Chapter 1

Primal-Dual Methods

In this section we consider methods which compute the primal and dual variables (Lagrange multipliers) simultaneously. We start with the case of a quadratic objective function.

1.1 Quadratic programming

1.1.1 Introduction

The minimization of quadratic functions will be an important tool for the minimization of general nonlinear function f. This comes about as follows: the minimization of a general, C^2-function f proceeds by iteration. We start from some point x_0 and proceed through a series of iterates x_k hopefully convergent to the minimizer x_* of f.

Given iterate x_k we want to find the step $p_k := x_{k+1} - x_k$ that gets us to the next iterate x_{k+1}. To do this we approximate f by the quadratic Taylor polynomial centered at the current iterate x_k

$$\overline{f}(x_k + p) = f_k + g_k' p + \tfrac{1}{2} p' H_k p, \tag{1.1}$$

where $f_k := f(x_k)$, $g_k = \nabla f(x_k)$ is the gradient and $H_k = D^2 f(x_k)$ the Hessian of f at x_k which is assumed to be *positive definite*. This approximation is accurate only locally and hence we trust it only on a ball

$$B_r(0) = \{\, p \in \mathbb{R}^n \,:\, \|p\| \le r \,\}$$

of some sufficiently small radius r. This radius $r = r_k$ will be chosen using some heuristics in each step.

With this we want to minimze the quadratic function $h(p) = \overline{f}(x_k + p)$ on the ball $B_r(0)$ and let p_k be that minimizer. With this x_{k+1} becomes the minimum of the quadratic approximation $\overline{f}$ on the ball $B_r(x_k)$ and is guaranteed to satisfy $\overline{f}(x_{k+1}) \le \overline{f}(x_k) = f_k = f(x_k)$.

However x_{k+1} is not necessarily the locus of the minimum of f on the ball $B_r(x_k)$. To check whether we have made sufficient progress we must evaluate $f(x_{k+1}$ and check if we have a sufficient decrease from the value of $f(x_k)$.

From the study of iterative solutions based on line searches (Wolf-condition) we know that it is sufficient to decrease the value of f by a constant multiple of the optimal decrease *of the linear approximation*

$$\tilde{f}(x_k + p) = f_k + g_k' p, \tag{1.2}$$

as this will imply convergence to a minimizer possibly requiring a very large number of steps. This motivates the definition of the

Cauchy point

Note that $g_k = \nabla f(x_k)$ is also the gradient $\nabla \overline{f}(x_k)$ of the quadratic approximation $\overline{f}$ at the point x_k. The *Cauchy point* $x^C_{k+1}(r)$ is defined as the minimizer x *of the quadratic approximation* $\overline{f}$ on the line $x = x_k - tg_k$ of steepest descent from the point x_k subject to the condition $\|x - x_k\| \leq r$:

$$x^C_{k+1}(r) := x_k - t_* g_k, \quad \text{where} \tag{1.3}$$

$$t_* = argmin_{t \geq 0}\, \overline{f}(x_k - tg_k), \quad \text{subject to } \|tg_k\| \leq r. \tag{1.4}$$

Here we assume that $g_k = \nabla f(x_k) \neq 0$ since the minimization algorithm terminates as soon as a zero gradient is encountered. Clearly this is equivalent to

$$t_* = argmin_{t \in [0, r/\|g_k\|} g(t) \quad \text{where } g(t) = \overline{f}(x_k - tg_k).$$

Since we are minmizing the quadratic approximaton $\overline{f}$ and not the function f itself, t_* and hence the Cauchy point can be found explicitly: we have

$$g(t) = f_k - t\,\|g_k\|^2 + \tfrac{1}{2} t^2 g_k' H_k g_k$$

This polynomial has its global minimum at $t_0 = \|g_k\|^2 / g_k' H_k g_k$ and $g(t)$ is strictly decreasing from $t = 0$ to this point. Thus to observe the condition $t \leq r/\|g_k\|$ we can simply cut off t at this bound if the global solution t_0 is too large:

$$t_* = \min \left\{ \frac{r}{\|g_k\|}, \frac{\|g_k\|^2}{g_k' H_k g_k} \right\}. \tag{1.5}$$

With this the condition for the sufficient decrease in the value of the objective function f then has the form

$$f(x_k) - f(x_{k+1}) \geq \kappa(f(x_k) - f(x^C_{k+1}(r))). \tag{1.6}$$

In the development above the objective function f has been replaced with its quadratic approximation $\overline{f}$ (second order Taylor polynomial) and each step of the iterative algorithm only depends on this approximation (centered at the current iterate).

We can therefore abstract from this setting of iterative solution and study the quadratic problem in isolation.

1.2 Quadratic minimization

As a warmup we start with quadratic minimization under equality constraints as this will introduce some of the important notions. Thereafter we turn to quadratic approximation on a ball (the trust region).

1.2.1 Quadratic problem with equality constraints only

Consider first the case where there are no inequality constraints (and hence no complementary slackness conditions in the first order KKT equations). The problem reads

$$? = argmin_x\, f(x) = argmin_x\, c'x + \tfrac{1}{2}x'Gx \quad \text{subject to} \tag{1.7}$$

$$Ax = b, \tag{1.8}$$

where G is a symmetric $n \times n$-matrix, A an $m \times n$-matrix with $1 \leq m < n$ rows, $x \in \mathbb{R}^n$ and $b \in \mathbb{R}^m$.

We assume that A has full rank and hence $dim(ker(A)) = n - m$. Let Z be a kernel matrix for A, that is the columns of Z form a basis for the kernel of A. Then Z is an $n \times (n - m)$ matrix of full rank satisfying $AZ = 0$.

We say that G is positive semidefinite on the kernel of A if we have

$$v'Gv \geq 0, \quad \forall v \in ker(A), \tag{1.9}$$

equivalently if the matrix $Z'GZ$ is positive semidefinite. (Strict) positive definiteness of G on the kernel of A is defined analogously and is equivalent to the matrix $Z'GZ$ being positive definite.

Note that the kernel of A describes the degrees of freedom in the problem since the solution set of the equality constraint $Ax = b$ has the form $x_0 + ker(A)$, where x_0 is any particular solution of this constraint.
Denoting the Lagrange multipliers for the equality constraints with y, the Langrangian has the form

$$L(x, y) = c'x + \tfrac{1}{2}x'Gx + y'(Ax - b) = c'x + \tfrac{1}{2}x'Gx + \sum\nolimits_j y_j(a_i'x - b_i),$$

where $a_i = row_i(A)$, and setting all partial derivatives equal to zero yields the KKT conditions for the problem (1.7), (1.8) as

$$\begin{pmatrix} G & A' \\ A & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -c \\ b \end{pmatrix} \tag{1.10}$$

In general the existence of such a Lagrane multiplier y is only a necessary condition for x to be a local minium under the constraint but here it is sufficient for a global minimum

Proposition 1.2.1. *Suppose that G is positive semidefinite definite on the kernel of A and $x \in \mathbb{R}^n$. If there exists a Lagrange multiplier $y \in \mathbb{R}^m$ such that (1.10) is satisfied, then x is a solution of the problem (1.7), (1.8).*

If G is strictly positive semidefinite on $ker(A)$ then x is the unique solution of (1.7), (1.8).

Proof. Assume that (x, y) satisfy (1.10) and let $w \in \mathbb{R}^n$ with $Aw = b$ be arbitrary. We have to show that $x'Gx + 2c'x = 2f(x) \leq wf(w) = w'Gw + 2c'w$. Set $v = w - x$. Then $v \in ker(A)$. From (1.10) we have $v'(Gx + c) = -v'A'y = -y'Av = 0$. It follows that

$$\begin{aligned} 2f(w) &= (x+v)'G(x+v) + 2c'(x+v) = x'Gx + 2c'x + 2v'(Gx+c) + v'Gv \\ &= 2f(x) + v'Gv \geq 2f(x), \end{aligned}$$

since G is positive semidefinite on $ker(A)$. If G is strictly positive semidefinite on $ker(A)$, then $w \neq x$ implies $v \neq 0$ and hence $v'Gv > 0$, thus $f(w) > f(x)$ and the claim follows. ∎

Step to solution. Assume that we are starting at some point x and want to compute a *step* Δx such that $(x + \Delta x, y)$ satisfies the above equations. I.e. here x is considered known and Δx and y are the new unknowns.

Substituting $x + \Delta x$ into (1.10) and moving the known terms to the right we obtain the following system for the step Δx:

$$\begin{pmatrix} G & A' \\ A & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ y \end{pmatrix} = \begin{pmatrix} -(Gx + c) \\ b - Ax \end{pmatrix}. \tag{1.11}$$

In the present setting there is no need to formulate the problem in this way since we can just as well solve (1.10) directly. However we will use the quadratic function above as a local approximation to a nonlinear function and then an iterative approach is necessary. Naturally then we start at an iterate x and want to compute a step Δx to move to the next iterate.

The matrix on the left of (1.11) is symmetric but *never* even positive semidefinite. However if G is (strictly) positive definite on the kernel of A then this matrix is *nonsingular*. Indeed assume that

$$\begin{aligned} Gx + A'y = 0 \quad &\text{and} \\ Ax = 0. \end{aligned}$$

Then $x \in ker(A)$ and $0 = x'(Gx + A'y) = x'Gx + y'Ax = x'Gx$ which implies $x = 0$. It follows that $y = 0$ since A' has full rank. We can thus solve the system (1.11) directly with an LDL' decomposition or we can bring it to block diagonal form as follows: multiply the first equation from the left first by G^{-1}, then by A to obtain

$$\begin{aligned} A\Delta x + AG^{-1}A'y &= -(Ax + AG^{-1}c) \\ A\Delta x &= b - Ax \end{aligned}$$

which yields the equation

$$AG^{-1}A'y = -(b + AG^{-1}c). \tag{1.12}$$

Since A has full rank we have $ker(A') = 0$ and it follows that the matrix $AG^{-1}A'$ is also positive definite. Consequently (1.12) can be solved for y using

the Cholesky decomposition of $H = AG^{-1}A'$ which has size $m \times m$ where usually $m << n$.

The computation of $G^{-1}A'$ and $G^{-1}c$ can be handled as the solution of $GX = [A', c]$ using the Cholesky factorization of G. Assuming an optimized computation of the matrix product $AG^{-1}A'$ by blocking (LaPack, optmized BLAS) this will be faster than the direct approach in all cases.

Nullspace method. The preceeding approach needed the matrix G to be invertible (hence positive definite) but Lemma (1.2.1) only needed G to be positive definite on the null space of A, i.e. the matrix $Z'GZ$ is positive definite, where Z is any $n \times (n-m)$ matrix Z such that the columns of Z are a basis for the null space of A.

Suppose we have such a matrix Z. Then the solution set of the constraint $Ax = b$ has the form $x = x_0 + Zz$, $z \in \mathbb{R}^{n-m}$, where x_0 is any particular solution of this constraint. Rewriting $f(x)$ in terms of the new variable z we obtain

$$\begin{aligned} f(x) = \overline{f}(z) &= \tfrac{1}{2}(x_0 + Zz)'G(x_0 + Zz) + c'(x_0 + Zz) \\ &= \tfrac{1}{2}z'(Z'GZ)z + z'Z'Gx_0 + z'Z'c + x_0'Gx_0 + c'x_0 \\ &= \tfrac{1}{2}z'Hz + d'z + const, \quad \text{where } H = Z'GZ \text{ and } d = Z'(Gx_0 + c), \end{aligned}$$

so that the problem (1.7), (1.8) is equivalent to the uncontrained problem

$$? = argmin_z \tfrac{1}{2}z'Hz + d'z \quad \text{with } H = Z'GZ \text{ and } d = Z'(Gx_0 + c) \tag{1.13}$$

the solution of which is the solution of the equation

$$(Z'GZ)z = Hz = -d \tag{1.14}$$

which is an $(n-m) \times (n-m)$ system that can be solved with a Cholesky factorization of the matrix $H = Z'GZ$. This matrix product is fast if optimized by blocking. But the computation of Z is expensive: we can get it from a QR-decomposition of A':

$$A' = QR$$

with Q $n \times m$ orthogonal and R $m \times m$ upper triangular with no zeros on the diagonal (since A' has full rank m). With this it is clear that $ker(A)^{\perp} = Im(A') = Im(Q)$ is the span of the columns of Q.

Thus we can get an ON-basis for the kernel of A by enlarging the columns of Q to an ON-basis of $\mathbb{R}^n$ and Z then consists of the newly added columns. Some factorizations in LaPack yield such a decomposition $A' = QR$, where Q is $n \times n$ orthogonal and R $n \times m$ upper triangular with the last $n-m$ rows equal to zero ("full" decomposition).

From this we can read off Z directly as $Z = Q[, (m+1) : n]$ (the last $n-m$ columns of Q. However this QR factorization of A' is more expensive than a Cholesky decomposition.

1.2.2 Quadratic optimization on a ball

Now we consider the problem

$$? = argmin_p f(p) \quad \text{subject to } \|p\| \leq r, \quad \text{where } f(p) = g'p + \tfrac{1}{2}p'Hp \quad (1.15)$$

with H $n \times n$ symmetric and positive semidefinite. It is easy to see that an unconstrained global minimum exists if and only if $g \in ker(H)^{\perp}$. However a global minimum under the constraint always exists.
Minimization problems of this sort will be important for trust region methods.

An uncontrained global minimizer x would have to satisfy the equation $\nabla f(p) = 0$, i.e. $Hp = -g$ which has a solution only if $g \in Im(H) = Im(H') = ker(H)^{\perp}$. Suppose we regularize this equation by replacing the matrix H with $H + \lambda I$, where $\lambda > 0$. This matrix is positive definite so a solution x_* of

$$(H + \lambda I)p = -g \quad (1.16)$$

always exists and is the unique global minimizer of the function

$$f_\lambda(p) = p'(H + \lambda I)p + g'p = f(p) + \lambda \|p\|^2 .$$

We claim that p_* is an absolute minmizer of the original objective function f on the ball $B_r(0) = \{ p \in \mathbb{R}^n \ \|p\| \leq r \}$ with radius $r = \|p_*\|$. Indeed if $v \in B_r(0$ satisfies $f(v) < f(p_*)$ then we would have

$$f_\lambda(v) = f(v) + \lambda \|v\| \leq f(p_*) + \lambda r = f(p_*) + \lambda \|p_*\| = f_\lambda(p_*)$$

contradicting the fact that p_* is a global minimizer of f_λ. We record this fact as

Proposition 1.2.2. *The unique solution p_* of (1.16) is the global minimizer of $f(p) = g'p + \frac{1}{2}p'Hp$ on the ball $B_r(0)$ with radius $r = \|p_*\|$.* ∎

From this it would seem that the solution of (1.15) should be simple but this is not the case as the radius r in the proposition above *depends on the solution* p_*.

The main application will be to trust region methods as follows: the optimization of a general nonlinear function proceeds iteratively moving in small steps from one iterate to the next. At each step we use a quadratic approximation of the objective function centered at the current iterate x_k to compute the next step to be taken from this iterate.

We trust this approximation only on a ball $B_r(x_k)$ of small radius r centered at the current iterate x_k and therefore need to have some control over that radius. We then minimize the quadratic approximation on that ball $B_r(x_k)$ and move in the direction of the minimizer x_* in the next step. By simple translation $g(p) = f(x_k + p)$ this can be reformulated as a minimization over a ball centered at zero as above. With this the variable p is the step to be taken from the iterate x_k.

For this we need some control over the radius r (which will be increased or decreeased depending on the performance in the previous step: how much the value of the actual objective function was decreased).

This radius $r = r(\lambda) = \|p_*(\lambda)\|$ depends on λ and we need to study this dependence. We can do this using the eigenvector decomposition $H = UDU'$ with U orthogonal, $D \geq 0$ a diagonal matrix with the eigenvalues $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ of H on the diagonal and the columns $u_j = col_j(U)$ the corresponding eigenvectors: $Gu_j = \lambda_j u_j$.
Now expand g as a linear combination of these eigenvectors u_j:

$$g = \sum\nolimits_j a_j u_j,$$

where the coefficients a_j are the inner products $a_j = g'u_j$. It then follows that the u_j are also the eigenvectors of the matrix function $(H + \lambda I)^{-1}$ with associated eigenvalues $(\lambda_j + \lambda)^{-1}$ and this implies that

$$p_*(\lambda) = (H + \lambda I)^{-1} \sum\nolimits_j a_j u_j = \sum\nolimits_j \frac{a_j}{\lambda_j + \lambda} u_j$$

from which it follows that

$$r(\lambda)^2 = \|p_*(\lambda)\|^2 = \sum\nolimits_j \frac{(g'u_j)^2}{(\lambda_j + \lambda)^2} \tag{1.17}$$

With this we can then easily solve the equation $r(\lambda) = r$ for λ to obtain any desired radius r (e.g. using Newton's algorithm) but the computation of the eigenvalue decomposition of H is too expensive and has to be avoided. [1] has an algorithm that uses only a small number of Cholesky factorizations (algorithm 4.3, chapter 4, p87) but the derivation is unclear.
Recall from (1.5) in the introduction 1.1.1 (with $x = 0$) that the Cauchy point p_r^C of (1.15) is given by

$$p_r^C = -t_* g \quad \text{where } t_* = \min \left\{ \frac{r}{\|g\|}, \frac{\|g\|^2}{g'Hg} \right\}. \tag{1.18}$$

We now discuss several methods to improve on the Cauchy point:

The Dog Leg Method

The Cauchy point above is the global minimizer x of the quadratic function f along the line $x = -tg$, $t \geq 0$, in the steepest descent direction $-g = -\nabla f(0)$ subject to the condition $\|x\| \leq r$.

In the dog leg method we extend this line by adding a line segment from the Cauchy point to the global minmizer $p^H = -H^{-1}g$ of the quadratic function f, all subject to remaining inside the ball $B_r(0)$.

The *dog leg minimizer* p_r^D is the global minimizer of the quadratic function f on this extended path to the boundary of the ball $B_r(0)$. If the Cauchy point p_r^C

is already on the boundary of the ball $B_r(0)$ (i.e. if $t_* = r/\|g\|$), then $p_r^D = p_r^C$ (as no extension is possible).

Otherwise we minimize f also on the line $p = p_r^C + t(p_r^C - p^H)$ subject to the constraint $\|p\| \leq r$ and choose the new minimizer if it is better than the Cauchy point. In fact it is guaranteed to be better than the Cauchy point, as the function

$$h(t) = f(p_r^C + t(p_r^C - p^H))$$

is strictly decreasing for $t \in [0,1]$. Moreover the function $q(t) = \left\|p_r^C + t(p_r^C - p^H)\right\|$ is strictly increasing (see the next lemma). This implies that the minimum is assumed on the boundary of the ball $B_r(0)$, i.e. t is given by the equation

$$r^2 = \|a + t(b-a)\|^2 = t^2\|b-a\|^2 + 2ta'(b-a) + \|a\|^2, \quad \text{where} \quad a = p_r^C \quad \text{and} \quad b = p^H \tag{1.19}$$

Alltogether we have

$$p_r^D = \begin{cases} p_r^C & \text{if } \left\|p_r^C\right\| = r, \text{ i.e. if } t_* = -r/\|g\|, \\ p_r^C + t(p^H - p_r^C), & \text{with } t \geq 0 \text{ in (1.19) otherwise.} \end{cases} \tag{1.20}$$

This is justified by the following

Lemma 1.2.3. *Assume that H is positive definite. Then*
(a) The function $q(t) = \left\|p_r^C + t(p_r^C - p^H)\right\|$ is strictly increasing for $t \in [0,1]$.
(b) The function $h(t) = f(p_r^C + t(p_r^C - p^H))$ is strictly decreasing for $t \in [0,1]$.

Proof. Set $a = p_r^C = -t_* g$ and $b = p^H$ and note that $Hb = -g$. With this we have

$$q(t)^2 = \|a + t(b-a)\|^2 = t^2\|b-a\|^2 + 2ta'(b-a) + \|a\|^2 \tag{1.21}$$

$$= t^2\|b-a\|^2 + 2ta'(b-a) + \|a\|^2 \tag{1.22}$$

To see that this is increasing it will suffice to show that $a'(b-a) \geq 0$, that is

$$\|a\|^2 \leq a'b = b'a.$$

Now $\|a\|^2 = t_*^2\|g\|^2$ where $t_* \leq \|g\|^2/(g'Hg)$ and so

$$\|a\|^2 \leq t_* \frac{\|g\|^4}{g'Hg}$$

Moreover

$$b'a = t_* b'(-g) = t_* b'Hb.$$

so that it will suiffice to show that

$$\frac{\|g\|^4}{g'Hg} \leq b'Hb.$$

Recalling that $\|g\|^2 = g'g = -g'Hb$ we can rewrite this as

$$(g'Hb)^2 \le (g'Hg)(b'Hb)$$

which is the Cauchy-Schwartz inequality for the iner product $(g, b) := g'Hb$. This shows (a). To see (b) compute the derivative $h'(t)$ as

$$\begin{aligned} h'(t) &:= \frac{d}{dt} f(a + t(b-a) \\ &= \frac{d}{dt}\left[g'(a + t(b-a) + \tfrac{1}{2}(a + t(b-a))'H(a + t(b-a))\right] \\ &= g'(b-a) + (a + t(b-a))'H(b-a) \\ &= g'(b-a) + a'H(b-a) + t(b-a)'H(b-a) \end{aligned}$$

We need to show that $\rho(t) \le 0$, for $t \in [0, 1]$. Since $(b-a)'H(b-a) \ge 0$, $h'(t)$ is increasing on this interval so this is equivalent with $h'(1) \le 0$ which simplifies to

$$g'(b-a) + b'H(b-a) \le 0.$$

Now for vectors u, v the inner product $u'v$ is equal to $v'u$ and using this, the symmetry of H and $Hb = -g$ we have $b'H(b-a) = (b-a)'Hb = -g'(b-a)$ so that the above sum is actually equal to zero. ∎

We can now summarize the dog leg method as follows

Dog Leg Method. Find an approximate minmizer p of $f(p) = g'p + \frac{1}{2}p'Hp$ subject to $\|p\| \le r$ as follows:

- Solve $Hb = -g$ (the global minimizer). If $\|b\| \le r$ set $p = b = p^H$.
- If $\|b\| > r$ compute the Cauchy point p_r^C as in (1.18).
- If $\left\|p_r^C\right\| = r$, equivalently if $t_* = r/\|g\|$, set $p = p_r^C$.
- If $\left\|p_r^C\right\| < r$, let p be the dog leg point $p = p_r^D$ from (1.20).

If the Cauch point satisfies $\left\|p_r^C\right\| < r$ then the dog leg point p_r^D will reduce the value of the quadratic function f more than the Cauchy point.

If however this quadratic function is the second order Taylor polynomial approximation of a general nonlinear function g in a neighborhood $B_r(x_k)$ of an iterate x_k, then the dog leg point $x_k + p_r^D$ is *not guaranteed* to reduce the value of g more than the Cauchy point $x_k + p_r^C$ and will do so only if the quadratic approximation f of g on $B_r(x_k)$ is accurate enough.

Generally this can be guaranteed only by making the radius r small enough. Iterative methods for minimizing a general nonlinear function g then use heuristics to reduce or enlarge the *trust radius* r in each step according as the reduction in the value of g in the last step was a satisfactory multiple of the reduction of the quadratic approximation f of g.

If the global minimizer $b = -H^{-1}g$ is outside the ball $B_r(0)$ then the Cauchy- or dog leg point p are guaranteed to satisfy $\|p\| = r$. This is a consequence of

the Lemma above. This ensures that trust region iterative methods do not take steps that are too short and is a significant advantage over line search methods.

Regularization of the dog leg method.

Recall that the dog leg method is applied in the context of an iterative minmization of some C^2-function f. At the current iterate x_k we use the second order Taylor polynomial f

$$f(x_k + p) \simeq q(p) = f_k + g'p + p'Hp,$$

where $f_k = f(x_k)$, $g = \nabla f(x_k)$ is the gradient and $H = D^2 f(x_k)$ the Hessian of f at the iterate x_k.

The radius r defines the region $\|p\| \leq r$ on which we believe that the quadratic approximation $f(x_k + p) \simeq q(p)$ is sufficiently accurate so that a step p that decreases the value of q will also lead to a sufficient decrease in the value of f if we move from the iterate x_k to the next iterate $x_{k+1} = x_k + p$. The ball $B_r(x_k)$ is the called the *trust region* around x_k.

Thus we can compute the step p by minmizing the quadratic function q on the ball $\|p\| \leq r$ and we are in the situation described above where the dog leg method applies.

Suppose now that we fix some $\lambda \geq 0$ and replace the matrix H with $H + \lambda I$. This replaces the quadratic objective q with the new function

$$q_\lambda(p) = g'p + p'(H + \lambda I)p = q(p) + \lambda \|p\|^2 . \tag{1.23}$$

and leaves the Cauchy point unchanged or moves it closer to the iterate x_k along the line $x_k - tg$, see (1.18), the value of t_* may be decreased via the second term in the min.

Instead of p^H we now compute the point $p^H_\lambda := p^{H+\lambda I}$ as the solution of

$$(H + \lambda I)p = -g \tag{1.24}$$

and we know from proposition 1.2.2 that this point is the absolute minimizer of q on the ball $\|p\| \leq r(\lambda)$, where $r(\lambda) := \left\|p^H_\lambda\right\|$. From (1.17) we know that $r(\lambda)$ is a decreasing function of $\lambda \geq 0$.

To keep an iterative algorithm efficient we do not want to also solve the equation $Hp = -g$ for the point $p = p^H$. There are now several cases:

Case (I) $r(\lambda) << r$. This is the bad case as it would lead to a very small step p and has to be avoided by decreasing λ. This case will however be unavoidable if the global minimum of q is located close to the point $p = 0$, i.e. if the iterate x_k above is already close to a local minimizer of f.

Case (II) $r(\lambda) \leq r$ and $r(\lambda) \simeq r$. This is the ideal case as the point p^H_λ is nearly the global optimizer of q on the ball $\|p\| \leq r$.

Case (III) $r(\lambda) > r$. This is the interesting case. We have two subcases.

Case (IIIa) the Cauchy $p^C_r(\lambda)$ point is on the boundary $\|p\| = r$.
In that case the algorithm takes the step $p = p^C_r(\lambda)$ and a step of sufficient size assured.

Case (IIIb) the Cauchy point $p_r^C(\lambda)$ satisfies $\left\|p_r^C(\lambda)\right\| < r$.
Set $\delta := r - \left\|p_r^C(\lambda)\right\|$. We know from the proof of lemma 1.2.3 (applied to the matrix $H + \lambda I$ instead of H) that the function

$$q_\lambda(p) = q(p) + \lambda \|p\|^2$$

decreases on the line segment $p = p(t)$ from the Cauchy point $p_r^C(\lambda)$ to the dog leg point $p_r^D(\lambda)$ (computed with the matrix $H + \lambda I$). But since $\|p\|$ increases by the amount δ along the same line segment of the dog leg path, the quadratic approxiation q of f must decrease at least by the amount $\lambda\delta^2$ from the value at the Cauchy point along the same line segment.
In other words we have

$$q(p_r^D(\lambda)) \leq q(p_r^C(\lambda)) - \lambda\delta^2 \leq q(0) - \lambda\delta^2.$$

Now such a decrease must be small if the global minimizer of q is close to zero since then the value of q cannot be reduced significantly from the value $q(0)$.

This will occur if the iterate x_k is close to a local minimizer of f above and the radius r is small enough so that the quadratic approximation $f(x_k + p) \simeq q(p)$ is accurate. Thus, as we approach a local minimizer of f and shrink the trust radius r below the distance to the minimizer the Cauchy point will approach the boundary $\|p\| = r$.

Replacing H with $H+\lambda I$, where $\lambda > 0$, has the advantage that now H can be singular and thus only needs to be positive semidefinite. The discussion above shows that that will not destroy the iterative algorithm as long as we ensure that $r(\lambda)$ is not too small and reduce $\lambda \downarrow 0$ as we approach a local minimum, i.e. as the norm of the gradient g approaches zero.

Now if $H = UDU'$ is the eigenvalue decomposition of H with eigenvalues $0 \leq \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ and corresponding eigenvectors $u_j = row_j(U)$ satisfying $Hu_j = \lambda_j u_j$ we have seen above that

$$\|r(\lambda)\|^2 = \sum\nolimits_j \frac{a_j^2}{(\lambda + \lambda_j)^2}, \tag{1.25}$$

where the $a_j = g'u_j$ are the components of the gradient g in direction of the eigenvectors u_j of the Hessian H. In particular we have

$$\sum\nolimits_j a_j^2 = \|g\|^2 .$$

Unfortunately these components a_j are unknown as it is far too costly to compute the eigendecomposition of H. Recall that we want to choose λ such that

$$\|r(\lambda)\| = r \tag{1.26}$$

but we cannot precisely control the norm $r(\lambda)$ from (1.25) since the eigenvalues λ_j and components a_j are unknown. However we do know something about the eigenvalues of H: we have $\lambda_j \geq 0$ and the sum

$$\sum\nolimits_j \lambda_j = tr(H)$$

is the trace of H. Recall that the trace satisfies $tr(AB) = tr(BA)$ and so $tr(H) = tr(UDU') = tr(DU'U) = tr(D) = \sum_j \lambda_j$). With this we can substitue the average value

$$\mu := \frac{1}{n} \sum\nolimits_j \lambda_j = tr(H)/n$$

for all the λ_j into (1.25) to obtain the approximate equation

$$r = \|r(\lambda)\|^2 \simeq \sum\nolimits_j \frac{a_j^2}{(\lambda + \mu)^2} = \frac{1}{(\lambda + \mu)^2} \|g\|^2 . \tag{1.27}$$

Solving for λ yields

$$\lambda = \|g\| / \sqrt{r} - \mu \tag{1.28}$$

which is useful only if $\lambda > 0$ and gives a rough indication for a useful value of λ. Note that this also indicates that λ should shrink with the size of the gradient g. Recall that we have already motivated above that we should move $\lambda \downarrow 0$ as $\|g\| \downarrow 0$. This indicates the following heuristics for λ:

$$\lambda = \kappa \|g\| / \sqrt{r}, \tag{1.29}$$

where $\kappa \in (0, 1)$ e.g. $\lambda = 0.1$.

Remark. In any minimization algorithm the various points which we compute are all minimizers of the quadratic Taylor approximation q to the objective function f centered at the current iterate x_k. We have verified above that they will provide a satisfactory decrease in the value q but not necessarily in the value of f.

A decrease in the value of q implies a decrease in the value of f only if q approximates f sufficiently closely on the ball $B_r(x_k)$. Clearly among the candidates for the next iterate which we have computed:

1. global minimizer $x_k + p^H$ of q,
2. Cauchy point $x_k + p_r^C$,
3. dog-leg point $x_k + p_r^D$ and
4. global minimizer $x_k + p^H(\lambda)$ of $q(x_k + h) + \lambda \|h\|^2$, i.e. the solution of $(H + \lambda I)p = -g$,

We will choose as the next iterate the one that provides the most decrease in the value of f and not in the value of q.

With this information we then also adjust the trust radius r. Suppose for example that the global minimizer $x_k + p^H$ of q is inside the ball $B_r(x_k)$ but does not provide the best decrease in the value of f (while it is guaranteed to provide the best decrease in the value of q).

Then the conclusion must be that q does not approximate f satisfactorily on all of $B_r(x_k)$ and the trust radius r needs to be decreased (note that the Cauchy point and dog-leg point both depend on the radius r).

The dog-leg point is always on the boundary of the ball $B_r(x_k)$ and is not a desirable next iterate if x_k is already very close to a local minimizer x^* of f.

In this case the gradient $g = \nabla f(x_k)$ is close to zero hence the solutions p^H of $Hp = -g$ and $p^H(\lambda)$ of $(H + \lambda I)p = -g$ will also be close to zero so that the points $x_k + p^H$ and $x_k + p^H(\lambda)$ are expected to be close to x^*. In fact if we keep $\lambda > 0$ bounded away from zero this is guaranteed to be true.

This suggests the following mixed strategy: in each step compute the Cauchy point $x_k + p_r^C$, the dog leg point $x_k + p_r^D(\lambda)$ and the point $x_k + p^H(\lambda)$, where

$$\lambda = \max\{\, \|g\| / \sqrt{r} - \mu,\, \kappa \,\}$$

for some small constant $\kappa > 0$. The dog leg step $p_r^D(\lambda)$ is computed for the matrix $H + \lambda I$ i.e. we move in the direction $x_k + p^H(\lambda)$ from the Cauchy point $x_k + p_r^C$. With this strategy we need to solve only one equation $(H + \lambda I)p = -g$ in each step entailing only one expensive matrix factorization. This approach has the advantage that it can handle singular Hessians H and has the further advantage that the regularized global minimizer $x_k + p^H(\lambda)$ is more likely to be inside the trust region $B_r(x_k)$ than the pure global minimizer $x_k + p^H$.

1.2.3 Trust region versus backtracking line search

To compute the step p from the current iterate x to the next iterate $x + p$ we minimize the quadratic Taylor approximation

$$\overline{f}(x + p) = f(x) + g \cdot p + \frac{1}{2} p' H p$$

as a function of the step p, where $g = \nabla f(x)$ is the gradient and $H = D^2 f(x)$ is the Hessian of the objective function f at x. This leads to the equation

$$Hp = -g, \tag{1.30}$$

for the step p and we called the solution the *full Newton step*, denoted $p = p^H$. However we do not expect this step to be optimal for a local minimization of the objective function f itself, since the quadratic approximation $\overline{f}$ of f centered at the current iterate x cannot be trusted to be a good approximation of f globally.

The trust region method tries to identify a spherical region $B_r(x)$ around the current iterate x on which this approximation can be trusted and computes the step by minimizing $\overline{f}(x + p)$ on the ball $B_r(x)$ instead. The problem is the identification of the proper trust radius r and, given r, carrying out the minimization, which is nontrivial.

In this regard we have found the following: if $\lambda > 0$ and $p(\lambda)$ is the solution of the regularized Newton equation

$$(H + \lambda I)p = -g, \tag{1.31}$$

then $x + p(\lambda)$ minimizes the quadratic approximation $\overline{f}(x + p)$ of f centered at x on the ball $B_{r(\lambda)}(x)$, where the radius $r(\lambda)$ is given by $r(\lambda) = \|p(\lambda)\|$. See proposition 1.2.2.

If we have a reasonable idea of a useful trust radius r the problem now becomes that of solving the equation

$$r(\lambda) = r \tag{1.32}$$

for the unknown $\lambda > 0$. This is a nontrivial problem however. We have seen above that it can be solved using the eigenvalue decomposition of H and expanding the gradient g in terms of the eigenvectors of H. We have found that

$$r(\lambda)^2 = \sum\nolimits_j \frac{a_j^2}{(\lambda_j + \lambda)^2},$$

where $Hu_j = \lambda_j u_j$ and $a_j = g \cdot u_j$ is the coefficient of u_j in the expansion of g in terms of the u_j. With this we could in principle solve the equation $r(\lambda)^2 = r^2$ but that approach is too expensive since the eigenvector decomposition of H is expensive.

There is a cheaper approach. The problem is to find a way to cheaply solve the equation (1.31) for many values of λ. The eigenvalue decomposition $H = UDU'$ with orthogonal U and diagonal $D = diag(\lambda_j)$ allows us to do this since it implies the decomposition

$$H + \lambda I = U(D + \lambda I)U'$$

with the same orthogonal matrix U and trivially modified diagonal matrix $D + \lambda I$. Basically this reduces the problem to the solution of the trivial equation $(D + \lambda I)z = -U'g$ followed by a multiplication $p(\lambda) = Uz$. These operations are of order $O(n^2)$ while the eigenvalue decomposition is of order $O(n^3)$ with a large constant.

However we do not need to transform H to a diagonal matrix with orthognal similarity, the same trick applies if we transform H to an upper triangular matrix, i.e we write $H = QRQ'$, where R is upper triangular and Q is orthogonal. This is the so called *Schur decomposition* of H, requires about $(8/3)n^3$ flops and is thus much cheaper than the eigenvalue decomposition of H.

It reveals the eigenvalues of H (as the diagonal elements of R) but we do not get the eigenvectors. With this we do not have a simple formula for $r(\lambda)$ as above but we still have the relation

$$H + \lambda I = Q(R + \lambda I)Q'$$

for all λ and so we can reduce the solution of the equation $(H + \lambda I)p = -g$ to the solution of $(R + \lambda I)z = -Q'g$ followed by $p(\lambda) = Qz$ which impies that $\|p(\lambda)\| = \|z\|$. The matrix $R + \lambda I$ is still upper triangular, so the equation $(R + \lambda I)z = -Q'g$ can be solved cheaply ($n(n + 1)/2$ flops, fewer than the matrix-vector product $Q'g$) so that an iterative approach to the solution of the equation $\|z(\lambda)\| = r$ becomes practical.

Let us note in particular that one single Schur decomposition is enough to handle all values of λ. But the Schur decomposition is still expensive compared to the Cholesky decomposition which only takes $n^3/3$ flops. Thus an iterative algorithm has been devised which is based on the Cholesky decomposition.

However in this algorithm we have to carry out a new Cholesky factorization of $H + \lambda I$ for each new value of λ. Still, since in general fewer than 8 iterations are needed, this is the most economical approach, see the document [2].

Let us note that all these approaches are incompatible with a Ruiz-rebalancing of the matrix H since this will imply a final multiplication of the solution with the diagonal balancing matrix D which alters the norm in uncontrollable ways. We do not need this rebalancing of H however if we know the eigenvalues of H, since we then know the effect of λ on the condition number of $H + \lambda I$.

All this is complicated and still does not resolve the question of determining a useful trust radius r. In practice this is handed by starting with a reasonbable guess and adjusting r in each step based on the performance in the last step. This is imperfect and in particular leaves the following question:

why would a spherical region $B_r(x)$ be preferred as a trust region for the quadratic approximation $\overline{f}$ of f centered at x?

In fact a reasonable trust region has the form $|f(x+p) - \overline{f}(x+p)| < \epsilon$ and it can have any shape whatsoever, certainly it does not need to be spherical. Given the difficulties with spherical trust regions outlined above this is certainly a reasonable question to ask.

Now let us recall the backtracking line search. In this approach we simply minimize the objective function f on the line segment

$$[x, x + p^H] = \{\, x + tp^H \;:\; t \in [0,1] \,\}.$$

I.e. we make the full Newton step p^H and backtrack from there in a straight line. This is certainly a much simpler approach. In this regard it is interesting to note that backtracking line search can also be motivated via trust regions of a different shape, namely of the form

$$D_r(x) = \{\, x + p \;:\; p'Hp \leq r \,\}.$$

This is an elliptical region centered at x defined by the Hessian H of f at x. Suppose now we try to minimize the quadratic approximation $\overline{f}$ on $D_r(x)$ by adding a penalty term $\frac{\lambda}{2}p'Hp$ to $\overline{f}(x+p)$ (for some fixed value $\lambda > 0$) and minimizing $\overline{f}$ plus the penalty term globally, i.e. minimize the function

$$\overline{f}_s(x+p) = \overline{f}(x+p) + \frac{\lambda}{2}p'Hp = f(x) + g'p + \frac{1+\lambda}{2}p'Hp$$

as a function of the step p leading to the equation

$$(1+\lambda)Hp = -g.$$

Clearly the solution $p(\lambda)$ is simply given by

$$p(s) = \frac{1}{1+\lambda}p^H$$

i.e. corresponds to a backtracking operation on the line segment $[x, x + p^H]$. Now exactly the same argument as in the proof of proposition 1.2.2 shows that

Proposition 1.2.4. *The step* $p(\lambda) = (1+\lambda)^{-1}p^H$ *minimizes the quadratic approximation* $\overline{f}(x+p)$ *on the region* $p'Hp \le r(\lambda)$ *where* $r(\lambda) = p(\lambda)'Hp(\lambda)$.

Proof. $p(\lambda)$ is the global minimizer of $\overline{f}(x+p) + (\lambda/2)p'Hp$. Now assume that $p'Hp \le p(\lambda)'Hp(\lambda)$. Then

$$\overline{f}(p(\lambda)) + \frac{\lambda}{2}p'Hp \le \overline{f}(p(\lambda)) + \frac{\lambda}{2}p(\lambda)'Hp(\lambda) \le \overline{f}(p) + \frac{\lambda}{2}p'Hp$$

which implies that $\overline{f}(p(\lambda)) \le \overline{f}(p)$. ∎

As in the case of spherical trust regions this concerns only the minimum of the quadratic approximation $\overline{f}$ of f not the objective function f itself and leaves open the question of where this approximation can be trusted. But there is no reason to prefer spherical trust regions to elliptical ones.

In this case however this entire question becomes irrelevant since we can simply minimize the objective function f itself on the line segment $[x, x+p^H]$ which can be done by a line search algorithm.

One idea is to mix the two approaches (backtracking line search and spherical trust region): first compute $p(\lambda)$ by backtracking line search (exact optimization), then define the trust radius r as $r = \|p(\lambda)\|$ and minimize on the spherical trust region $B_r(x)$ with one of the approaches above.

Example 1.2.1. As an example let us consider the notorious Rosenbrook function

$$f(x,y) = (x-a)^2 + b(y-x^2)^2 \tag{1.33}$$

The global minimum is at $(x,y) = (a, a^2)$ and the function is nasty if b is big, e.g. $a = 1$, $b = 100$. We have

$$f_x = 2(x-a) - 4b(y-x^2)x, \quad f_y = 2b(y-x^2), \quad \text{and thus}$$
$$f_{xx} = 2 - 4b(y-3x^2), \quad f_{xy} = f_{yx} = -4bx, \text{ and } f_{yy} = 2b.$$

The function itself is a polynomial and the quadratic Taylor approximation $\overline{f}(h,k)$ of $f(x+h, y+k)$ contains all the second order terms in h, k while the residual $f(x+h,y+k) - \overline{f}(h,k)$ contains the rest and is easily seen to be the quantity

$$f(x+h,y+k) - \overline{f}(h,k) = h^2(bxh + h^2 - 2bk)$$

Thus reasonable trust regions for this function have the form

$$|h^2(bxh + h^2 - 2bk)| \le \epsilon$$

and are neither spherical nor elliptical. ∎

1.2.4 Iteration to refine λ

Recall that we want to find a regularization parameter $\lambda > 0$ such that the solution $p(\lambda)$ of the equation $(H+\lambda I)p = -g$ satisfies $\tilde{r} := \|p(\lambda)\| \approx r$, where r is the value of the trust radius.

Here we have a reasonable guess for a starting value $\lambda_0 = \|g\|/\sqrt{r}$ which however overestimates λ in every case (λ_0 is too large). This means that $p(\lambda)$ will certainly be in $B_r(x)$ but may be very close to the current iterate x, which is undesirable unless this iterate is already close to the minimizer of f. See equation (1.27) in section 1.2.2 and subsequent discussion.

With one single eigenvalue or Schur decomposition of $H + \lambda_0 I$ a cheap iterative algorithm to refine λ can be implemented. However we want to work with Cholesky factorizations (since these are both more accurate and much faster than both the Schur decomposition and the Eigenvalue decomposition). Moreover we want to keep the number of Cholesky factorizations to a minimum.

In reference [2] such an iteration for λ is developped which however requires a Cholesky factorization of $H + \lambda I$ for each new value of λ. We can get the iteration from the Cholesky factorization of

$$H + \lambda I = LL'$$

as follows:

$$\lambda_{next} = \lambda + \frac{\|p(\lambda)\| - r}{r} \times \frac{\|p(\lambda)\|^2}{\|w\|^2}, \quad \text{where} \tag{1.34}$$
$$Lw = p(\lambda).$$

However we do then need a second Cholesky factorization of $H + \lambda_{next} I$ to compute the new step $p(\lambda_{next})$.

1.3 Line search methods

A robust line search is the backbone of our optimization routines. At the current iterate x we have a direction p^H (the full Newton step) and the global minimizer $x^H := x + p^H$ of the quadratic Taylor approximation $\overline{f}$ of the objective function f.

As a first step we minimize f on the line segment $[x, x + p^H]$, that is we minimize the function

$$\phi(t) = f(x + tp^H), \quad t \in [0, 1].$$

In principle we have both the first and second derivative of ϕ as

$$\phi'(t) = \nabla f(x + tp^H) \cdot p^H \quad \text{and} \quad \phi''(t) = (p^H)' H(x + tp^H) p^H,$$

where $H(z)$ is the Hessian of f at z. Here the computation of values of f is relatively cheap, but the gradient is more expensive and the Hessian is most expensive. For this reason we only discuss methods that make no use of the second derivative. This rules out the Newton method.

In practice we will not be able to avoid the use of the first derivative since line search termination conditions are based on derivative information.

1.3.1 Methods which do not use derivative information

We start with methods that use only the values of the objective function ϕ but not the derivative ϕ'.

Golden Search

Suppose we want to minimize a function $\phi = \phi(t)$ on the interval $[0, 1]$. The algorithm produces a sequence triples $0 \leq a_k < c_k < b_k \leq 1$. The point c_k splits the current interval $[a_k, b_k]$ which brackets a minimizer and the search continues in one of the subintervals $[a_k, c_k], [c_k, b_k] \subseteq [a_k, b_k]$.
The split maintains the invariant

$$|I| = \rho|J|, \tag{1.35}$$

where I is the longer and J the shorter of the subintervals $[a_k, c_k], [c_k, b_k] \subseteq [a_k, b_k]$. The following invariant is also maintained, once it is obtained:

$$\phi(c_k) \leq \min(\phi(a_k), \phi(b_k)) \tag{1.36}$$

This condition ensures that the open interval (a_k, b_k) contains a local minimum of ϕ. If this condition is never obtained (boundary minimum), then each triple contains the best minimizer found so far. The points are spaced in such a fashion that the length of the interval $[a_k, b_k]$ decreases geometrically.

In the description below we assume that condition (1.36) has been obtained, the other case being trivial. The algorithm proceeds as follows: we split the larger of the two subinterval $[a_k, c_k]$ and $[c_k, b_k]$, *wolog* the interval $[c_k, b_k]$ into subintervals $[c_k, d_k]$ and $[d_k, b_k]$. There are now two cases:

(A) $\phi(d_k) \leq \phi(c_k)$. Then we have $\phi(d_k) \leq \min\{\phi(c_k), \phi(b_k)\}$ and we can replace the triple $a_k < c_k < b_k$ with the triple $c_k < d_k < b_k$.

(B) $\phi(c_k) \leq \phi(d_k)$. Then we have $\phi(c_k) \leq \min\{\phi(a_k), \phi(d_k)\}$ and we can replace the triple $a_k < c_k < b_k$ with the triple $a_k < c_k < d_k$.

In every case the invariant (1.36) is maintained. The question now becomes how we place the point d_k. We have $a_k < c_k < d_k < b_k$. Let

$$\alpha = c_k - a_k,\ \beta = d_k - c_k \text{ and } \gamma = b_k - d_k$$

the length of the three subintervals. Since the next interval will either be $[a_k, d_k]$ or $[c_k, b_k]$ it is reasonable to require that these have the same length, i.e. $\alpha + \beta = \beta + \gamma$ from which we obtain

$$\alpha = \gamma.$$

Next we require that the proportion in the spacing between the points be maintained by passing from the triple (a_k, c_k, b_k) to the triple (c_k, d_k, b_k), i.e.:

$$\frac{\alpha}{\beta + \gamma} = \frac{\beta}{\gamma} \quad \text{i.e.} \quad \frac{\alpha}{\beta + \alpha} = \frac{\beta}{\alpha} := \rho.$$

Note that this implies $\alpha > \beta$ and thus $\rho = \beta/\alpha$ is also the corresponding ratio for the triple (a_k, c_k, d_k). It follows that this ratio is maintained as an invariant of the algorithm in both cases (A) and (B).
Dividing by α we obtain $\rho = 1/(1+\rho)$, i.e. $\rho^2 + \rho - 1 = 0$ from which

$$\rho = (\sqrt{5} - 1)/2$$

is the golden ratio. Let us note in passing that the relation $\rho = 1/(1+\rho)$ implies that all the continued fractions $[1, 1, \ldots, 1, \rho]$ are equal to ρ. To see this, simply replace ρ in the fraction $1/(1+\rho)$ with $1/(1+\rho)$ and continue such replacements. Note that the spacing of the point c_k is given by the fraction

$$\frac{b_k - c_k}{b_k - a_k} = \frac{\beta + \gamma}{\alpha + \beta + \gamma} = \frac{\alpha + \beta}{\alpha + \beta + \alpha} = \frac{1}{1 + \alpha/(\alpha + \beta)} = \frac{1}{1 + \rho} = \rho.$$

The new interval is $[c_k, b_k]$ or $[a_k, d_k]$, both of equal length. Thus the interval length $b_k - a_k$ decreases geometrically by a factor of $\rho = 0.618034$ in each step. We have seen above that $\rho = \alpha/(\beta + \gamma)$ and so

$$\frac{c_k - a_k}{b_k - c_k} = \frac{\alpha}{\beta + \gamma} = \rho = \frac{b_k - c_k}{b_k - a_k}$$

i.e.: the fraction of the shorter to the longer subinterval = the fraction of the longer subinterval to the whole. This is the rule of the Golden Ratio and that is where this method derives its name.

In this description we have *assumed* that $[c_k, b_k]$ is the longer subinterval of $[a_k, b_k]$. But not in general will the longer subinterval be the interval on the right. Suppose for example that we pass to the triple (a_k, c_k, d_k). We have seen above that $\alpha > \beta$, i.e. in this triple $[a_k, c_k]$ is the longer subinterval and is the interval that has to be split in the next step.

Algorithm: start with $a_k = 0$, $b_k = 1$ and set $c_k = \rho$. Given any triple (a_k, c_k, b_k) where the point c_k splits the interval $[a_k, b_k]$ such that

$$\text{longer subinterval} = \rho \times \text{full interval}$$

split the longer subinterval with the point d_k in the same proportion and such that these proportions are maintained in both subsequent triples (this determines if the shorter subinterval is on the right or on the left).

This gives us two new triples of points $(a_k, c_k \wedge d_k, c_k \vee d_k)$ and $(c_k \wedge d_k, c_k \vee d_k, b_k)$. Move to the triple which contains the best minimizer so far.

Quadratic Interpolation

Given a triple of points $a < c < b$ with associated values $\phi(a)$, $\phi(b)$ and $\phi(c)$ we now approximate ϕ with a quadratic polynomial $P(t)$ which interpolates the points $(a, \phi(a))$, $(b, \phi(b))$ and $(c, \phi(c))$, then minimize this polynomial for a final candiate point for the minimum. Finally we retain the point at which the smallest value of ϕ so far occurs.

This is suitable for mixing with the strategy of Golden Search since at each iteration of Golden search we have exactly such a triple of points.
Note that $P(t)$ has the form

$$P(t) = \phi(a)\frac{(t-b)(t-c)}{(a-b)(a-c)} + \phi(b)\frac{(t-a)(t-c)}{(b-a)(b-c)} + \phi(c)\frac{(t-a)(t-b)}{(c-a)(c-b)}.$$

from which it follows that

$$P'(t) = \phi(a)\frac{2t-(b+c)}{(a-b)(a-c)} + \phi(b)\frac{2t-(a+c)}{(b-a)(b-c)} + \phi(c)\frac{2t-(a+b)}{(c-a)(c-b)}.$$

Setting the derivative equal to zero yields

$$\begin{aligned} &t = N/D, \quad \text{where} \\ &N = \phi(a)\frac{b+c}{(a-b)(a-c)} + \phi(b)\frac{a+c}{(b-a)(b-c)} + \phi(c)\frac{a+b}{(c-a)(c-b)} \quad \text{and} \\ &D = \phi(a)\frac{2}{(a-b)(a-c)} + \phi(b)\frac{2}{(b-a)(b-c)} + \phi(c)\frac{2}{(c-a)(c-b)}. \end{aligned}$$

Here we need to check that $P''(t) > 0$ to verify that the interpolating polynomial P does assume a minimum. To this end we note that $P''(t) = D$, for all t. Note that the above formulas can be simplified by multiplying both N and D with $(a-b)(a-c)(b-c) > 0$ to obtain

$$t = \frac{1}{2} \times \frac{\phi(a)(b^2-c^2) + \phi(b)(c^2-a^2) + \phi(c)(a^2-b^2)}{\phi(a)(b-c) + \phi(b)(c-a) + \phi(c)(a-b)}. \tag{1.37}$$

and we note that $P''(t)$ has the same sign as the new denominator.

1.3.2 Methods which use the first derivative

Now we move on to methods which use the first derivative

$$\phi'(t) = \nabla f(x + tp^H) \cdot p^H.$$

Quadratic Interpolation

With derivative information we need only two points to determine the quadratic interpolating polynomial. Say we have points a, b and associated values $\phi(a)$, $\phi(b)$ as well as the derivative $\phi'(a)$. Then there is a unique quadratic polynomial $P(t)$ which interpolates ϕ at the points $(a, \phi(a))$ and $(b, \phi(b))$ and which satisfies $P'(a) = \phi'(a)$. We then minimize this polynomial to obtain a new candidate point.
Develop P centered at a to obtain

$$P(t) = \phi(a) + \phi'(a)(t-a) + q(t-a)^2,$$

where the coefficient q is determined from the equation $P(b) = \phi(b)$ yielding

$$q = \frac{\phi(b) - \phi(a) - \phi'(a)(b-a)}{(b-a)^2}$$

and setting the derivative $P'(t) = 0$ we obtain the location of the minimum as

$$t = a - \frac{\phi'(a)}{2q}. \tag{1.38}$$

Here we need to check that $q > 0$ to ensure that the interpolating polynomial does in fact have a minimum.

1.3.3 Trust radius from line search

Since the line search is relatively cheap we can try to derive a guess for the trust radius based on a study of the objective function f on the line segment $[x, x+p^H]$. Clearly the global minimizer $x+p^H$ of the quadratic approxiation $\overline{f}$ of f also minimizes $\overline{f}$ on this line segment. Actually the quadratic approximation $\overline{f}(x + tp^H)$ is *decreasing* on this entire line segment.

To see this recall that $p := p^H$ is the solution of $Hp = -g$, where $g = \nabla f(x)$ and $H = \nabla^2 f(x)$ is the hessian of f at the current iterate x. With this the quadratic approximation $\overline{f}$ has the form

$$\overline{f}(x + h) = f(x) + g'h + \frac{1}{2}h'Hh$$

and especially for $h = tp$, $t \in [0, 1]$ we obtain

$$\overline{f}(x + tp) = f(x) + tg'p + \frac{1}{2}t^2p'Hp$$

from which, using $Hp = -g$,

$$\frac{d}{dt}\overline{f}(x + tp) = g'p - tg'p = (1 - t)g'p$$

However it is easy to see that $g \cdot p < 0$ by writing both g and p as linear combinations of the eigenvectors of H. Indeed this is true for all the steps $p(\lambda)$ computed as the solutions of the regularized Newton equation $(H + \lambda I)p = -g$, where $\lambda \geq 0$. Indeed, if $Hu_j = \lambda_j u_j$ and

$$g = \sum\nolimits_j a_j u_j$$

then the solution $p(\lambda)$ of $(H + \lambda I)p = -g$ is given by

$$p(\lambda) = -\sum \frac{a_j}{\lambda_j + \lambda} u_j,$$

where all the eigenvalues λ_j are strictly positive (convexity of f). It readily follows that

$$g'p(\lambda) = g \cdot p(\lambda) = -\sum\nolimits_j \frac{a_j^2}{\lambda_j + \lambda} < 0,$$

i.e every step $p(\lambda)$ is a step in a descent direction of f and we have

$$\begin{aligned}\frac{d}{dt}\overline{f}(x + tp(\lambda)) &= g'p(\lambda) + tp(\lambda)'Hp(\lambda) \\ &= g'p(\lambda) + tp(\lambda)'(H + \lambda I)p(\lambda) - t\lambda \left\|p(\lambda)\right\|^2 \\ &= (1-t)g'p(\lambda) - t\lambda \left\|p(\lambda)\right\|^2 = g'p(\lambda) - t(g'p(\lambda) + \lambda \left\|p(\lambda)\right\|^2).\end{aligned}$$

Assume now that $\lambda \geq 0$ is so small that $g'p(\lambda) + \lambda \left\|p(\lambda)\right\|^2$ is still negative. Then

$$\frac{d}{dt}\overline{f}(x + tp(\lambda)) \leq 0, \quad \text{for all } 0 \leq t \leq \frac{g'p}{g'p + \lambda \left\|p(\lambda)\right\|^2}. \tag{1.39}$$

So for $\lambda = 0$, where $p(\lambda) = p^H$ is the full Newton step, this derivative is negative, and hence the quadratic approximation $\overline{f}$ decreasing, on the full interval $[0, 1)$. For the steps $p(\lambda)$ this holds on an even larger interval $[0, 1 + \epsilon)$ but these steps are smaller than the full Newton step. This means that the quadratic approximation will always assume its minimum at the point $x + p$ on the line segment $[x, x + p]$.

Now this is not necessarily true also for the objective function f itself and this is where we can get information about the trust radius: the maximum value for the trust radius would be the size $r = \|p\|$ of the full step p. However if we detect that the objective function $f(x + tp)$ assumes its minimum at a point $t < 1$, then the trust radius needs to be reduced below the value $r = t\|p\|$.
This applies to the full Newton step $p = p^H$ as well as to the regularized Newton steps $p = p(\lambda)$ and suggests the following algorithm: start off with a small value of λ, e.g. $\lambda = \min(0.001, \|g\|/10)$ and compute the corresponding point $p(\lambda)$. Now conduct golden search on the line segment $[x, x + p(\lambda)]$.

At every triple $a < c < b$ the quadratic approximation $\overline{f}$ satisfies $\overline{f}(a) > \overline{f}(c) > \overline{f}(b)$ and the expectation is that the objective function f satisfies the same inequalities. So long as this is the case we continue with golden search, always moving to the right (choose the triple containing the current minimizer b).

If the inequality $f(a) > f(c) > f(b)$ is maintained for three or four iterations, the trust radius r is set equal to $\|p(\lambda)\|$.
When these inequalities fail for the first time there are two possibilities:

(A) $f(a)$ is the smallest value. We reduce the trust radius to $r = a\|p(\lambda)\|$ and refine λ as described in section 1.2.4.

(B) $f(c)$ is the smallest value. Then we know that the quadratic polynomial interpolating the points $(a, f(a))$, $(b, f(b))$ and $(c, f(c))$ must have its minimum in the open interval (a, b). We compute the location of this minimum as described in section 1.3.1 and either continue to narrow down the location of the local

minimum or reduce the trust radius immediately to the value $r = t_* \|p(\lambda)\|$, where t_* is the location of the current minimum in the interval $(0, 1)$.

Bibliography

[1] Jorge Nocedal, Stephen J. Wright *Numerical Optimization*, 2nd edition, 2006, Springer Verlag.

[2] Nick Gould, *Trust-region methods for unconstrained optimization*, `https://www.numerical.rl.ac.uk/people/nimg/msc/lectures/part3.2.pdf`