2\nabla f(\mathbf{x}_k)^T\mathbf{A}\nabla f(\mathbf{x}_k)}$. The MATLAB function is described below.

```
function [x,fun_val]=gradient_method_quadratic(A,b,x0,epsilon)
% INPUT
% ======================
% A ....... the positive definite matrix associated with the
%           objective function
% b ....... a column vector associated with the linear part of the
%           objective function
% x0 ...... starting point of the method
% epsilon . tolerance parameter
% OUTPUT
% =======================
% x ....... an optimal solution (up to a tolerance) of
%           min(x^T A x+2 b^T x)
% fun_val . the optimal function value up to a tolerance


x=x0;
iter=0;
grad=2*(A*x+b);
while (norm(grad)>epsilon)
    iter=iter+1;
    t=norm(grad)^2/(2*grad'*A*grad);
    x=x-t*grad;
    grad=2*(A*x+b);
    fun_val=x'*A*x+2*b'*x;
    fprintf('iter_number = %3d norm_grad = %2.6f fun_val = %2.6f\n',...
    iter,norm(grad),fun_val);
end
```

In order to solve problem (4.5) using the gradient method with exact line search, tolerance parameter $\varepsilon = 10^{-5}$, and initial vector $\mathbf{x}_0 = (2,1)^T$, the following MATLAB command was executed:

```
[x,fun_val]=gradient_method_quadratic([1,0;0,2],[0;0],[2;1],1e-5)
```

The output is

```
iter_number =   1 norm_grad = 1.885618 fun_val = 0.666667
iter_number =   2 norm_grad = 0.628539 fun_val = 0.074074
iter_number =   3 norm_grad = 0.209513 fun_val = 0.008230
iter_number =   4 norm_grad = 0.069838 fun_val = 0.000914
iter_number =   5 norm_grad = 0.023279 fun_val = 0.000102
iter_number =   6 norm_grad = 0.007760 fun_val = 0.000011
iter_number =   7 norm_grad = 0.002587 fun_val = 0.000001
iter_number =   8 norm_grad = 0.000862 fun_val = 0.000000
iter_number =   9 norm_grad = 0.000287 fun_val = 0.000000
iter_number =  10 norm_grad = 0.000096 fun_val = 0.000000
iter_number =  11 norm_grad = 0.000032 fun_val = 0.000000
iter_number =  12 norm_grad = 0.000011 fun_val = 0.000000
iter_number =  13 norm_grad = 0.000004 fun_val = 0.000000
```

The method therefore stopped after 13 iterations with a solution which is pretty close to the optimal value:

```
x =
  1.0e-005 *
    0.1254
   -0.0627
```

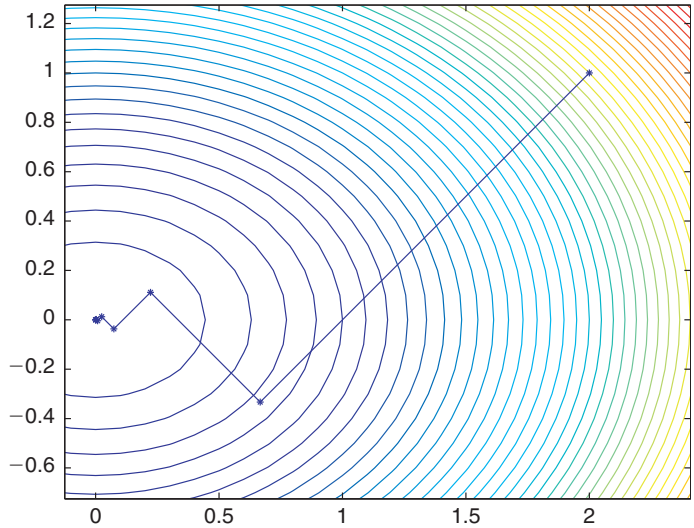

Figure 4.1. *The iterates of the gradient method along with the contour lines of the objective function.*

It is also very informative to visually look at the progress of the iterates. The iterates and the contour plots of the objective function are given in Figure 4.1. ∎

An evident behavior of the gradient method as illustrated in Figure 4.1 is the "zig-zag" effect, meaning that the direction found at the kth iteration $\mathbf{x}_{k+1}-\mathbf{x}_k$ is orthogonal to the direction found at the $(k+1)$th iteration $\mathbf{x}_{k+2}-\mathbf{x}_{k+1}$. This is a general property whose proof will be given now.

Lemma 4.7. *Let $\{\mathbf{x}_k\}_{k\geq 0}$ be the sequence generated by the gradient method with exact line search for solving a problem of minimizing a continuously differentiable function f. Then for any $k=0,1,2,\ldots$*

$$(\mathbf{x}_{k+2}-\mathbf{x}_{k+1})^T(\mathbf{x}_{k+1}-\mathbf{x}_k)=0.$$

Proof. By the definition of the gradient method we have that $\mathbf{x}_{k+1}-\mathbf{x}_k=-t_k\nabla f(\mathbf{x}_k)$ and $\mathbf{x}_{k+2}-\mathbf{x}_{k+1}=-t_{k+1}\nabla f(\mathbf{x}_{k+1})$. Therefore, we wish to prove that $\nabla f(\mathbf{x}_k)^T\nabla f(\mathbf{x}_{k+1})=0$. Since

$$t_k\in \operatorname{argmin}_{t\geq 0}\{g(t)\equiv f(\mathbf{x}_k-t\nabla f(\mathbf{x}_k))\},$$

and the optimal solution is not $t_k=0$, it follows that $g'(t_k)=0$. Hence,

$$-\nabla f(\mathbf{x}_k)^T\nabla f(\mathbf{x}_k-t_k\nabla f(\mathbf{x}_k))=0,$$

meaning that the desired result $\nabla f(\mathbf{x}_k)^T\nabla f(\mathbf{x}_{k+1})=0$ holds. □

Let us now consider an example with a constant stepsize.

Example 4.8 (constant stepsize). Consider the same optimization problem given in Example 4.6

$$\min_{x,y}\ x^2+2y^2.$$

To solve this problem using the gradient method with a constant stepsize, we use the following MATLAB function that employs the gradient method with a constant stepsize for an arbitrary objective function.

```
function [x,fun_val]=gradient_method_constant(f,g,x0,t,epsilon)
% Gradient method with constant stepsize
%
% INPUT
%=======================================
% f ......... objective function
% g ......... gradient of the objective function
% x0......... initial point
% t ......... constant stepsize
% epsilon ... tolerance parameter
% OUTPUT
%=======================================
% x ......... optimal solution (up to a tolerance)
%             of min f(x)
% fun_val ... optimal function value
x=x0;
grad=g(x);
iter=0;
while (norm(grad)>epsilon)
    iter=iter+1;
    x=x-t*grad;
    fun_val=f(x);
    grad=g(x);
    fprintf('iter_number = %3d norm_grad = %2.6f fun_val = %2.6f \n',...
    iter,norm(grad),fun_val);
end
```

We can employ the gradient method with constant stepsize $t_k = 0.1$ and initial vector $\mathbf{x}_0 = (2,1)^T$ by executing the MATLAB commands

```
A=[1,0;0,2];
[x,fun_val]=gradient_method_constant(@(x)x'*A*x,@(x)2*A*x,[2;1],0.1,1e-5);
```

and the long output is

```
iter_number =   1 norm_grad = 4.000000 fun_val = 3.280000
iter_number =   2 norm_grad = 2.937210 fun_val = 1.897600
iter_number =   3 norm_grad = 2.222791 fun_val = 1.141888
      :                  :                    :
iter_number =  56 norm_grad = 0.000015 fun_val = 0.000000
iter_number =  57 norm_grad = 0.000012 fun_val = 0.000000
iter_number =  58 norm_grad = 0.000010 fun_val = 0.000000
```

The excessive number of iterations is due to the fact that the stepsize was chosen to be too small. However, taking a stepsize which is large might lead to divergence of the iterates. For example, taking the constant stepsize to be 100 results in a divergent sequence.

```
>> A=[1,0;0,2];
>> [x,fun_val]=gradient_method_constant(@(x)x'*A*x,@(x)2*A*x,[2;1],100,1e-5);
iter_number =   1 norm_grad = 1783.488716 fun_val = 476806.000000
iter_number =   2 norm_grad = 656209.693339 fun_val = 56962873606.000000
iter_number =   3 norm_grad = 256032703.004797 fun_val = 8318300807190406.0
      :                  :                                    :
iter_number = 119 norm_grad = NaN fun_val = NaN
```

The important question is therefore how to choose the constant stepsize so that it will not be too large (to ensure convergence) and not too small (to ensure that the convergence will not be too slow). We will consider again the theoretical issue of choosing the constant stepsize in Section 4.7. ∎

Let us now consider an example with a backtracking stepsize selection rule.

Example 4.9 (stepsize selection by backtracking). The following MATLAB function implements the gradient method with a backtracking stepsize selection rule.

```
function [x,fun_val]=gradient_method_backtracking(f,g,x0,s,alpha,...
  beta,epsilon)
% Gradient method with backtracking stepsize rule
%
% INPUT
%=======================================
% f ......... objective function
% g ......... gradient of the objective function
% x0......... initial point
% s ......... initial choice of stepsize
% alpha ..... tolerance parameter for the stepsize selection
% beta ...... the constant in which the stepsize is multiplied
%             at each backtracking step (0<beta<1)
% epsilon ... tolerance parameter for stopping rule
% OUTPUT
%=======================================
% x ......... optimal solution (up to a tolerance)
%             of min f(x)
% fun_val ... optimal function value
x=x0;
grad=g(x);
fun_val=f(x);
iter=0;
while (norm(grad)>epsilon)
    iter=iter+1;
    t=s;
    while (fun_val-f(x-t*grad)<alpha*t*norm(grad)^2)
        t=beta*t;
    end
    x=x-t*grad;
    fun_val=f(x);
    grad=g(x);
    fprintf('iter_number = %3d norm_grad = %2.6f fun_val = %2.6f \n',...
    iter,norm(grad),fun_val);
end
```

As in the previous examples, we will consider the problem of minimizing the function x^2+2y^2. Employing the gradient method with backtracking stepsize selection rule, a starting vector $\mathbf{x}_0=(2,1)^T$ and parameters $\varepsilon=10^{-5}, s=2, \alpha=\frac{1}{4}, \beta=\frac{1}{2}$ results in the following output:

```
>> A=[1,0;0,2];
>> [x,fun_val]=gradient_method_backtracking(@(x)x'*A*x,@(x)2*A*x,...
[2;1],2,0.25,0.5,1e-5);
iter_number =   1 norm_grad = 2.000000 fun_val = 1.000000
iter_number =   2 norm_grad = 0.000000 fun_val = 0.000000
```

That is, the gradient method with backtracking terminated after only 2 iterations. In fact, in this case (and this is probably a result of pure luck), it converged to the *exact*

optimal solution. In this example there was no advantage in performing an exact line search procedure, and in fact better results were obtained by the nonexact/backtracking line search. Computational experience teaches us that in practice backtracking does not have real disadvantages in comparison to exact line search.

The gradient method can behave quite badly. As an example, consider the minimization problem

$$\min_{x,y} x^2 + \frac{1}{100}y^2,$$

and suppose that we employ the backtracking gradient method with initial vector $(\frac{1}{100}, 1)^T$:

```
>> A=[1,0;0,0.01];
>> [x,fun_val]=gradient_method_backtracking(@(x)x'*A*x,@(x)2*A*x,...
[0.01;1],2,0.25,0.5,1e-5);
iter_number =   1 norm_grad = 0.028003 fun_val = 0.009704
iter_number =   2 norm_grad = 0.027730 fun_val = 0.009324
iter_number =   3 norm_grad = 0.027465 fun_val = 0.008958
      :                  :                      :
iter_number = 201 norm_grad = 0.000010 fun_val = 0.000000
```

■

Clearly, 201 iterations is a large number of steps in order to obtain convergence for a two-dimensional problem. The main question that arises is whether we can find a quantity that can predict in some sense the number of iterations required by the gradient method for convergence; this measure should quantify in some sense the "hardness" of the problem. This is an important issue that actually does not have a full answer, but a partial answer can be found in the notion of *condition number*.

4.3 ▪ The Condition Number

Consider the quadratic minimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^n}\{f(\mathbf{x}) \equiv \mathbf{x}^T\mathbf{A}\mathbf{x}\}, \tag{4.6}$$

where $\mathbf{A} \succ \mathbf{0}$. The optimal solution is obviously $\mathbf{x}^* = \mathbf{0}$. The gradient method with exact line search takes the form

$$\mathbf{x}_{k+1} = \mathbf{x}_k - t_k\mathbf{d}_k,$$

where $\mathbf{d}_k = 2\mathbf{A}\mathbf{x}_k$ is the gradient of f at $\mathbf{x}_k$ and the stepsize t_k chosen by the exact minimization rule is (see formula (4.3))

$$t_k = \frac{\mathbf{d}_k^T\mathbf{d}_k}{2\mathbf{d}_k^T\mathbf{A}\mathbf{d}_k}. \tag{4.7}$$

It holds that

$$\begin{aligned} f(\mathbf{x}_{k+1}) &= \mathbf{x}_{k+1}^T\mathbf{A}\mathbf{x}_{k+1} \\ &= (\mathbf{x}_k - t_k\mathbf{d}_k)^T\mathbf{A}(\mathbf{x}_k - t_k\mathbf{d}_k) \\ &= \mathbf{x}_k^T\mathbf{A}\mathbf{x}_k - 2t_k\mathbf{d}_k^T\mathbf{A}\mathbf{x}_k + t_k^2\mathbf{d}_k^T\mathbf{A}\mathbf{d}_k \\ &= \mathbf{x}_k^T\mathbf{A}\mathbf{x}_k - t_k\mathbf{d}_k^T\mathbf{d}_k + t_k^2\mathbf{d}_k^T\mathbf{A}\mathbf{d}_k. \end{aligned}$$

Plugging in the expression for t_k given in (4.7) into the last equation we obtain that

$$f(\mathbf{x}_{k+1}) = \mathbf{x}_k^T\mathbf{A}\mathbf{x}_k - \frac{1}{4}\frac{(\mathbf{d}_k^T\mathbf{d}_k)^2}{\mathbf{d}_k^T\mathbf{A}\mathbf{d}_k} = \mathbf{x}_k^T\mathbf{A}\mathbf{x}_k\left(1-\frac{1}{4}\frac{(\mathbf{d}_k^T\mathbf{d}_k)^2}{(\mathbf{d}_k^T\mathbf{A}\mathbf{d}_k)(\mathbf{x}_k^T\mathbf{A}\mathbf{A}^{-1}\mathbf{A}\mathbf{x}_k)}\right)$$
$$= \left(1-\frac{(\mathbf{d}_k^T\mathbf{d}_k)^2}{(\mathbf{d}_k^T\mathbf{A}\mathbf{d}_k)(\mathbf{d}_k^T\mathbf{A}^{-1}\mathbf{d}_k)}\right)f(\mathbf{x}_k). \tag{4.8}$$

We will now use the following well-known result, also known as the *Kantorovich inequality*.

Lemma 4.10 (Kantorovich inequality). *Let* $\mathbf{A}$ *be a positive definite* $n \times n$ *matrix. Then for any* $0 \neq \mathbf{x} \in \mathbb{R}^n$ *the inequality*

$$\frac{(\mathbf{x}^T\mathbf{x})^2}{(\mathbf{x}^T\mathbf{A}\mathbf{x})(\mathbf{x}^T\mathbf{A}^{-1}\mathbf{x})} \geq \frac{4\lambda_{\max}(\mathbf{A})\lambda_{\min}(\mathbf{A})}{(\lambda_{\max}(\mathbf{A})+\lambda_{\min}(\mathbf{A}))^2} \tag{4.9}$$

holds.

Proof. Denote $m = \lambda_{\min}(\mathbf{A})$ and $M = \lambda_{\max}(\mathbf{A})$. The eigenvalues of the matrix $\mathbf{A}+Mm\mathbf{A}^{-1}$ are $\lambda_i(\mathbf{A})+\frac{Mm}{\lambda_i(\mathbf{A})}, i = 1,\ldots,n$. It is easy to show that the maximum of the one-dimensional function $\varphi(t) = t + \frac{Mm}{t}$ over $[m,M]$ is attained at the endpoints m and M with a corresponding value of $M+m$, and therefore, since $m \leq \lambda_i(\mathbf{A}) \leq M$, it follows that the eigenvalues of $\mathbf{A}+Mm\mathbf{A}^{-1}$ are smaller than $(M+m)$. Thus,

$$\mathbf{A}+Mm\mathbf{A}^{-1} \preceq (M+m)\mathbf{I}.$$

Consequently, for any $\mathbf{x} \in \mathbb{R}^n$,

$$\mathbf{x}^T\mathbf{A}\mathbf{x}+Mm(\mathbf{x}^T\mathbf{A}^{-1}\mathbf{x}) \leq (M+m)(\mathbf{x}^T\mathbf{x}),$$

which combined with the simple inequality $\alpha\beta \leq \frac{1}{4}(\alpha+\beta)^2$ (for any two real numbers α,β) yields

$$(\mathbf{x}^T\mathbf{A}\mathbf{x})[Mm(\mathbf{x}^T\mathbf{A}^{-1}\mathbf{x})] \leq \frac{1}{4}\left[(\mathbf{x}^T\mathbf{A}\mathbf{x})+Mm(\mathbf{x}^T\mathbf{A}^{-1}\mathbf{x})\right]^2 \leq \frac{(M+m)^2}{4}(\mathbf{x}^T\mathbf{x})^2,$$

which after some simple rearrangement of terms establishes the desired result. □

Coming back to the convergence rate analysis of the gradient method on problem (4.6), it follows by using the Kantorovich inequality that (4.8) yields

$$f(\mathbf{x}_{k+1}) \leq \left(1-\frac{4Mm}{(M+m)^2}\right)f(\mathbf{x}_k) = \left(\frac{M-m}{M+m}\right)^2 f(\mathbf{x}_k),$$

where $M = \lambda_{\max}(\mathbf{A}), m = \lambda_{\min}(\mathbf{A})$. We summarize the above discussion in the following lemma.

Lemma 4.11. *Let* $\{\mathbf{x}_k\}_{k\geq 0}$ *be the sequence generated by the gradient descent method with exact line search for solving problem* (4.6). *Then for any* $k = 0,1,\ldots$

$$f(\mathbf{x}_{k+1}) \leq \left(\frac{M-m}{M+m}\right)^2 f(\mathbf{x}_k), \tag{4.10}$$

where $M = \lambda_{\max}(\mathbf{A}), m = \lambda_{\min}(\mathbf{A})$.

Inequality (4.10) implies

$$f(\mathbf{x}_k) \leq c^k f(\mathbf{x}_0),$$

where $c = (\frac{M-m}{M+m})^2$. That is, the sequence of function values is bounded above by a decreasing geometric sequence. In this case we say that the sequence of function values converges at a *linear rate* to the optimal value. The speed of the convergence depends on c; as c gets larger, the convergence speed becomes slower. The quantity c can also be written as

$$c = \left(\frac{\varkappa - 1}{\varkappa + 1}\right)^2,$$

where $\varkappa = \frac{M}{m} = \frac{\lambda_{\max}(\mathbf{A})}{\lambda_{\min}(\mathbf{A})}$. Since c is an increasing function of $\varkappa$, it follows that the behavior of the gradient method depends on the ratio between the maximal and minimal eigenvalues of $\mathbf{A}$; this number is called the *condition number*. Although the condition number can be defined for general matrices, we will restrict ourselves to positive definite matrices.

Definition 4.12 (condition number). *Let* $\mathbf{A}$ *be an* $n \times n$ *positive definite matrix. Then the* **condition number** *of* $\mathbf{A}$ *is defined by*

$$\varkappa(\mathbf{A}) = \frac{\lambda_{\max}(\mathbf{A})}{\lambda_{\min}(\mathbf{A})}.$$

We have already found one illustration in Example 4.9 that the gradient method applied to problems with large condition number might require a large number of iterations and vice versa, the gradient method employed on problems with small condition number is likely to converge within a small number of steps. Indeed, the condition number of the matrix associated with the function $x^2 + 0.01y^2$ is $\frac{1}{0.01} = 100$, which is relatively large, is the cause for the 201 iterations that were required for convergence, and the small condition number of the matrix associated with the function $x^2 + 2y^2$ ($\varkappa = 2$) is the reason for the small number of required steps. Matrices with large condition number are called *ill-conditioned*, and matrices with small condition number are called *well-conditioned*. Of course, the entire discussion until now was on the restrictive class of quadratic objective functions, where the Hessian matrix is constant, but the notion of condition number also appears in the context of nonquadratic objective functions. In that case, it is well known that the rate of convergence of $\mathbf{x}_k$ to a given stationary point $\mathbf{x}^*$ depends on the condition number of $\varkappa(\nabla^2 f(\mathbf{x}^*))$. We will not focus on these theoretical results, but will illustrate it on a well-known ill-conditioned problem.

Example 4.13 (Rosenbrock function). The *Rosenbrock function* is the following function:

$$f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2.$$

The optimal solution is obviously $(x_1, x_2) = (1, 1)$ with corresponding optimal value 0. The Rosenbrock function is extremely ill-conditioned at the optimal solution. Indeed,

$$\nabla f(\mathbf{x}) = \begin{pmatrix} -400x_1(x_2 - x_1^2) - 2(1 - x_1) \\ 200(x_2 - x_1^2) \end{pmatrix},$$

$$\nabla^2 f(\mathbf{x}) = \begin{pmatrix} -400x_2 + 1200x_1^2 + 2 & -400x_1 \\ -400x_1 & 200 \end{pmatrix}.$$

It is not difficult to show that $(x_1, x_2) = (1, 1)$ is the unique stationary point. In addition,

$$\nabla^2 f(1, 1) = \begin{pmatrix} 802 & -400 \\ -400 & 200 \end{pmatrix}$$

and hence the condition number is

```
>> A=[802,-400;-400,200];
>> cond(A)
ans =
  2.5080e+003
```

A condition number of more than 2500 should have severe effects on the convergence speed of the gradient method. Let us then employ the gradient method with backtracking on the Rosenbrock function with starting vector $\mathbf{x}_0 = (2,5)^T$:

```
>> f=@(x)100*(x(2)-x(1)^2)^2+(1-x(1))^2;
>> g=@(x)[-400*(x(2)-x(1)^2)*x(1)-2*(1-x(1));200*(x(2)-x(1)^2)];
>> [x,fun_val]=gradient_method_backtracking(f,g,[2;5],2,0.25,0.5,1e-5);
iter_number =   1 norm_grad = 118.254478 fun_val = 3.221022
iter_number =   2 norm_grad = 0.723051 fun_val = 1.496586
     :                 :                    :
iter_number = 6889 norm_grad = 0.000019 fun_val = 0.000000
iter_number = 6890 norm_grad = 0.000009 fun_val = 0.000000
```

This run required the huge amount of 6890 iterations, so the ill-conditioning effect has a significant impact. To better understand the nature of this pessimistic run, consider the contour plots of the Rosenbrock function along with the iterates as illustrated in Figure 4.2. Note that the function has banana-shaped contour lines surrounding the unique stationary point $(1,1)$. ∎

Sensitivity of Solutions to Linear Systems

The condition number has an important role in the study of the sensitivity of linear systems of equations to perturbation in the right-hand-side vector. Specifically, suppose that

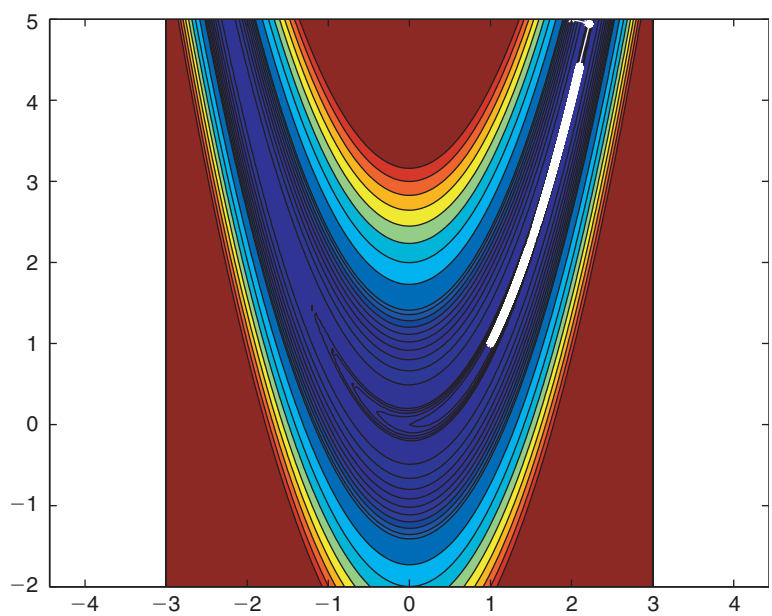

Figure 4.2. *Contour lines of the Rosenbrock function along with thousands of iterations of the gradient method.*

we are given a linear system $\mathbf{Ax} = \mathbf{b}$, and for the sake of simplicity, let us assume that $\mathbf{A}$ is positive definite. The solution of the linear system is of course $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$. Now, suppose that instead of $\mathbf{b}$ in the right-hand side, we consider a perturbation $\mathbf{b} + \Delta\mathbf{b}$. Let us denote the solution of the new system by $\mathbf{x} + \Delta\mathbf{x}$, that is, $\mathbf{A}(\mathbf{x} + \Delta\mathbf{x}) = \mathbf{b} + \Delta\mathbf{b}$. We have

$$\mathbf{x} + \Delta\mathbf{x} = \mathbf{A}^{-1}(\mathbf{b} + \Delta\mathbf{b}) = \mathbf{x} + \mathbf{A}^{-1}\Delta\mathbf{b},$$

so that the change in the solution is $\Delta\mathbf{x} = \mathbf{A}^{-1}\Delta\mathbf{b}$. Our purpose is to find a bound on the relative error $\frac{||\Delta\mathbf{x}||}{||\mathbf{x}||}$ in terms of the relative error of the right-hand-side perturbation $\frac{||\Delta\mathbf{b}||}{||\mathbf{b}||}$:

$$\frac{||\Delta\mathbf{x}||}{||\mathbf{x}||} = \frac{||\mathbf{A}^{-1}\Delta\mathbf{b}||}{||\mathbf{x}||} \leq \frac{||\mathbf{A}^{-1}|| \cdot ||\Delta\mathbf{b}||}{||\mathbf{x}||} = \frac{\lambda_{\max}(\mathbf{A}^{-1})||\Delta\mathbf{b}||}{||\mathbf{x}||},$$

where the last equality follows from the fact that the spectral norm of a positive definite matrix $\mathbf{D}$ is $||\mathbf{D}|| = \lambda_{\max}(\mathbf{D})$. By the positive definiteness of $\mathbf{A}$, it follows that $\lambda_{\max}(\mathbf{A}^{-1}) = \frac{1}{\lambda_{\min}(\mathbf{A})}$, and we can therefore continue the chain of equalities and inequalities:

$$\begin{aligned}
\frac{||\Delta\mathbf{x}||}{||\mathbf{x}||} &\leq \frac{1}{\lambda_{\min}(\mathbf{A})}\frac{||\Delta\mathbf{b}||}{||\mathbf{x}||} = \frac{1}{\lambda_{\min}(\mathbf{A})}\frac{||\Delta\mathbf{b}||}{||\mathbf{A}^{-1}\mathbf{b}||} \\
&\leq \frac{1}{\lambda_{\min}(\mathbf{A})} \cdot \frac{||\Delta\mathbf{b}||}{\lambda_{\min}(\mathbf{A}^{-1})||\mathbf{b}||} \qquad (4.11) \\
&= \frac{\lambda_{\max}(\mathbf{A})}{\lambda_{\min}(\mathbf{A})}\frac{||\Delta\mathbf{b}||}{||\mathbf{b}||} \\
&= \varkappa(\mathbf{A})\frac{||\Delta\mathbf{b}||}{||\mathbf{b}||},
\end{aligned}$$

where inequality (4.11) follows from the fact that for a positive definite matrix $\mathbf{A}$, the inequality $||\mathbf{A}^{-1}\mathbf{b}|| \geq \lambda_{\min}(\mathbf{A}^{-1})||\mathbf{b}||$ holds. Indeed,

$$||\mathbf{A}^{-1}\mathbf{b}|| = \sqrt{\mathbf{b}^T\mathbf{A}^{-2}\mathbf{b}} \geq \sqrt{\lambda_{\min}(\mathbf{A}^{-2})||\mathbf{b}||^2} = \lambda_{\min}(\mathbf{A}^{-1})||\mathbf{b}||.$$

We can therefore deduce that the sensitivity of the solution of the linear system to right-hand-side perturbations depends on the condition number of the coefficients matrix.

Example 4.14. As an example, consider the matrix

$$\mathbf{A} = \begin{pmatrix} 1 + 10^{-5} & 1 \\ 1 & 1 + 10^{-5} \end{pmatrix},$$

whose condition number is larger than 200000:

```
>> format long
>> A=[1+1e-5,1;1,1+1e-5];
>> cond(A)
ans =
    2.000009999998795e+005
```

The solution of the system

$$\mathbf{Ax} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

is

```
>> A\[1;1]
ans =
   0.499997500018278
   0.499997500006722
```

That is, approximately $(0.5, 0.5)^T$. However, if we make a small change to the right-hand-side vector: $(1.1, 1)^T$ instead of $(1, 1)^T$ (relative error of $\frac{\|(0.1,0)^T\|}{\|(1,1)^T\|} = 0.0707$), then the perturbed solution is very much different from $(0.5, 0.5)^T$:

```
>> A\[1.1;1]
ans =
  1.0e+003 *
   5.000524997400047
  -4.999475002650021
```

■

4.4 ▪ Diagonal Scaling

The problem of ill-conditioned problems is a major one, and many methods have been developed in order to circumvent it. One of the most popular approaches is to "condition" the problem by making an appropriate linear transformation of the decision variables. More precisely, consider the unconstrained minimization problem

$$\min\{f(\mathbf{x}) : \mathbf{x} \in \mathbb{R}^n\}.$$

For a given nonsingular matrix $\mathbf{S} \in \mathbb{R}^{n\times n}$, we make the linear transformation $\mathbf{x} = \mathbf{S}\mathbf{y}$ and obtain the equivalent problem

$$\min\{g(\mathbf{y}) \equiv f(\mathbf{S}\mathbf{y}) : \mathbf{y} \in \mathbb{R}^n\}.$$

Since $\nabla g(\mathbf{y}) = \mathbf{S}^T \nabla f(\mathbf{S}\mathbf{y}) = \mathbf{S}^T \nabla f(\mathbf{x})$, it follows that the gradient method applied to the transformed problem takes the form

$$\mathbf{y}_{k+1} = \mathbf{y}_k - t_k \mathbf{S}^T \nabla f(\mathbf{S}\mathbf{y}_k).$$

Multiplying the last equality by $\mathbf{S}$ from the left, and using the notation $\mathbf{x}_k = \mathbf{S}\mathbf{y}_k$, we obtain the recursive formula

$$\mathbf{x}_{k+1} = \mathbf{x}_k - t_k \mathbf{S}\mathbf{S}^T \nabla f(\mathbf{x}_k).$$

Defining $\mathbf{D} = \mathbf{S}\mathbf{S}^T$, we obtain the following version of the gradient method, which we call *the scaled gradient method* with scaling matrix $\mathbf{D}$:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - t_k \mathbf{D} \nabla f(\mathbf{x}_k).$$

By its definition, the matrix $\mathbf{D}$ is positive definite. The direction $-\mathbf{D}\nabla f(\mathbf{x}_k)$ is a descent direction of f at $\mathbf{x}_k$ when $\nabla f(\mathbf{x}_k) \neq 0$ since

$$f'(\mathbf{x}_k; -\mathbf{D}\nabla f(\mathbf{x}_k)) = -\nabla f(\mathbf{x}_k)^T \mathbf{D} \nabla f(\mathbf{x}_k) < 0,$$

where the latter strict inequality follows from the positive definiteness of the matrix $\mathbf{D}$. To summarize the above discussion, we have shown that the scaled gradient method with

scaling matrix $\mathbf{D}$ is equivalent to the gradient method employed on the function $g(\mathbf{y}) = f(\mathbf{D}^{1/2}\mathbf{y})$. We note that the gradient and Hessian of g are given by

$$\begin{aligned}\nabla g(\mathbf{y}) &= \mathbf{D}^{1/2}\nabla f(\mathbf{D}^{1/2}\mathbf{y}) = \mathbf{D}^{1/2}\nabla f(\mathbf{x}),\\ \nabla^2 g(\mathbf{y}) &= \mathbf{D}^{1/2}\nabla^2 f(\mathbf{D}^{1/2}\mathbf{y})\mathbf{D}^{1/2} = \mathbf{D}^{1/2}\nabla^2 f(\mathbf{x})\mathbf{D}^{1/2},\end{aligned}$$

where $\mathbf{x} = \mathbf{D}^{1/2}\mathbf{y}$.

The stepsize in the scaled gradient method can be chosen by each of the three options described in Section 4.1. It is often beneficial to choose the scaling matrix $\mathbf{D}$ differently at each iteration, and we describe this version explicitly below.

Scaled Gradient Method

Input: ε - tolerance parameter.

Initialization: Pick $\mathbf{x}_0 \in \mathbb{R}^n$ arbitrarily.
General step: For any $k = 0, 1, 2, \ldots$ execute the following steps:

(a) Pick a scaling matrix $\mathbf{D}_k \succ 0$.

(b) Pick a stepsize t_k by a line search procedure on the function

$$g(t) = f(\mathbf{x}_k - t\mathbf{D}_k\nabla f(\mathbf{x}_k)).$$

(c) Set $\mathbf{x}_{k+1} = \mathbf{x}_k - t_k\mathbf{D}_k\nabla f(\mathbf{x}_k)$.

(d) If $\|\nabla f(\mathbf{x}_{k+1})\| \le \varepsilon$, then STOP, and $\mathbf{x}_{k+1}$ is the output.

Note that the stopping criteria was chosen to be $\|\nabla f(\mathbf{x}_{k+1})\| \le \varepsilon$. A different and perhaps more appropriate stopping criteria might involve bounding the norm of $\mathbf{D}_k^{1/2}\nabla f(\mathbf{x}_{k+1})$.

The main question that arises is of course how to choose the scaling matrix $\mathbf{D}_k$. To accelerate the rate of convergence of the generated sequence, which depends on the condition number of the scaled Hessian $\mathbf{D}_k^{1/2}\nabla^2 f(\mathbf{x}_k)\mathbf{D}_k^{1/2}$, the scaling matrix is often chosen to make this scaled Hessian to be as close as possible to the identity matrix. When $\nabla^2 f(\mathbf{x}_k) \succ 0$, we can actually choose $\mathbf{D}_k = (\nabla^2 f(\mathbf{x}_k))^{-1}$ and the scaled Hessian becomes the identity matrix. The resulting method

$$\mathbf{x}_{k+1} = \mathbf{x}_k - t_k\nabla^2 f(\mathbf{x}_k)^{-1}\nabla f(\mathbf{x}_k)$$

is the celebrated *Newton's method*, which will be the topic of Chapter 5. One difficulty associated with Newton's method is that it requires full knowledge of the Hessian and in addition, the term $\nabla^2 f(\mathbf{x}_k)^{-1}\nabla f(\mathbf{x}_k)$ suggests that a linear system of the form

$\nabla^2 f(\mathbf{x}_k)\mathbf{d} = \nabla f(\mathbf{x}_k)$ needs to be solved at each iteration, which might be costly from a computational point of view. This is why simpler scaling matrices are suggested in the literature. The simplest of all scaling matrices are diagonal matrices. Diagonal scaling is in fact a natural idea since the ill-conditioning of optimization problems often arises as a result of a large variety of magnitudes of the decision variables. For example, if one variable is given in kilometers and the other in millimeters, then the first variables is 6 orders of magnitude larger than the second. This is a problem that could have been solved at the initial stage of the formulation of the problem, but when it is not, it can be resolved via diagonal rescaling. A natural choice for diagonal elements is

$$D_{ii} = (\nabla^2 f(\mathbf{x}_k))_{ii}^{-1}.$$

With the above choice, the diagonal elements of $\mathbf{D}^{1/2}\nabla^2 f(\mathbf{x}_k)\mathbf{D}^{1/2}$ are all one. Of course, this choice can be made only when the diagonal of the Hessian is positive.

Example 4.15. Consider the problem

$$\min\{1000x_1^2 + 40x_1x_2 + x_2^2\}.$$

We begin by employing the gradient method with exact stepsize, initial point $(1,1000)^T$, and tolerance parameter $\varepsilon = 10^{-5}$ using the MATLAB function `gradient_method_quadratic` that uses an exact line search.

```
>> A=[1000,20;20,1];
>> gradient_method_quadratic(A,[0;0],[1;1000],1e-5)

iter_number =   1 norm_grad = 1199.023961 fun_val = 598776.964973
iter_number =   2 norm_grad = 24186.628410 fun_val = 344412.923902
iter_number =   3 norm_grad = 689.671401 fun_val = 198104.25098
    :                         :                    :
iter_number =  68 norm_grad = 0.000287 fun_val = 0.000000
iter_number =  69 norm_grad = 0.000008 fun_val = 0.000000
ans =
  1.0e-005 *
   -0.0136
    0.6812
```

The excessive number of iterations is not surprising since the condition number of the associated matrix is large:

```
>> cond(A)
ans =
  1.6680e+003
```

The scaled gradient method with diagonal scaling matrix

$$\mathbf{D} = \begin{pmatrix} \frac{1}{1000} & 0 \\ 0 & 1 \end{pmatrix}$$

should converge faster since the condition number of the scaled matrix $\mathbf{D}^{1/2}\mathbf{A}\mathbf{D}^{1/2}$ is substantially smaller:

```
>> D=diag(1./diag(A));
>> sqrtm(D)*A*sqrtm(D)
```

```
ans =
    1.0000    0.6325
    0.6325    1.0000
>> cond(ans)
ans =
    4.4415
```

To check the performance of the scaled gradient method, we will use a slight modification of the MATLAB function `gradient_method_quadratic`, which we call `gradient_scaled_quadratic`.

```
function [x,fun_val]=gradient_scaled_quadratic(A,b,D,x0,epsilon)
% INPUT
% ======================
% A ....... the positive definite matrix associated
%           with the objective function
% b ....... a column vector associated with the linear part
%           of the objective function
% D ....... scaling matrix
% x0 ...... starting point of the method
% epsilon . tolerance parameter
% OUTPUT
% =======================
% x ....... an optimal solution (up to a tolerance)...
            of min(x^T A x+2 b^T x)
% fun_val . the optimal function value up to a tolerance


x=x0;
iter=0;
grad=2*(A*x+b);
while (norm(grad)>epsilon)
    iter=iter+1;
    t=grad'*D*grad/(2*(grad'*D')*A*(D*grad));
    x=x-t*D*grad;
    grad=2*(A*x+b);
    fun_val=x'*A*x+2*b'*x;
    fprintf('iter_number = %3d norm_grad = %2.6f fun_val = %2.6f \n',
    iter,norm(grad),fun_val);
end
```

The running of this code requires only 19 iterations:

```
>> gradient_scaled_quadratic(A,[0;0],D,[1;1000],1e-5)
iter_number =   1 norm_grad = 10461.338850 fun_val = 102437.875289
iter_number =   2 norm_grad = 4137.812524 fun_val = 10080.228908
     :                    :                        :
iter_number =  18 norm_grad = 0.000036 fun_val = 0.000000
iter_number =  19 norm_grad = 0.000009 fun_val = 0.000000
ans =
  1.0e-006 *
   -0.0106
    0.3061
```

■

4.5 ▪ The Gauss–Newton Method

In Section 3.5 we considered the nonlinear least squares problem

$$\min_{\mathbf{x}\in\mathbb{R}^n}\left\{g(\mathbf{x})\equiv\sum_{i=1}^{m}(f_i(\mathbf{x})-c_i)^2\right\}. \tag{4.12}$$

We will assume here that $f_1,\ldots,f_m$ are continuously differentiable over $\mathbb{R}^n$ for all $i = 1,2,\ldots,m$ and that $c_1,\ldots,c_m\in\mathbb{R}$. The problem is sometimes also written in the terms of the vector-valued function

$$F(\mathbf{x})=\begin{pmatrix} f_1(\mathbf{x})-c_1\\ f_2(\mathbf{x})-c_2\\ \vdots\\ f_m(\mathbf{x})-c_m\end{pmatrix},$$

and then it takes the form

$$\min\|F(\mathbf{x})\|^2.$$

The general step of the Gauss–Newton method goes as follows: given the kth iterate $\mathbf{x}_k$, the next iterate is chosen to minimize the sum of squares of the linear approximations of f_i at $\mathbf{x}_k$, that is,

$$\mathbf{x}_{k+1}=\operatorname{argmin}_{\mathbf{x}\in\mathbb{R}^n}\left\{\sum_{i=1}^{m}\left[f_i(\mathbf{x}_k)+\nabla f_i(\mathbf{x}_k)^T(\mathbf{x}-\mathbf{x}_k)-c_i\right]^2\right\}. \tag{4.13}$$

The minimization problem above is essentially a linear least squares problem

$$\min_{\mathbf{x}\in\mathbb{R}^n}\|\mathbf{A}_k\mathbf{x}-\mathbf{b}_k\|^2,$$

where

$$\mathbf{A}_k=\begin{pmatrix}\nabla f_1(\mathbf{x}_k)^T\\ \nabla f_2(\mathbf{x}_k)^T\\ \vdots\\ \nabla f_m(\mathbf{x}_k)^T\end{pmatrix}=J(\mathbf{x}_k)$$

is the so-called *Jacobian* matrix and

$$\mathbf{b}_k=\begin{pmatrix}\nabla f_1(\mathbf{x}_k)^T\mathbf{x}_k-f_1(\mathbf{x}_k)+c_1\\ \nabla f_2(\mathbf{x}_k)^T\mathbf{x}_k-f_2(\mathbf{x}_k)+c_2\\ \vdots\\ \nabla f_m(\mathbf{x}_k)^T\mathbf{x}_k-f_m(\mathbf{x}_k)+c_m\end{pmatrix}=J(\mathbf{x}_k)\mathbf{x}_k-F(\mathbf{x}_k).$$

The underlying assumption is of course that $J(\mathbf{x}_k)$ is of a full column rank; otherwise the minimization in (4.13) will not produce a unique minimizer. In that case, we can also write an explicit expression for the Gauss–Newton iterates (see formula (3.1) in Chapter 3):

$$\mathbf{x}_{k+1}=(J(\mathbf{x}_k)^TJ(\mathbf{x}_k))^{-1}J(\mathbf{x}_k)^T\mathbf{b}_k.$$

Note that the method can also be written as

$$\begin{aligned}\mathbf{x}_{k+1}&=(J(\mathbf{x}_k)^TJ(\mathbf{x}_k))^{-1}J(\mathbf{x}_k)^T(J(\mathbf{x}_k)\mathbf{x}_k-F(\mathbf{x}_k))\\ &=\mathbf{x}_k-(J(\mathbf{x}_k)^TJ(\mathbf{x}_k))^{-1}J(\mathbf{x}_k)^TF(\mathbf{x}_k).\end{aligned}$$

The Gauss–Newton direction is therefore $\mathbf{d}_k = (J(\mathbf{x}_k)^T J(\mathbf{x}_k))^{-1} J(\mathbf{x}_k)^T F(\mathbf{x}_k)$. Noting that $\nabla g(\mathbf{x}) = 2J(\mathbf{x})^T F(\mathbf{x})$, we can conclude that

$$\mathbf{d}_k = \frac{1}{2}(J(\mathbf{x}_k)^T J(\mathbf{x}_k))^{-1} \nabla g(\mathbf{x}_k),$$

meaning that the Gauss–Newton method is essentially a scaled gradient method with the following positive definite scaling matrix

$$\mathbf{D}_k = \frac{1}{2}(J(\mathbf{x}_k)^T J(\mathbf{x}_k))^{-1}.$$

This fact also explains why the Gauss–Newton method is a descent direction method. The method described so far is also called *the pure Gauss–Newton method* since no stepsize is involved. To transform this method into a practical algorithm, a stepsize is introduced, leading to the *damped Gauss–Newton* method.

Damped Gauss–Newton Method

Input: $\varepsilon > 0$ - tolerance parameter.

Initialization: Pick $\mathbf{x}_0 \in \mathbb{R}^n$ arbitrarily.
General step: For any $k = 0, 1, 2, \ldots$ execute the following steps:

(a) Set $\mathbf{d}_k = (J(\mathbf{x}_k)^T J(\mathbf{x}_k))^{-1} J(\mathbf{x}_k)^T F(\mathbf{x}_k)$.

(b) Set t_k by a line search procedure on the function

$$h(t) = g(\mathbf{x}_k - t\mathbf{d}_k).$$

(c) Set $\mathbf{x}_{k+1} = \mathbf{x}_k - t_k \mathbf{d}_k$.

(c) If $\|\nabla g(\mathbf{x}_{k+1})\| \le \varepsilon$, then STOP, and $\mathbf{x}_{k+1}$ is the output.

4.6 ▪ The Fermat–Weber Problem

The gradient method is the basis for many other methods that might seem at first glance to be unrelated to it. One interesting example is the Fermat–Weber problem. In the 17th century Pierre de Fermat posed the following problem: "Given three distinct points in the plane, find the point having the minimal sum of distances to these three points." The Italian physicist Torricelli solved this problem and defined a construction by ruler and compass for finding it (the point is thus called "the Torricelli point" or "the Torricelli–Fermat point"). Later on, it was generalized by the German economist Weber to a problem in the space $\mathbb{R}^n$ and with an arbitrary number of points. The problem known today as "the Fermat–Weber problem" is the following: given m points in $\mathbb{R}^n$: $\mathbf{a}_1, \ldots, \mathbf{a}_m$—also called the "anchor points"—and m weights $\omega_1, \omega_2, \ldots, \omega_m > 0$, find a point $\mathbf{x} \in \mathbb{R}^n$ that minimizes the weighted distance of $\mathbf{x}$ to each of the points $\mathbf{a}_1, \ldots, \mathbf{a}_m$. Mathematically, this problem can be cast as the minimization problem

$$\min_{\mathbf{x} \in \mathbb{R}^n} \left\{ f(\mathbf{x}) \equiv \sum_{i=1}^{m} \omega_i \|\mathbf{x} - \mathbf{a}_i\| \right\}.$$

Note that the objective function is not differentiable at the anchor points $\mathbf{a}_1, \ldots, \mathbf{a}_m$. This problem is one of the fundamental localization problems, and it is an instance of a *facility location* problem. For example, $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_m$ can represent locations of cities, and $\mathbf{x}$ will be the location of a new airport or hospital (or any other facility that serves the cities); the weights might be proportional to the size of population at each of the cities. One popular approach for solving the problem was introduced by Weiszfeld in 1937. The starting point is the first order optimality condition:

$$\nabla f(\mathbf{x}) = 0.$$

Note that we implicitly assume here that $\mathbf{x}$ is not an anchor point. The latter equality can be explicitly written as

$$\sum_{i=1}^{m} \omega_i \frac{\mathbf{x} - \mathbf{a}_i}{\|\mathbf{x} - \mathbf{a}_i\|} = 0.$$

After some algebraic manipulation, the latter relation can be written as

$$\left(\sum_{i=1}^{m} \frac{\omega_i}{\|\mathbf{x} - \mathbf{a}_i\|} \right) \mathbf{x} = \sum_{i=1}^{m} \frac{\omega_i \mathbf{a}_i}{\|\mathbf{x} - \mathbf{a}_i\|},$$

which is the same as

$$\mathbf{x} = \frac{1}{\sum_{i=1}^{m} \frac{\omega_i}{\|\mathbf{x} - \mathbf{a}_i\|}} \sum_{i=1}^{m} \frac{\omega_i \mathbf{a}_i}{\|\mathbf{x} - \mathbf{a}_i\|}.$$

We can thus reformulate the optimality condition as $\mathbf{x} = T(\mathbf{x})$, where T is the operator

$$T(\mathbf{x}) \equiv \frac{1}{\sum_{i=1}^{m} \frac{\omega_i}{\|\mathbf{x} - \mathbf{a}_i\|}} \sum_{i=1}^{m} \frac{\omega_i \mathbf{a}_i}{\|\mathbf{x} - \mathbf{a}_i\|}.$$

Thus, the problem of finding a stationary point of f can be recast as the problem of finding a fixed point of T. Therefore, a natural approach for solving the problem is via a fixed point method, that is, by the iterations

$$\mathbf{x}_{k+1} = T(\mathbf{x}_k).$$

We can now write explicitly Weiszfeld's theorem for solving the Fermat–Weber problem.

Weiszfeld's Method

Initialization: Pick $\mathbf{x}_0 \in \mathbb{R}^n$ such that $\mathbf{x}_0 \neq \mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_m$.
General step: For any $k = 0, 1, 2, \ldots$ compute

$$\mathbf{x}_{k+1} = T(\mathbf{x}_k) = \frac{1}{\sum_{i=1}^{m} \frac{\omega_i}{\|\mathbf{x}_k - \mathbf{a}_i\|}} \sum_{i=1}^{m} \frac{\omega_i \mathbf{a}_i}{\|\mathbf{x}_k - \mathbf{a}_i\|}. \tag{4.14}$$

Note that the algorithm is defined only when the iterates $\mathbf{x}_k$ are all different from $\mathbf{a}_1, \ldots, \mathbf{a}_m$. Although the algorithm was initially presented as a fixed point method, the

surprising fact is that it is basically a gradient method. Indeed,

$$\begin{aligned}\mathbf{x}_{k+1} &= \frac{1}{\sum_{i=1}^m \frac{\omega_i}{\|\mathbf{x}_k-\mathbf{a}_i\|}} \sum_{i=1}^m \frac{\omega_i \mathbf{a}_i}{\|\mathbf{x}_k-\mathbf{a}_i\|} \\ &= \mathbf{x}_k - \frac{1}{\sum_{i=1}^m \frac{\omega_i}{\|\mathbf{x}_k-\mathbf{a}_i\|}} \sum_{i=1}^m \omega_i \frac{\mathbf{x}_k-\mathbf{a}_i}{\|\mathbf{x}_k-\mathbf{a}_i\|} \\ &= \mathbf{x}_k - \frac{1}{\sum_{i=1}^m \frac{\omega_i}{\|\mathbf{x}_k-\mathbf{a}_i\|}} \nabla f(\mathbf{x}_k).\end{aligned}$$

Therefore, Weiszfeld's method is essentially the gradient method with a special choice of stepsize:

$$t_k = \frac{1}{\sum_{i=1}^m \frac{\omega_i}{\|\mathbf{x}_k-\mathbf{a}_i\|}}.$$

We are left of course with several questions. Is the method well-defined? That is, can we guarantee that none of the iterates $\mathbf{x}_k$ is equal to any of the points $\mathbf{a}_1,\ldots,\mathbf{a}_m$? Is the sequence of objective function values decreases? Does the sequence $\{\mathbf{x}_k\}_{k\geq 0}$ converge to a global optimal solution? We will answer at least part of these questions in this section.

We would like to show that the generated sequence of function values is nonincreasing. For that, we define the auxiliary function

$$h(\mathbf{y},\mathbf{x}) \equiv \sum_{i=1}^m \omega_i \frac{\|\mathbf{y}-\mathbf{a}_i\|^2}{\|\mathbf{x}-\mathbf{a}_i\|}, \qquad \mathbf{y}\in\mathbb{R}^n, \quad \mathbf{x}\in\mathbb{R}^n\backslash\mathscr{A},$$

where $\mathscr{A} \equiv \{\mathbf{a}_1,\mathbf{a}_2,\ldots,\mathbf{a}_m\}$. The function $h(\cdot,\cdot)$ has several important properties. First of all, the operator T can be computed on a vector $\mathbf{x}\notin\mathscr{A}$ by minimizing the function $h(\mathbf{y},\mathbf{x})$ over all $\mathbf{y}\in\mathbb{R}^n$.

Lemma 4.16. *For any $\mathbf{x}\in\mathbb{R}^n\backslash\mathscr{A}$, one has*

$$T(\mathbf{x}) = \operatorname{argmin}_{\mathbf{y}}\{h(\mathbf{y},\mathbf{x}) : \mathbf{y}\in\mathbb{R}^n\}. \tag{4.15}$$

Proof. The function $h(\cdot,\mathbf{x})$ is a quadratic function whose associated matrix is positive definite. In fact, the associated matrix is $(\sum_{i=1}^m \frac{\omega_i}{\|\mathbf{x}-\mathbf{a}_i\|})\mathbf{I}$. Therefore, by Lemma 2.41, the unique global minimum of (4.15), which we denote by $\mathbf{y}^*$, is the unique stationary point of $h(\cdot,\mathbf{x})$, that is, the point for which the gradient vanishes:

$$\nabla_{\mathbf{y}} h(\mathbf{y}^*,\mathbf{x}) = \mathbf{0}.$$

Thus,

$$2\sum_{i=1}^m \omega_i \frac{\mathbf{y}^*-\mathbf{a}_i}{\|\mathbf{x}-\mathbf{a}_i\|} = \mathbf{0}.$$

Extracting $\mathbf{y}^*$ from the last equation yields $\mathbf{y}^* = T(\mathbf{x})$, and the result is established. □

Lemma 4.16 basically shows that the update formula of Weiszfeld's method can be written as

$$\mathbf{x}_{k+1} = \operatorname{argmin}\{h(\mathbf{x},\mathbf{x}_k) : \mathbf{x}\in\mathbb{R}^n\}.$$

We are now ready to prove other important properties of h, which will be crucial for showing that the sequence $\{f(\mathbf{x}_k)\}_{k\geq 0}$ is nonincreasing.

Lemma 4.17. *If* $\mathbf{x} \in \mathbb{R}^n \setminus \mathscr{A}$, *then*

(a) $h(\mathbf{x},\mathbf{x}) = f(\mathbf{x})$,

(b) $h(\mathbf{y},\mathbf{x}) \geq 2f(\mathbf{y}) - f(\mathbf{x})$ *for any* $\mathbf{y} \in \mathbb{R}^n$,

(c) $f(T(\mathbf{x})) \leq f(\mathbf{x})$ *and* $f(T(\mathbf{x})) = f(\mathbf{x})$ *if and only if* $\mathbf{x} = T(\mathbf{x})$,

(d) $\mathbf{x} = T(\mathbf{x})$ *if and only if* $\nabla f(\mathbf{x}) = \mathbf{0}$.

Proof. (a) $h(\mathbf{x},\mathbf{x}) = \sum_{i=1}^m \omega_i \frac{\|\mathbf{x}-\mathbf{a}_i\|^2}{\|\mathbf{x}-\mathbf{a}_i\|} = \sum_{i=1}^m \omega_i \|\mathbf{x}-\mathbf{a}_i\| = f(\mathbf{x})$.

(b) For any nonnegative number a and positive number b, the inequality

$$\frac{a^2}{b} \geq 2a - b$$

holds. Substituting $a = \|\mathbf{y}-\mathbf{a}_i\|$ and $b = \|\mathbf{x}-\mathbf{a}_i\|$, it follows that for any $i = 1,2,\ldots,m$

$$\frac{\|\mathbf{y}-\mathbf{a}_i\|^2}{\|\mathbf{x}-\mathbf{a}_i\|} \geq 2\|\mathbf{y}-\mathbf{a}_i\| - \|\mathbf{x}-\mathbf{a}_i\|.$$

Multiplying the above inequality by ω_i and summing over $i = 1,2,\ldots,m$, we obtain that

$$\sum_{i=1}^m \omega_i \frac{\|\mathbf{y}-\mathbf{a}_i\|^2}{\|\mathbf{x}-\mathbf{a}_i\|} \geq 2\sum_{i=1}^m \omega_i \|\mathbf{y}-\mathbf{a}_i\| - \sum_{i=1}^m \omega_i \|\mathbf{x}-\mathbf{a}_i\|.$$

Therefore,

$$h(\mathbf{y},\mathbf{x}) \geq 2f(\mathbf{y}) - f(\mathbf{x}). \tag{4.16}$$

(c) Since $T(\mathbf{x}) = \operatorname{argmin}_{\mathbf{y}\in\mathbb{R}^n} h(\mathbf{y},\mathbf{x})$, it follows that

$$h(T(\mathbf{x}),\mathbf{x}) \leq h(\mathbf{x},\mathbf{x}) = f(\mathbf{x}), \tag{4.17}$$

where the last equality follows from part (a). Part (b) of the lemma yields

$$h(T(\mathbf{x}),\mathbf{x}) \geq 2f(T(\mathbf{x})) - f(\mathbf{x}),$$

which combined with (4.17) implies

$$f(\mathbf{x}) \geq h(T(\mathbf{x}),\mathbf{x}) \geq 2f(T(\mathbf{x})) - f(\mathbf{x}), \tag{4.18}$$

establishing the fact that $f(T(\mathbf{x})) \leq f(\mathbf{x})$. To complete the proof we need to show that $f(T(\mathbf{x})) = f(\mathbf{x})$ if and only if $T(\mathbf{x}) = \mathbf{x}$. Of course, if $T(\mathbf{x}) = \mathbf{x}$, then $f(T(\mathbf{x})) = f(\mathbf{x})$. To show the reverse implication, let us assume that $f(T(\mathbf{x})) = f(\mathbf{x})$. By the chain of inequalities (4.18) it follows that $h(\mathbf{x},\mathbf{x}) = f(\mathbf{x}) = h(T(\mathbf{x}),\mathbf{x})$. Since the unique minimizer of $h(\cdot,\mathbf{x})$ is $T(\mathbf{x})$, it follows that $\mathbf{x} = T(\mathbf{x})$.

(d) The proof of (d) follows by simple algebraic manipulation. □

We are now ready to prove the descent property of the sequence $\{f(\mathbf{x}_k)\}_{k\geq 0}$ under the assumption that all the iterates are not anchor points.

Lemma 4.18. *Let $\{\mathbf{x}_k\}_{k\geq 0}$ be the sequence generated by Weiszfeld's method* (4.14)*, where we assume that $\mathbf{x}_k \notin \mathscr{A}$ for all $k \geq 0$. Then we have the following:*

(a) *The sequence $\{f(\mathbf{x}_k)\}$ is nonincreasing: for any $k \geq 0$ the inequality $f(\mathbf{x}_{k+1}) \leq f(\mathbf{x}_k)$ holds.*

(b) *For any k, $f(\mathbf{x}_k) = f(\mathbf{x}_{k+1})$ if and only if $\nabla f(\mathbf{x}_k) = 0$.*

Proof. (a) Since $\mathbf{x}_k \notin \mathscr{A}$ for all k, this result follows by substituting $\mathbf{x} = \mathbf{x}_k$ in part (c) of Lemma 4.17.

(b) By part (c) of Lemma 4.17 it follows that $f(\mathbf{x}_k) = f(\mathbf{x}_{k+1}) = f(T(\mathbf{x}_k))$ if and only if $\mathbf{x}_k = \mathbf{x}_{k+1} = T(\mathbf{x}_k)$. By part (d) of Lemma 4.17 the latter is equivalent to $\nabla f(\mathbf{x}_k) = 0$. □

We have thus shown that sequence $\{f(\mathbf{x}_k)\}_{k\geq 0}$ is strictly decreasing as long as we are not stuck at a stationary point. The underlying assumption that $\mathbf{x}_k \notin \mathscr{A}$ is problematic in the sense that it cannot be easily guaranteed. One approach to make sure that the sequence generated by the method does not contain anchor points is to choose the starting point $\mathbf{x}_0$ so that its value is strictly smaller than the values of the anchor points:

$$f(\mathbf{x}_0) < \min\{f(\mathbf{x}_0), f(\mathbf{x}_1), \ldots, f(\mathbf{a}_m)\}.$$

This assumption, combined with the monotonicity of the function values of the sequence generated by the method, implies that the iterates do not include anchor points. We will prove that under this assumption any convergent subsequence of $\{\mathbf{x}_k\}_{k\geq 0}$ converges to a stationary point.

Theorem 4.19. *Let $\{\mathbf{x}_k\}_{k\geq 0}$ be the sequence generated by Weiszfeld's method and assume that $f(\mathbf{x}_0) < \min\{f(\mathbf{a}_1), f(\mathbf{a}_2), \ldots, f(\mathbf{a}_m)\}$. Then all the limit points of $\{\mathbf{x}_k\}_{k\geq 0}$ are stationary points of f.*

Proof. Let $\{\mathbf{x}_{k_n}\}_{n\geq 0}$ be a subsequence of $\{\mathbf{x}_k\}_{k\geq 0}$ that converges to a point $\mathbf{x}^*$. By the monotonicity of the method and the continuity of the objective function we have

$$f(\mathbf{x}^*) \leq f(\mathbf{x}_0) < \min\{f(\mathbf{a}_1), f(\mathbf{a}_2), \ldots, f(\mathbf{a}_m)\}.$$

Therefore, $\mathbf{x}^* \notin \mathscr{A}$, and hence $\nabla f(\mathbf{x}^*)$ is defined. We will show that $\nabla f(\mathbf{x}^*) = 0$. By the continuity of the operator T at $\mathbf{x}^*$, it follows that the sequence $\mathbf{x}_{k_n+1} = T(\mathbf{x}_{k_n}) \to T(\mathbf{x}^*)$ as $n \to \infty$. The sequence of function values $\{f(\mathbf{x}_k)\}_{k\geq 0}$ is nonincreasing and bounded below by 0 and thus converges to a value which we denote by f^*. Obviously, both $\{f(\mathbf{x}_{k_n})\}_{n\geq 0}$ and $\{f(\mathbf{x}_{k_n+1})\}_{n\geq 0}$ converge to f^*. By the continuity of f, we thus obtain that $f(T(\mathbf{x}^*)) = f(\mathbf{x}^*) = f^*$, which by parts (c) and (d) of Lemma 4.17 implies that $\nabla f(\mathbf{x}^*) = 0$. □

It can be shown that for the Fermat–Weber problem, stationary points are global optimal solutions, and hence the latter theorem shows that all limit points of the sequence generated by the method are global optimal solutions. In fact, it is also possible to show that the entire sequence converges to a global optimal solution, but this analysis is beyond the scope of the book.

4.7 ▪ Convergence Analysis of the Gradient Method

4.7.1 ▪ Lipschitz Property of the Gradient

In this section we will present a convergence analysis of the gradient method employed on the unconstrained minimization problem

$$\min\{f(\mathbf{x}) : \mathbf{x} \in \mathbb{R}^n\}.$$

We will assume that the objective function f is continuously differentiable and that its gradient ∇f is Lipschitz continuous over $\mathbb{R}^n$, meaning that there exists $L > 0$ for which

$$||\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})|| \leq L||\mathbf{x} - \mathbf{y}|| \text{ for any } \mathbf{x}, \mathbf{y} \in \mathbb{R}^n.$$

Note that if ∇f is Lipschitz with constant L, then it is also Lipschitz with constant $\tilde{L}$ for all $\tilde{L} \geq L$. Therefore, there are essentially infinite number of Lipschitz constants for a function with Lipschitz gradient. Frequently, we are interested in the smallest possible Lipschitz constant. The class of functions with Lipschitz gradient with constant L is denoted by $C_L^{1,1}(\mathbb{R}^n)$ or just $C_L^{1,1}$. Occasionally, when the exact value of the Lipschitz constant is unimportant, we will omit it and denote the class by $C^{1,1}$. For a given $D \subseteq \mathbb{R}^n$, the set of all continuously differentiable functions over D whose gradient satisfies the above Lipschitz condition for any $\mathbf{x}, \mathbf{y} \in D$ is denoted by $C_L^{1,1}(D)$. The following are some simple examples of $C^{1,1}$ functions:

- **Linear functions** Given $\mathbf{a} \in \mathbb{R}^n$, the function $f(\mathbf{x}) = \mathbf{a}^T\mathbf{x}$ is in $C_0^{1,1}$.
- **Quadratic functions** Let $\mathbf{A}$ be an $n \times n$ symmetric matrix, $\mathbf{b} \in \mathbb{R}^n$, and $c \in \mathbb{R}$. Then the function $f(\mathbf{x}) = \mathbf{x}^T\mathbf{A}\mathbf{x} + 2\mathbf{b}^T\mathbf{x} + c$ is a $C^{1,1}$ function.

To compute a Lipschitz constant of the quadratic function $f(\mathbf{x}) = \mathbf{x}^T\mathbf{A}\mathbf{x} + 2\mathbf{b}^T\mathbf{x} + c$, we can use the definition of Lipschitz continuity:

$$||\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})|| = 2||(\mathbf{A}\mathbf{x} + \mathbf{b}) - (\mathbf{A}\mathbf{y} + \mathbf{b})|| = 2||\mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{y}|| = 2||\mathbf{A}(\mathbf{x} - \mathbf{y})|| \leq 2||\mathbf{A}|| \cdot ||\mathbf{x} - \mathbf{y}||.$$

We thus conclude that a Lipschitz constant of ∇f is $2||\mathbf{A}||$.

A useful fact, stated in the following result, is that for twice continuously differentiable functions, Lipschitz continuity of the gradient is equivalent to boundedness of the Hessian.

Theorem 4.20. *Let f be a twice continuously differentiable function over $\mathbb{R}^n$. Then the following two claims are equivalent:*

(a) $f \in C_L^{1,1}(\mathbb{R}^n)$.

(b) $||\nabla^2 f(\mathbf{x})|| \leq L$ *for any* $\mathbf{x} \in \mathbb{R}^n$.

Proof. (b) $\Rightarrow$ (a). Suppose that $||\nabla^2 f(\mathbf{x})|| \leq L$ for any $\mathbf{x} \in \mathbb{R}^n$. Then by the fundamental theorem of calculus we have for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$

$$\begin{aligned}\nabla f(\mathbf{y}) &= \nabla f(\mathbf{x}) + \int_0^1 \nabla^2 f(\mathbf{x} + t(\mathbf{y} - \mathbf{x}))(\mathbf{y} - \mathbf{x}) dt \\ &= \nabla f(\mathbf{x}) + \left(\int_0^1 \nabla^2 f(\mathbf{x} + t(\mathbf{y} - \mathbf{x})) dt\right) \cdot (\mathbf{y} - \mathbf{x}),\end{aligned}$$

Thus,

$$\begin{aligned}
\|\nabla f(\mathbf{y})-\nabla f(\mathbf{x})\| &= \left\|\left(\int_0^1 \nabla^2 f(\mathbf{x}+t(\mathbf{y}-\mathbf{x}))dt\right)\cdot(\mathbf{y}-\mathbf{x})\right\| \\
&\le \left\|\int_0^1 \nabla^2 f(\mathbf{x}+t(\mathbf{y}-\mathbf{x}))dt\right\| \|\mathbf{y}-\mathbf{x}\| \\
&\le \left(\int_0^1 \|\nabla^2 f(\mathbf{x}+t(\mathbf{y}-\mathbf{x}))\|dt\right)\|\mathbf{y}-\mathbf{x}\| \\
&\le L\|\mathbf{y}-\mathbf{x}\|,
\end{aligned}$$

establishing the desired result $f \in C_L^{1,1}$.

(a) $\Rightarrow$ (b). Suppose now that $f \in C_L^{1,1}$. Then by the fundamental theorem of calculus for any $\mathbf{d} \in \mathbb{R}^n$ and $\alpha > 0$ we have

$$\nabla f(\mathbf{x}+\alpha\mathbf{d})-\nabla f(\mathbf{x}) = \int_0^\alpha \nabla^2 f(\mathbf{x}+t\mathbf{d})\mathbf{d}dt.$$

Thus,

$$\left\|\left(\int_0^\alpha \nabla^2 f(\mathbf{x}+t\mathbf{d})dt\right)\mathbf{d}\right\| = \|\nabla f(\mathbf{x}+\alpha\mathbf{d})-\nabla f(\mathbf{x})\| \le \alpha L\|\mathbf{d}\|.$$

Dividing by α and taking the limit $\alpha \to 0^+$, we obtain

$$\left\|\nabla^2 f(\mathbf{x})\mathbf{d}\right\| \le L\|\mathbf{d}\|,$$

implying that $\|\nabla^2 f(\mathbf{x})\| \le L$. □

Example 4.21. Let $f : \mathbb{R} \to \mathbb{R}$ be given by $f(x) = \sqrt{1+x^2}$. Then

$$0 \le f''(x) = \frac{1}{(1+x^2)^{3/2}} \le 1$$

for any $x \in \mathbb{R}$, and thus $f \in C_1^{1,1}$. ■

4.7.2 ▪ The Descent Lemma

An important result for $C^{1,1}$ functions is that they can be bounded above by a quadratic function over the entire space. This result, known as "the descent lemma," is fundamental in convergence proofs of gradient-based methods.

Lemma 4.22 (descent lemma). *Let $D \subseteq \mathbb{R}^n$ and $f \in C_L^{1,1}(D)$ for some $L > 0$. Then for any $\mathbf{x},\mathbf{y} \in D$ satisfying $[\mathbf{x},\mathbf{y}] \subseteq D$ it holds that*

$$f(\mathbf{y}) \le f(\mathbf{x}) + \nabla f(\mathbf{x})^T(\mathbf{y}-\mathbf{x}) + \frac{L}{2}\|\mathbf{x}-\mathbf{y}\|^2.$$

Proof. By the fundamental theorem of calculus,

$$f(\mathbf{y})-f(\mathbf{x}) = \int_0^1 \langle \nabla f(\mathbf{x}+t(\mathbf{y}-\mathbf{x})), \mathbf{y}-\mathbf{x}\rangle dt.$$

Therefore,

$$f(\mathbf{y})-f(\mathbf{x})=\langle\nabla f(\mathbf{x}),\mathbf{y}-\mathbf{x}\rangle+\int_0^1\langle\nabla f(\mathbf{x}+t(\mathbf{y}-\mathbf{x}))-\nabla f(\mathbf{x}),\mathbf{y}-\mathbf{x}\rangle dt.$$

Thus,

$$\begin{aligned}|f(\mathbf{y})-f(\mathbf{x})-\langle\nabla f(\mathbf{x}),\mathbf{y}-\mathbf{x}\rangle| &= \left|\int_0^1\langle\nabla f(\mathbf{x}+t(\mathbf{y}-\mathbf{x}))-\nabla f(\mathbf{x}),\mathbf{y}-\mathbf{x}\rangle dt\right| \\ &\le \int_0^1|\langle\nabla f(\mathbf{x}+t(\mathbf{y}-\mathbf{x}))-\nabla f(\mathbf{x}),\mathbf{y}-\mathbf{x}\rangle|dt \\ &\le \int_0^1||\nabla f(\mathbf{x}+t(\mathbf{y}-\mathbf{x}))-\nabla f(\mathbf{x})||\cdot||\mathbf{y}-\mathbf{x}||dt \\ &\le \int_0^1 tL||\mathbf{y}-\mathbf{x}||^2dt \\ &= \frac{L}{2}||\mathbf{y}-\mathbf{x}||^2. \quad \square\end{aligned}$$

Note that the proof of the descent lemma actually shows both upper and lower bounds on the function:

$$f(\mathbf{x})+\nabla f(\mathbf{x})^T(\mathbf{y}-\mathbf{x})-\frac{L}{2}||\mathbf{x}-\mathbf{y}||^2\le f(\mathbf{y})\le f(\mathbf{x})+\nabla f(\mathbf{x})^T(\mathbf{y}-\mathbf{x})+\frac{L}{2}||\mathbf{x}-\mathbf{y}||^2.$$

4.7.3 ▪ Convergence of the Gradient Method

Equipped with the descent lemma, we are now ready to prove the convergence of the gradient method for $C^{1,1}$ functions. Of course, we cannot guarantee convergence to a global optimal solution, but we can show the convergence to stationary points in the sense that the gradient converges to zero. We begin by the following result showing a "sufficient decrease" of the gradient method at each iteration. More precisely, we show that at each iteration the decrease in the function value is at least a constant times the squared norm of the gradient.

Lemma 4.23 (sufficient decrease lemma). *Suppose that $f\in C_L^{1,1}(\mathbb{R}^n)$. Then for any $\mathbf{x}\in\mathbb{R}^n$ and $t>0$*

$$f(\mathbf{x})-f(\mathbf{x}-t\nabla f(\mathbf{x}))\ge t\left(1-\frac{Lt}{2}\right)||\nabla f(\mathbf{x})||^2. \tag{4.19}$$

Proof. By the descent lemma we have

$$\begin{aligned}f(\mathbf{x}-t\nabla f(\mathbf{x})) &\le f(\mathbf{x})-t||\nabla f(\mathbf{x})||^2+\frac{Lt^2}{2}||\nabla f(\mathbf{x})||^2 \\ &= f(\mathbf{x})-t\left(1-\frac{Lt}{2}\right)||\nabla f(\mathbf{x})||^2.\end{aligned}$$

The result then follows by simple rearrangement of terms. $\square$

Our goal now is to show that a sufficient decrease property occurs in each of the stepsize selection strategies: constant, exact line search, and backtracking. In the constant stepsize setting, we assume that $t_k = \bar{t} \in \left(0, \frac{2}{L}\right)$. Substituting $\mathbf{x} = \mathbf{x}_k, t = \bar{t}$ in (4.19) yields the inequality

$$f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \geq \bar{t}\left(1 - \frac{L\bar{t}}{2}\right)\|\nabla f(\mathbf{x}_k)\|^2. \tag{4.20}$$

Note that the guaranteed descent in the gradient method per iteration is

$$\bar{t}\left(1 - \frac{\bar{t}L}{2}\right)\|\nabla f(\mathbf{x}_k)\|^2.$$

If we wish to obtain the largest guaranteed bound on the decrease, then we seek the maximum of $\bar{t}(1 - \frac{\bar{t}L}{2})$ over $(0, \frac{2}{L})$. This maximum is attained at $\bar{t} = \frac{1}{L}$, and thus a popular choice for a stepsize is $\frac{1}{L}$. In this case we have

$$f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) = f(\mathbf{x}_k) - f\left(\mathbf{x}_k - \frac{1}{L}\nabla f(\mathbf{x}_k)\right) \geq \frac{1}{2L}\|\nabla f(\mathbf{x}_k)\|^2. \tag{4.21}$$

In the exact line search setting, the update formula of the algorithm is

$$\mathbf{x}_{k+1} = \mathbf{x}_k - t_k \nabla f(\mathbf{x}_k),$$

where $t_k \in \operatorname{argmin}_{t \geq 0} f(\mathbf{x}_k - t\nabla f(\mathbf{x}_k))$. By the definition of t_k we know that

$$f(\mathbf{x}_k - t_k \nabla f(\mathbf{x}_k)) \leq f\left(\mathbf{x}_k - \frac{1}{L}\nabla f(\mathbf{x}_k)\right),$$

and thus we have

$$f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \geq f(\mathbf{x}_k) - f\left(\mathbf{x}_k - \frac{1}{L}\nabla f(\mathbf{x}_k)\right) \geq \frac{1}{2L}\|\nabla f(\mathbf{x}_k)\|^2, \tag{4.22}$$

where the last inequality was shown in (4.21).

In the backtracking setting we seek a small enough stepsize t_k for which

$$f(\mathbf{x}_k) - f(\mathbf{x}_k - t_k \nabla f(\mathbf{x}_k)) \geq \alpha t_k \|\nabla f(\mathbf{x}_k)\|^2, \tag{4.23}$$

where $\alpha \in (0,1)$. We would like to find a lower bound on t_k. There are two options. Either $t_k = s$ (the initial value of the stepsize) or t_k is determined by the backtracking procedure, meaning that the stepsize $\tilde{t}_k = t_k/\beta$ is not acceptable and does not satisfy (4.23):

$$f(\mathbf{x}_k) - f\left(\mathbf{x}_k - \frac{t_k}{\beta}\nabla f(\mathbf{x}_k)\right) < \alpha\frac{t_k}{\beta}\|\nabla f(\mathbf{x}_k)\|^2, \tag{4.24}$$

Substituting $\mathbf{x} = \mathbf{x}_k, t = \frac{t_k}{\beta}$ in (4.19) we obtain that

$$f(\mathbf{x}_k) - f\left(\mathbf{x}_k - \frac{t_k}{\beta}\nabla f(\mathbf{x}_k)\right) \geq \frac{t_k}{\beta}\left(1 - \frac{Lt_k}{2\beta}\right)\|\nabla f(\mathbf{x}_k)\|^2,$$

which combined with (4.24) implies that

$$\frac{t_k}{\beta}\left(1 - \frac{Lt_k}{2\beta}\right) < \alpha\frac{t_k}{\beta},$$

which is the same as $t_k > \frac{2(1-\alpha)\beta}{L}$. Overall, we obtained that in the backtracking setting we have

$$t_k \geq \min\left\{s, \frac{2(1-\alpha)\beta}{L}\right\},$$

which combined with (4.23) implies that

$$f(\mathbf{x}_k) - f(\mathbf{x}_k - t_k \nabla f(\mathbf{x}_k)) \geq \alpha \min\left\{s, \frac{2(1-\alpha)\beta}{L}\right\} ||\nabla f(\mathbf{x}_k)||^2. \tag{4.25}$$

We summarize the above discussion, or, better said, the three inequalities (4.20), (4.22), and (4.25), in the following result.

Lemma 4.24 (sufficient decrease of the gradient method). *Let $f \in C_L^{1,1}(\mathbb{R}^n)$. Let $\{\mathbf{x}_k\}_{k\geq 0}$ be the sequence generated by the gradient method for solving*

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

with one of the following stepsize strategies:

- *constant stepsize $\bar{t} \in (0, \frac{2}{L})$,*
- *exact line search,*
- *backtracking procedure with parameters $s \in \mathbb{R}_{++}, \alpha \in (0,1)$, and $\beta \in (0,1)$.*

Then

$$f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \geq M||\nabla f(\mathbf{x}_k)||^2, \tag{4.26}$$

where

$$M = \begin{cases} \bar{t}\left(1 - \frac{\bar{t}L}{2}\right) & \textit{constant stepsize}, \\ \frac{1}{2L} & \textit{exact line search}, \\ \alpha \min\left\{s, \frac{2(1-\alpha)\beta}{L}\right\} & \textit{backtracking}. \end{cases}$$

We will now show the convergence of the norms of the gradients $||\nabla f(\mathbf{x}_k)||$ to zero.

Theorem 4.25 (convergence of the gradient method). *Let $f \in C_L^{1,1}(\mathbb{R}^n)$, and let $\{\mathbf{x}_k\}_{k\geq 0}$ be the sequence generated by the gradient method for solving*

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

with one of the following stepsize strategies:

- *constant stepsize $\bar{t} \in (0, \frac{2}{L})$,*
- *exact line search,*
- *backtracking procedure with parameters $s \in \mathbb{R}_{++}, \alpha \in (0,1)$, and $\beta \in (0,1)$.*

Assume that f is bounded below over $\mathbb{R}^n$, that is, there exists $m \in \mathbb{R}$ such that $f(\mathbf{x}) > m$ for all $\mathbf{x} \in \mathbb{R}^n$.

Then we have the following:

(a) *The sequence $\{f(\mathbf{x}_k)\}_{k\geq 0}$ is nonincreasing. In addition, for any $k \geq 0$, $f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k)$ unless $\nabla f(\mathbf{x}_k) = 0$.*

(b) *$\nabla f(\mathbf{x}_k) \to 0$ as $k \to \infty$.*

Proof. (a) By (4.26) we have that

$$f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \geq M\|\nabla f(\mathbf{x}_k)\|^2 \geq 0 \tag{4.27}$$

for some constant $M > 0$, and hence the equality $f(\mathbf{x}_k) = f(\mathbf{x}_{k+1})$ can hold only when $\nabla f(\mathbf{x}_k) = 0$.

(b) Since the sequence $\{f(\mathbf{x}_k)\}_{k\geq 0}$ is nonincreasing and bounded below, it converges. Thus, in particular $f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \to 0$ as $k \to \infty$, which combined with (4.27) implies that $\|\nabla f(\mathbf{x}_k)\| \to 0$ as $k \to \infty$. $\square$

We can also get an estimate for the rate of convergence of the gradient method, or more precisely of the norm of the gradients. This is done in the following result.

Theorem 4.26 (rate of convergence of gradient norms). *Under the setting of Theorem 4.25, let f^* be the limit of the convergent sequence $\{f(\mathbf{x}_k)\}_{k\geq 0}$. Then for any $n = 0, 1, 2, \ldots$*

$$\min_{k=0,1,\ldots,n} \|\nabla f(\mathbf{x}_k)\| \leq \sqrt{\frac{f(\mathbf{x}_0) - f^*}{M(n+1)}},$$

where

$$M = \begin{cases} \bar{t}\left(1 - \frac{\bar{t}L}{2}\right) & \textit{constant stepsize}, \\ \frac{1}{2L} & \textit{exact line search}, \\ \alpha \min\left\{s, \frac{2\beta(1-\alpha)}{L}\right\} & \textit{backtracking}. \end{cases}$$

Proof. Summing the inequality (4.26) over $k = 0, 1, \ldots, n$, we obtain

$$f(\mathbf{x}_0) - f(\mathbf{x}_{n+1}) \geq M \sum_{k=0}^{n} \|\nabla f(\mathbf{x}_k)\|^2.$$

Since $f(\mathbf{x}_{n+1}) \geq f^*$, we can thus conclude that

$$f(\mathbf{x}_0) - f^* \geq M \sum_{k=0}^{n} \|\nabla f(\mathbf{x}_k)\|^2.$$

Finally, using the latter inequality along with the fact that for every $k = 0,1,\ldots,n$ the obvious inequality $\|\nabla f(\mathbf{x}_k)\|^2 \geq \min_{k=0,1,\ldots,n} \|\nabla f(\mathbf{x}_k)\|^2$ holds, it follows that

$$f(\mathbf{x}_0) - f^* \geq M(n+1) \min_{k=0,1,\ldots,n} \|\nabla f(\mathbf{x}_k)\|^2,$$

implying the desired result. □

Exercises

4.1. Let $f \in C_L^{1,1}(\mathbb{R}^n)$ and let $\{\mathbf{x}_k\}_{k\geq 0}$ be the sequence generated by the gradient method with a constant stepsize $t_k = \frac{1}{L}$. Assume that $\mathbf{x}_k \to \mathbf{x}^*$. Show that if $\nabla f(\mathbf{x}_k) \neq 0$ for all $k \geq 0$, then $\mathbf{x}^*$ is *not* a local maximum point.

4.2. [9, Exercise 1.3.3] Consider the minimization problem

$$\min\{\mathbf{x}^T \mathbf{Q}\mathbf{x} : \mathbf{x} \in \mathbb{R}^2\},$$

where $\mathbf{Q}$ is a positive definite 2×2 matrix. Suppose we use the diagonal scaling matrix

$$\mathbf{D} = \begin{pmatrix} Q_{11}^{-1} & 0 \\ 0 & Q_{22}^{-1} \end{pmatrix}.$$

Show that the above scaling matrix improves the condition number of $\mathbf{Q}$ in the sense that

$$\varkappa(\mathbf{D}^{1/2}\mathbf{Q}\mathbf{D}^{1/2}) \leq \varkappa(\mathbf{Q}).$$

4.3. Consider the quadratic minimization problem

$$\min\{\mathbf{x}^T \mathbf{A}\mathbf{x} : \mathbf{x} \in \mathbb{R}^5\},$$

where $\mathbf{A}$ is the 5×5 Hilbert matrix defined by

$$A_{i,j} = \frac{1}{i+j-1}, \quad i,j = 1,2,3,4,5.$$

The matrix can be constructed via the MATLAB command `A = hilb(5)`. Run the following methods and compare the number of iterations required by each of the methods when the initial vector is $\mathbf{x}_0 = (1,2,3,4,5)^T$ to obtain a solution $\mathbf{x}$ with $\|\nabla f(\mathbf{x})\| \leq 10^{-4}$:

- gradient method with backtracking stepsize rule and parameters $\alpha = 0.5, \beta = 0.5, s = 1$;
- gradient method with backtracking stepsize rule and parameters $\alpha = 0.1, \beta = 0.5, s = 1$;
- gradient method with exact line search;
- diagonally scaled gradient method with diagonal elements $D_{ii} = \frac{1}{A_{ii}}, i = 1,2,3,4,5$ and exact line search;
- diagonally scaled gradient method with diagonal elements $D_{ii} = \frac{1}{A_{ii}}, i = 1,2,3,4,5$ and backtracking line search with parameters $\alpha = 0.1, \beta = 0.5, s = 1$.

4.4. Consider the Fermat–Weber problem

$$\min_{\mathbf{x}\in\mathbb{R}^n}\left\{f(\mathbf{x})\equiv\sum_{i=1}^{m}\omega_i\|\mathbf{x}-\mathbf{a}_i\|\right\},$$

where $\omega_1,\ldots,\omega_m>0$ and $\mathbf{a}_1,\ldots,\mathbf{a}_m\in\mathbb{R}^n$ are m different points. Let

$$p\in\operatorname{argmin}_{i=1,2,\ldots,m}f(\mathbf{a}_i).$$

Suppose that

$$\left\|\sum_{i\neq p}\omega_i\frac{\mathbf{a}_p-\mathbf{a}_i}{\|\mathbf{a}_p-\mathbf{a}_i\|}\right\|>\omega_p.$$

(i) Show that there exists a direction $\mathbf{d}\in\mathbb{R}^n$ such that $f'(\mathbf{a}_p;\mathbf{d})<0$.

(ii) Show that there exists $\mathbf{x}_0\in\mathbb{R}^n$ satisfying $f(\mathbf{x}_0)<\min\{f(\mathbf{a}_1),f(\mathbf{a}_2),\ldots,f(\mathbf{a}_p)\}$. Explain how to compute such a vector.

4.5. In the "source localization problem" we are given m locations of sensors $\mathbf{a}_1,\mathbf{a}_2,\ldots,\mathbf{a}_m\in\mathbb{R}^n$ and approximate distances between the sensors and an unknown "source" located at $\mathbf{x}\in\mathbb{R}^n$:

$$d_i\approx\|\mathbf{x}-\mathbf{a}_i\|.$$

The problem is to find and estimate $\mathbf{x}$ given the locations $\mathbf{a}_1,\mathbf{a}_2,\ldots,\mathbf{a}_m$ and the approximate distances $d_1,d_2,\ldots,d_m$. A natural formulation as an optimization problem is to consider the nonlinear least squares problem

$$\text{(SL)}\quad\min\left\{f(\mathbf{x})\equiv\sum_{i=1}^{m}(\|\mathbf{x}-\mathbf{a}_i\|-d_i)^2\right\}.$$

We will denote the set of sensors by $\mathscr{A}\equiv\{\mathbf{a}_1,\mathbf{a}_2,\ldots,\mathbf{a}_m\}$.

(i) Show that the optimality condition $\nabla f(\mathbf{x})=0$ $(\mathbf{x}\notin\mathscr{A})$ is the same as

$$\mathbf{x}=\frac{1}{m}\left\{\sum_{i=1}^{m}\mathbf{a}_i+\sum_{i=1}^{m}d_i\frac{\mathbf{x}-\mathbf{a}_i}{\|\mathbf{x}-\mathbf{a}_i\|}\right\}.$$

(ii) Show that the corresponding fixed point method

$$\mathbf{x}_{k+1}=\frac{1}{m}\left\{\sum_{i=1}^{m}\mathbf{a}_i+\sum_{i=1}^{m}d_i\frac{\mathbf{x}_k-\mathbf{a}_i}{\|\mathbf{x}_k-\mathbf{a}_i\|}\right\}$$

is a gradient method, assuming that $\mathbf{x}_k\notin\mathscr{A}$ for all $k\geq 0$. What is the stepsize?

4.6. Another formulation of the source localization problem consists of minimizing the following objective function:

$$\text{(SL2)}\quad\min_{\mathbf{x}\in\mathbb{R}^n}\left\{f(\mathbf{x})\equiv\sum_{i=1}^{m}(\|\mathbf{x}-\mathbf{a}_i\|^2-d_i^2)^2\right\}.$$

This is of course a nonlinear least squares problem, and thus the Gauss–Newton method can be employed in order to solve it. We will assume that $n=2$.

(i) Show that as long as all the points $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_m$ do not reside on the same line in the plane, the method is well-defined, meaning that the linear least squares problem solved at each iteration has a unique solution.

(ii) Write a MATLAB function that implements the damped Gauss–Newton method employed on problem (SL2) with a backtracking line search strategy with parameters $s = 1, \alpha = \beta = 0.5, \varepsilon = 10^{-4}$. Run the function on the two-dimensional problem ($n = 2$) with 5 anchors ($m = 5$) and data generated by the MATLAB commands

```
randn('seed',317);
A=randn(2,5);
x=randn(2,1);
d=sqrt(sum((A-x*ones(1,5)).^2))+0.05*randn(1,5);
d=d';
```

The columns of the 2×5 matrix $\mathbf{A}$ are the locations of the five sensors, $\mathbf{x}$ is the "true" location of the source, and $\mathbf{d}$ is the vector of noisy measurements between the source and the sensors. Compare your results (e.g., number of iterations) to the gradient method with backtracking and parameters $s = 1, \alpha = \beta = 0.5, \varepsilon = 10^{-4}$. Start both methods with the initial vector $(1000, -500)^T$.

4.7. Let $f(\mathbf{x}) = \mathbf{x}^T\mathbf{A}\mathbf{x} + 2\mathbf{b}^T\mathbf{x} + c$, where $\mathbf{A}$ is a symmetric $n \times n$ matrix, $\mathbf{b} \in \mathbb{R}^n$, and $c \in \mathbb{R}$. Show that the *smallest* Lipschitz constant of ∇f is $2||\mathbf{A}||$.

4.8. Let $f : \mathbb{R}^n \to \mathbb{R}$ be given by $f(\mathbf{x}) = \sqrt{1 + ||\mathbf{x}||^2}$. Show that $f \in C_1^{1,1}$.

4.9. Let $f \in C_L^{1,1}(\mathbb{R}^m)$, and let $\mathbf{A} \in \mathbb{R}^{m \times n}, \mathbf{b} \in \mathbb{R}^m$. Show that the function $g : \mathbb{R}^n \to \mathbb{R}$ defined by $g(\mathbf{x}) = f(\mathbf{A}\mathbf{x} + \mathbf{b})$ satisfies $g \in C_{\tilde{L}}^{1,1}(\mathbb{R}^n)$, where $\tilde{L} = ||\mathbf{A}||^2 L$.

4.10. Give an example of a function $f \in C_L^{1,1}(\mathbb{R})$ and a starting point $x_0 \in \mathbb{R}$ such that the problem $\min f(x)$ has an optimal solution and the gradient method with constant stepsize $t = \frac{2}{L}$ diverges.

4.11. Suppose that $f \in C_L^{1,1}(\mathbb{R}^n)$ and assume that $\nabla^2 f(\mathbf{x}) \succeq 0$ for any $\mathbf{x} \in \mathbb{R}^n$. Suppose that the optimal value of the problem $\min_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x})$ is f^*. Let $\{\mathbf{x}_k\}_{k \geq 0}$ be the sequence generated by the gradient method with constant stepsize $\frac{1}{L}$. Show that if $\{\mathbf{x}_k\}_{k \geq 0}$ is bounded, then $f(\mathbf{x}_k) \to f^*$ as $k \to \infty$.