where $\lambda_{\min}$ is the smallest eigenvalue of $\boldsymbol{H}_f(\boldsymbol{x}^*)$. That the sensitivity of the solution $\boldsymbol{x}^*$ depends on the Hessian matrix $\boldsymbol{H}_f(\boldsymbol{x}^*)$ is not surprising, since this matrix determines the shape of the contours (level sets) of f near $\boldsymbol{x}^*$. If $\boldsymbol{H}_f(\boldsymbol{x}^*)$ is ill-conditioned, then the contours of f will be relatively long and thin along some directions (namely eigenvectors of $\boldsymbol{H}_f(\boldsymbol{x}^*)$ corresponding to relatively small eigenvalues), and the solution $\boldsymbol{x}^*$ will be highly sensitive, and the value of f correspondingly insensitive, to perturbations in those directions.

Analyzing the sensitivity of a solution to a constrained optimization problem is significantly more complicated, and we will merely highlight the main issues. For an equality-constrained problem, $\min f(\boldsymbol{x})$ subject to $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{o}$, the error in the solution can be resolved into two components, one parallel to the constraint surface and the other orthogonal to the constraint surface. The sensitivity of the component parallel to the constraints depends on the conditioning of the projected Hessian matrix $\boldsymbol{Z}^T\boldsymbol{B}(\boldsymbol{x}^*, \boldsymbol{\lambda}^*)\boldsymbol{Z}$ (see Section 6.2.3), much as in the unconstrained case just considered. The sensitivity of the component orthogonal to the constraints depends on the magnitudes of the Lagrange multipliers, which in turn depend on the conditioning of the Jacobian matrix $\boldsymbol{J}_g^T(\boldsymbol{x}^*)$ of the constraint function $\boldsymbol{g}$. In particular, the larger a given Lagrange multiplier, the more influential the corresponding constraint is on the solution. If the Jacobian matrix $\boldsymbol{J}_g^T(\boldsymbol{x}^*)$ is nearly rank deficient, then the Lagrange multipliers will be highly sensitive and the resulting solution will likely be inaccurate.

6.4 Optimization in One Dimension

We begin with methods for optimization in one dimension, which is an important problem in its own right, and will also be a key subproblem in many algorithms for optimization in higher dimensions. First, we need a way of bracketing a minimum in an interval, analogous to the way we used a sign change for bracketing solutions to nonlinear equations in one dimension. A function $f: \mathbb{R} \to \mathbb{R}$ is *unimodal* on an interval $[a, b]$ if there is a unique value $x^* \in [a, b]$ such that $f(x^*)$ is the minimum of f on $[a, b]$, and for any $x_1, x_2 \in [a, b]$ with $x_1 < x_2$,

$$x_2 < x^* \text{ implies } f(x_1) > f(x_2) \quad \text{and} \quad x_1 > x^* \text{ implies } f(x_1) < f(x_2).$$

Thus, $f(x)$ is strictly decreasing for $x \le x^*$ and strictly increasing for $x \ge x^*$. The significance of this property is that it will enable us to refine an interval containing a solution by computing sample values of the function within the interval and discarding portions of the interval according to the function values obtained, analogous to bisection for solving nonlinear equations.

6.4.1 Golden Section Search

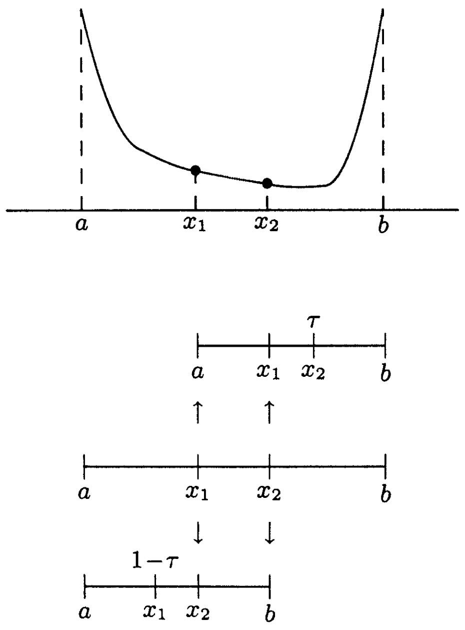
Suppose f is unimodal on $[a, b]$, and let $x_1, x_2 \in [a, b]$ with $x_1 < x_2$. Comparing the function values $f(x_1)$ and $f(x_2)$ and using the unimodality property allows us to discard a subinterval, either $(x_2, b]$ or $[a, x_1)$, and know that the minimum of the function lies within the remaining subinterval. In particular, if $f(x_1) < f(x_2)$, then the minimum cannot lie in the interval $(x_2, b]$, and if $f(x_1) > f(x_2)$, then the minimum cannot lie in the interval $[a, x_1)$. Thus, we are left with a shorter interval, either $[a, x_2]$ or $[x_1, b]$, within which we already have one function value, either $f(x_1)$ or $f(x_2)$, respectively. Hence, we will need to compute only one new function evaluation to repeat this process.

To make consistent progress in reducing the length of the interval containing the minimum, each new pair of points should have the same relative positions within the new interval that the previous pair had within the previous interval. Such an arrangement will enable us to reduce the length of the interval by a fixed fraction at each iteration, much as we reduced the length by half at each iteration of the bisection method for computing a zero of a function.

To accomplish this objective, we choose the relative positions of the two points within the current interval to be τ and $1 - \tau$, where $\tau^2 = 1 - \tau$, so that $\tau = (\sqrt{5}-1)/2 \approx 0.618$ and $1-\tau \approx 0.382$. With this choice, no matter which subinterval is retained, its length will be τ relative to the previous interval, and the interior point retained will be at position either τ or $1 - \tau$ relative to the new interval. Thus, we will need to compute only one new function value, at the complementary point, to continue the iteration. This choice of sample points is called *golden section search*, after the "golden ratio," $1 + \sqrt{5}/2 \approx 1.618$, of antiquity. The complete procedure is shown in Algorithm 6.1, for which the initial input is a function f, an interval $[a, b]$ on which f is unimodal, and an error tolerance *tol*. For a unimodal function, golden section search is safe but slowly convergent. Specifically, its convergence rate is linear, with $r = 1$ and $C \approx 0.618$.

Algorithm 6.1 Golden Section Search

```
$\tau = (\sqrt{5} - 1)/2$
$x_1 = a + (1 - \tau)(b - a)$
$f_1 = f(x_1)$
$x_2 = a + \tau(b - a)$
$f_2 = f(x_2)$
while $((b - a) > tol)$ do
    if $(f_1 > f_2)$ then
        $a = x_1$
        $x_1 = x_2$
        $f_1 = f_2$
        $x_2 = a + \tau(b - a)$
        $f_2 = f(x_2)$
    else
        $b = x_2$
        $x_2 = x_1$
        $f_2 = f_1$
        $x_1 = a + (1 - \tau)(b - a)$
        $f_1 = f(x_1)$
    end
end
```



Example 6.8 Golden Section Search. We illustrate golden section search by using it to minimize the function

$$f(x) = 0.5 - xe^{-x^2}.$$

Starting with the initial interval $[0, 2]$, we evaluate the function at points $x_1 = 0.764$ and $x_2 = 1.236$, obtaining $f(x_1) = 0.074$ and $f(x_2) = 0.232$. Because $f(x_1) < f(x_2)$, we know that the minimum must lie in the interval $[a, x_2]$, and thus we may replace b by x_2 and repeat the process. The first iteration is depicted in Fig. 6.6, and the full sequence of iterates is given next.

x_1	$f(x_1)$	x_2	$f(x_2)$
0.763932	0.073809	1.236068	0.231775
0.472136	0.122204	0.763932	0.073809
0.763932	0.073809	0.944272	0.112868
0.652476	0.073740	0.763932	0.073809
0.583592	0.084857	0.652476	0.073740
0.652476	0.073740	0.695048	0.071243
0.695048	0.071243	0.721360	0.071291
0.678787	0.071815	0.695048	0.071243
0.695048	0.071243	0.705098	0.071122
0.705098	0.071122	0.711310	0.071133
0.701260	0.071147	0.705098	0.071122
0.705098	0.071122	0.707471	0.071118
0.707471	0.071118	0.708937	0.071121
0.706565	0.071118	0.707471	0.071118

Note that the function values are relatively insensitive near the minimum, as expected (see Section 6.3).

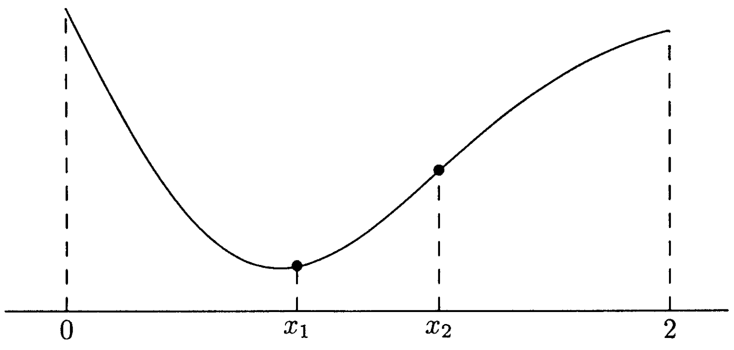


Figure 6.6: First iteration of golden section search for example problem.

Although unimodality plays a role in one-dimensional optimization similar to that played by a sign change in root finding, there are important practical differences. A sign change brackets a root of an equation regardless of how large the bracketing interval may be. The same is true of unimodality, but in practice most functions cannot be expected to be unimodal unless both endpoints of the interval are reasonably near a minimum, or unless the function has a special property such as convexity. Thus, more trial and error may be required to find a suitable starting interval for one-dimensional optimization than is typically required for root finding. In practice, one might simply search for three points such that the value of the objective function is greater at the two outer points than at the intermediate point. Although golden section search always converges, it is not guaranteed to find the global minimum, or even a local minimum, unless the objective function is unimodal on the starting interval.

6.4.2 Successive Parabolic Interpolation

As we have seen, golden section search for optimization is analogous in a number of ways to bisection for solving a nonlinear equation; in particular, golden section search makes no use of the function values other than to compare them. As with nonlinear equations, faster methods can be obtained by making greater use of the function values, such as fitting them with some simpler function. Fitting a straight line to two points, as in the secant method, is of no value for optimization because the resulting linear function has no minimum. Instead, we must use a polynomial of degree at least two.

The simplest example of this approach is *successive parabolic interpolation*. Initially, the function f to be minimized is evaluated at three points and a quadratic polynomial is fit to the three resulting function values. The minimum of the resulting parabola, assuming it has one, is then taken as a new approximate minimum of the function. One of the previous points is then dropped and the process repeated until convergence. At a given iteration, we have three points, say u, v, and w, with corresponding function values f_u, f_v, and f_w, respectively, where v is the best approximate minimum thus far. The minimum of the parabola interpolating the three function values is given by $v + p/q$, where

$$\begin{aligned} p &= \pm(v-u)^2(f_v - f_w) - (v-w)^2(f_v - f_u), \\ q &= \mp 2((v-u)(f_v - f_w) - (v-w)(f_v - f_u)). \end{aligned}$$

We now replace u by w, w by v, and v by the new approximate minimum $v+p/q$ and repeat until convergence. This process is illustrated in Fig. 6.7. Successive parabolic interpolation is not guaranteed to converge, but under normal circumstances if started close enough to a minimum it converges superlinearly with convergence rate $r \approx 1.324$.

Example 6.9 Successive Parabolic Interpolation. We illustrate successive parabolic interpolation by using it to minimize the function from Example 6.8,

$$f(x) = 0.5 - xe^{-x^2}.$$

We evaluate the function at three points, say, $x_0 = 0$, $x_1 = 1.2$, and $x_2 = 0.6$, obtaining $f(x_0) = 0.5$, $f(x_1) = 0.216$, $f(x_2) = 0.081$. We fit a parabola to these three points and take its minimizer, $x_3 = 0.754$, to be the next approximation to