Now suppose that $(x_k - x^*)/\|x_k - x^*\| \to \bar{d}$. It is obvious that

$$\|\nabla f(x_k)\|^2 = \|x_k - x^*\|^2(\|\nabla^2 f(x^*)\bar{d}\|^2 + o(1))$$

and

$$f(x_k) - f(x^*) = \frac{1}{2}\|x_k - x^*\|^2(\bar{d}^T\nabla^2 f(x^*)\bar{d} + o(1)).$$

Using the above equalities and (3.1.22) yields

$$\lim_{k\to\infty} \frac{\|\nabla f(x_k)\|^2}{f(x_k) - f(x^*)} = \frac{2\|\nabla^2 f(x^*)\bar{d}\|^2}{\bar{d}^T\nabla^2 f(x^*)\bar{d}} \geq 2m. \tag{3.1.25}$$

Hence, it follows from (3.1.24) and (3.1.25) that

$$\begin{aligned}\limsup_{k\to\infty} \beta_k &\leq 1 - \liminf_{k\to\infty} \frac{\|\nabla f(x_k)\|^2}{2M[f(x_k) - f(x^*)]} \\ &\leq 1 - \frac{m}{M} < 1.\end{aligned}$$

We complete the proof. □

3.1.3 Barzilai and Borwein Gradient Method

From the above discussions we know that the classical steepest descent method performs poorly, converges linearly, and is badly affected by ill-conditioning.

Barzilai and Borwein [8] presented a two-point step size gradient method, which is called usually the Barzilai-Borwein (or BB) gradient method. In the method, the step size is derived from a two-point approximation to the secant equation underlying quasi-Newton methods (see Chapter 5).

Consider the gradient iteration form

$$x_{k+1} = x_k - \alpha_k g_k \tag{3.1.26}$$

which can be written as

$$x_{k+1} = x_k - D_k g_k, \tag{3.1.27}$$

where $D_k = \alpha_k I$. In order to make the matrix D_k have quasi-Newton property, we compute α_k such that

$$\min \quad \|s_{k-1} - D_k y_{k-1}\|. \tag{3.1.28}$$

This yields that

$$\alpha_k = \frac{s_{k-1}^T y_{k-1}}{y_{k-1}^T y_{k-1}}, \tag{3.1.29}$$

where $s_{k-1} = x_k - x_{k-1}, y_{k-1} = g_k - g_{k-1}$.

By symmetry, we may minimize $\|D_k^{-1} s_{k-1} - y_{k-1}\|$ with respect to α_k and get

$$\alpha_k = \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}}. \tag{3.1.30}$$

The above description produces the following algorithm.

Algorithm 3.1.8 *(The Barzilai-Borwein gradient method)*

Step 0. *Given* $x_0 \in R^n, 0 < \varepsilon \ll 1$. *Set* $k = 0$.

Step 1. *If* $\|g_k\| \leq \varepsilon$, *stop ; otherwise let* $d_k = -g_k$.

Step 2. *If* $k = 0$, *find* α_0 *by line search; otherwise compute* α_k *by (3.1.29) or (3.1.30).*

Step 3. *Set* $x_{k+1} = x_k + \alpha_k d_k$.

Step 4. $k := k + 1$, *return to Step 1.* □

It is easy to see that in this method no matrix computations and no line searches (except $k = 0$) are required. The Barzilai-Borwein method is, in fact, a gradient method, but requires less computational work, and greatly speeds up the convergence of the gradient method. Barzilai and Borwein [8] proved that the above algorithm is R-superlinearly convergent for the quadratic case.

In the general non-quadratic case, a globalization strategy based on non-monotone line search is suitable to Barzilai-Borwein gradient method. In addition, in general non-quadratic case, α_k computed by (3.1.29) or (3.1.30) can be unacceptably large or small. Therefore, we must assume that α_k satisfies the condition

$$0 < \alpha^{(l)} \leq \alpha_k \leq \alpha^{(u)}, \quad \text{for all } k,$$

where $\alpha^{(l)}$ and $\alpha^{(u)}$ are previously determined numbers.

If we employ the iteration

$$x_{k+1} = x_k - \frac{1}{\alpha_k} g_k = x_k - \lambda_k g_k \tag{3.1.31}$$

with

$$\alpha_k = \frac{s_{k-1}^T y_{k-1}}{s_{k-1}^T s_{k-1}}, \quad \lambda_k = \frac{1}{\alpha_k}, \tag{3.1.32}$$

note that $s_k = -\frac{1}{\alpha_k} g_k = -\lambda_k g_k$, then we have

$$\alpha_{k+1} = \frac{s_k^T y_k}{s_k^T s_k} = \frac{-\lambda_k g_k^T y_k}{\lambda_k^2 g_k^T g_k} = -\frac{g_k^T y_k}{\lambda_k g_k^T g_k}.$$

Now we give the following Barzilai-Borwein gradient algorithm with nonmonotone globalization.

Algorithm 3.1.9 *(The Barzilai-Borwein gradient algorithm with nonmonotone linesearch)*

Step 0. *Given* $x_0 \in R^n, 0 < \varepsilon \ll 1$, *an integer* $M \geq 0$, $\rho \in (0,1), \delta > 0, 0 < \sigma_1 < \sigma_2 < 1$, $\alpha^{(l)}, \alpha^{(u)}$. *Set* $k = 0$.

Step 1. *If* $\|g_k\| \leq \varepsilon$, *stop.*

Step 2. *If* $\alpha_k \leq \alpha^{(l)}$ *or* $\alpha_k \geq \alpha^{(u)}$ *then set* $\alpha_k = \delta$.

Step 3. *Set* $\lambda = 1/\alpha_k$.

Step 4. *(nonmonotone line search) If*

$$f(x_k - \lambda g_k) \leq \max_{0 \leq j \leq \min(k,M)} f(x_{k-j}) - \rho \lambda g_k^T g_k,$$

then set

$$\lambda_k = \lambda, \quad x_{k+1} = x_k - \lambda_k g_k,$$

and go to Step 6.

Step 5. *Choose* $\sigma \in [\sigma_1, \sigma_2]$, *set* $\lambda = \sigma\lambda$, *and go to Step 4.*

Step 6. *Set* $\alpha_{k+1} = -(g_k^T y_k)/(\lambda_k g_k^T g_k)$, $k := k+1$, *return to Step 1.* □

Obviously, the above algorithm is globally convergent.