SECTION C: CONTINUOUS OPTIMISATION LECTURE 15: THE BARRIER METHOD FOR NONLINEAR PROGRAMMING

HONOUR SCHOOL OF MATHEMATICS, OXFORD UNIVERSITY
HILARY TERM 2005, DR RAPHAEL HAUSER

1. The Merit Function. Again we consider the general nonlinear programming problem

$$\text{(NLP)} \qquad \begin{aligned} &\min_{x\in\mathbb{R}^n} f(x) \\ &\text{s.t.} \quad g_{\mathcal{E}}(x) = 0 \\ &\qquad\ \ g_{\mathcal{I}}(x) \geq 0. \end{aligned}$$

In the previous two Lectures we have learned that nonlinear constraints can be dealt with by incorporating a term forcing asymptotic feasibility into the objective function. So far we used quadratic penalty terms to construct merit functions. The *barrier method* makes another choice and is based on the merit function

$$P(x,\mu) = f(x) - \mu \sum_{j\in\mathcal{I}} \ln g_j(x) + \frac{1}{2\mu} \sum_{i\in\mathcal{E}} g_i^2(x),$$

where $\mu > 0$ is the *homotopy parameter*. Note that equality constraints are again enforced using quadratic penalty terms that become gradually more stringent. The penalty terms are defined for all x. Inequality constraints are managed via the *barrier term*

$$-\mu \sum_{j\in\mathcal{I}} \ln g_j(x)$$

which is only defined when all the $g_j(x)$ are strictly positive.

Definition 1.1. *A point $x \in \mathbb{R}^n$ is* admissible *for (NLP) if all inequality constraints are satisfied. The point is called* strictly admissible *if all inequality constraints are strictly satisfied, that is,*

$$g_j(x) > 0 \qquad (j = \mathcal{I})$$

holds. Note that admissible points may violate some or all of the equality constraints. The sets of admissible and strictly admissible points respectively are called admissible *and* strictly admissible domain.

The barrier term is thus only defined for strictly admissible points. We can extend it outside the admissible domain by the convention

$$P(x,\mu) = \begin{cases} P(x,\mu) & (x \text{ admissible}), \\ +\infty & (x \text{ inadmissible}). \end{cases}$$

Note also that if $(x_k)_{\mathbb{N}}$ is a sequence of admissible points such that $x_k \to x^*$, where x^* is on the boundary of the admissible domain, then there exists $j \in \mathcal{I}$ such that $g_j(x_k) \to g_j(x^*) = 0$, and then $\lim_{k\to\infty} P(x_k,\mu) = +\infty$.

Similarly to Algorithms QPen and AL, in each inner loop of the barrier method a value $\mu_k > 0$ is fixed and an unconstrained subproblem

$$\min_{y \in \mathbb{R}^n} P(y, \mu_k) \tag{1.1}$$

is solved.

When we studied the method of Lagrange multipliers we saw that if the set of active constraints at an optimal solution x^* was known, then (NLP) would reduce to a nonlinear zero-finding problem. Most of the techniques we developed for unconstrained minimisation were based on a reduction to a zero-finding problem and have a straightforward generalisation to the latter. Thus, constrained optimisation is the same as a finite number of unconstrained problems with the extra difficulty of finding which among the unconstrained problems yields the optimal solution. In other words, the optimal active set $\mathcal{A}(x^*)$ has to be found, a combinatorial overhead that is challenging, because the number of possible choices for $\mathcal{A}(x^*)$ is exponential in the number of constraints.

The power of homotopy methods derives from avoiding that the full complexity of this combinatorial overhead problem comes to the bear initially. In fact, the combinatorial problem is only introduced gradually through penalty and/or barrier terms. The approach works because as an optimal solution x^* is approached, the number of different choices for $\mathcal{A}(x^*)$ narrows down dramatically. The barrier method achieves this reduction in complexity because the barrier term keeps the iterates strictly admissible and only gradually allows them to approach the boundary, and the quadratic penalty term on the equality constraints ensures that the constraints are asymptotically satisfied, by making the equality constraint manifold $\{x : g_{\mathcal{E}}(x) = 0\}$ attracting.

A mechanistic interpretation is as follows: imagine that a positive point charge is moving in the admissible domain under the influence of a force acting on the particle due to an electrostatic potential described by f. The boundary of the admissible domain is positively charged, so that the point particle is repelled from it, and the charge on the boundary is gradually reduced as μ decreases. On the other hand, the equality constraint manifold is negatively charged, so that it attracts the point charge, and the the charge is increased as μ decreases. See Figure 1.1 for an illustration. Of course, this mechanistic interpretation is just a mental picture; neither the barrier nor the penalty term have gradients that are proportional to electrostatic forces as a function of position. Instead, they are chosen so as to render the algorithms efficient in practice.

2. The Primal Barrier Method for Nonlinear Programming.

We are ready to formulate a first algorithm based on the merit function introduced above.

```
ALGORITHM 2.1. [PBM]
S0 Initialisation
        choose $\mu_{-1} > \mu_0 > 0$
        choose $\epsilon_0 > 0$
        choose $x_0$ strictly admissible
S1 For $k = 0, 1, 2, \dots$ repeat
        solve the following equation for $\dot{x}$,
```

$$D^2_{xx} P(x_k, \mu_{k-1}) \dot{x} + \frac{\partial}{\partial \mu} \nabla_x P(x_k, \mu_{k-1}) = 0$$

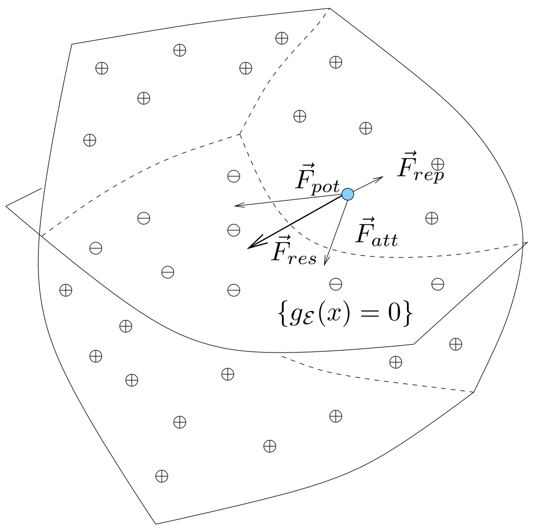

FIG. 1.1. *Mechanistic interpretation of the merit function: the external forces acting on a positive point charge are the attracting force $\vec{F}_{att}$ towards the equality constraint manifold, the repelling force $\vec{F}_{rep}$ away from inequality constraints, and the force $\vec{F}_{pot}$ due to the potential, resulting in the total force $\vec{F}_{res}$.*

```
        $y^{[0]} := x_k + (\mu_k - \mu_{k-1})\dot{x}$
        l:=0
        until $\|\nabla_x P(y^{[l]}, \mu_k)\| \le \epsilon_k$ repeat
            compute $y^{[l+1]}$ such that $P(y^{[l+1]}, \mu_k) < P(y^{[l]}, \mu_k)$
                % (using unconstrained minimisation method)
            $l \leftarrow l + 1$
        end
        $x_{k+1} := y^{[l]}$
        choose $\mu_{k+1} < \mu_k$        % (usually $\mu_{k+1} = \theta\mu_k$ for some $\theta \in (0,1)$)
        choose $\epsilon_{k+1} < \epsilon_k$
    end
```

2.1. Finding an Admissible Starting Point. Note that in the initialisation step we need to find a strictly admissible point. This is not a problem, because we can use the same algorithm to generate such a point by solving the auxiliary optimisation problem

$$\text{(AUX)} \qquad \min_{(x,t)} t$$
$$\text{s.t.} \quad g_j(x) + t \ge 0, \qquad (j \in \mathcal{I}),$$

for which finding an admissible initial solution can be readily generated as follows,

$$x_0 \in \mathbb{R}^n \qquad \text{arbitrary,}$$
$$t_0 := -\min\{g_j(x_0) : j \in \mathcal{I}\} + 1,$$

and for which the algorithm will eventually find $t_k < 0$ if and only if (NLP) has strictly admissible points. When such a t_k is found, x_k is strictly admissible for (NLP) and can be used as a starting point for Algorithm PBM applied to (NLP). The auxiliary problem (AUX) is sometimes referred to as *phase I problem.*

2.2. Convergence of the Algorithm. The termination criterion in the inner loop guarantees that

$$\nabla_x P(x_k, \mu_{k-1}) = \nabla f(x_k) - \sum_{j\in\mathcal{I}} \frac{\mu_{k-1}}{g_j(x_k)} \nabla g_j(x_k) + \sum_{i\in\mathcal{E}} \frac{g_i(x_k)}{\mu_{k-1}} \nabla g_i(x_k) = O(\epsilon_{k-1}). \quad (2.1)$$

Arguments similar to those used Lecture 13 show the following convergence result:

THEOREM 2.2. *Let $x^* = \lim_{l\to\infty} x_{k_l}$ be an accumulation point of the sequence $(x_k)_{\mathbb{N}_0}$ generated by Algorithm PBM, where $(k_l)_{\mathbb{N}_0}$ is a subsequence of $(k)_{\mathbb{N}_0}$. If the set of gradient vectors $\{\nabla g_i(x^*) : i \in \mathcal{V}(x^*)\}$ is linearly independent, then the following properties hold true:*

(i) x^* is feasible,
(ii) the LICQ holds at x^*,
(iii) the limit $\lambda^* = \lim_{l\to\infty} \lambda^{[k_l]}$ exists, where

$$\lambda_i^{[k]} = \begin{cases} \frac{\mu_{k-1}}{g_i(x_k)}, & (i \in \mathcal{I}) \\ -\frac{g_i(x_k)}{\mu_{k-1}} & (i \in \mathcal{E}), \end{cases} \quad (2.2)$$

(iv) (x^*, λ^*) is a KKT point of (NLP).

2.3. Selection of the Starting Point in the Inner Loop. Note that Algorithm PBM determines a starting point $y^{[0]}$ for the inner loop in a somewhat intricate way. Why not use $y^{[0]} = x_k$ instead? While this is certainly a valid choice, it leads to poor convergence because the step sizes are restricted to small values. Let us now analyse this phenomenon and assume that the starting point $y^{[0]} = x_k$ is chosen for the inner loop:

When a Newton step is applied to the subproblem (1.1) at $y = x_k$: the Newton–Raphson update $\Delta_x \in \mathbb{R}^n$ satisfies the system

$$D^2_{xx} P(x_k, \mu_k)\Delta_x = -\nabla_x P(x_k, \mu_k). \quad (2.3)$$

But

$$\begin{aligned} D^2_{xx} P(x_k, \mu_k) &= \left[D^2 f(x_k) - \frac{\mu_k}{\mu_{k-1}} \sum_{j\in\mathcal{I}} \Big(\frac{\mu_{k-1}}{g_j(x_k)}\Big) D^2 g_i(x_k) - \frac{\mu_{k-1}}{\mu_k} \sum_{i\in\mathcal{E}} \Big(-\frac{g_i(x_k)}{\mu_{k-1}}\Big) D^2 g_i(x_k) \right] \\ &\quad + \left[\sum_{j\in\mathcal{I}} \frac{\mu_k}{g_j^2(x_k)} \nabla g_j(x_k) \nabla g_j(x_k)^{\mathrm{T}} + \frac{1}{\mu_k} \sum_{i\in\mathcal{E}} \nabla g_i(x_k) \nabla g_i(x_k)^{\mathrm{T}} \right] \\ &= C(x_k, \mu_k) + A^{\mathrm{T}}(x_k, \mu_k) A(x_k, \mu_k), \end{aligned}$$

where the columns of the matrix A^{T} consists of the vectors

$$\left\{\frac{\sqrt{\mu_k}}{g_j(x_k)}\nabla g_j(x_k) : (j \in \mathcal{I})\right\} \cup \left\{\frac{1}{\sqrt{\mu_k}}\nabla g_i(x_k) : (i \in \mathcal{E})\right\}.$$

Note that (2.2) implies $\|C(x_k, \mu_k)\| = O(1)$, whereas

$$\left\|A^{\mathrm{T}}(x_k, \mu_k)A(x_k, \mu_k)\right\| = O\left(\max\left(\mu_k^{-1}, \max_i \left(g_i(x_k)^{-1}\right)\right)\right).$$

On the one hand, this leads to an ill-conditioned Newton system (2.3), unless

$$A^{\mathrm{T}}(x^*, 0)A(x^*, 0) := \lim_{l\to\infty} A(x_{k_l}, \mu_{k_l})$$

has either rank 0 or n. This is a minor problem, because a careful implementation as outlined in the last paragraph of Lecture 12 replaces the system (2.3) by a well-conditioned equivalent one.

On the other hand, a problem that affects the performance of the primal barrier method far more adversely is that the solution Δ_x of (2.3) is badly scaled: whenever $D^2_{xx}P(x_k, \mu_k)$ is ill-conditioned due to $\operatorname{rank}\big(A^{\mathrm{T}}(x^*, 0)A(x^*, 0)\big) \notin \{0, n\}$, we have

$$\begin{aligned}
&\sum_{j\in\mathcal{V}(x^*)\cap\mathcal{I}} \frac{\mu_k \nabla g_j^{\mathrm{T}}(x_k)\Delta_x}{g_j^2(x_k)}\nabla g_j(x_k) + \sum_{i\in\mathcal{E}} \frac{\nabla g_i^{\mathrm{T}}(x_k)\Delta_x}{\mu_k}\nabla g_i(x_k)\\
&\approx A^{\mathrm{T}}(x_k, \mu_k)A(x_k, \mu_k)\Delta_x \approx D^2_{xx}P(x_k, \mu_k)\Delta_x = -\nabla_x P(x_k, \mu_k)\\
&\approx -\nabla f(x_k) + \sum_{j\in\mathcal{V}(x^*)\cap\mathcal{I}} \frac{\mu_k}{g_j(x_k)}\nabla g_j(x_k) - \sum_{i\in\mathcal{E}} \frac{g_i(x_k)}{\mu_k}\nabla g_i(x_k)\\
&\overset{(2.1)}{\approx} \sum_{j\in\mathcal{V}(x^*)\cap\mathcal{I}} \frac{\mu_k - \mu_{k-1}}{g_j(x_k)}\nabla g_j(x_k) + \sum_{i\in\mathcal{E}} \frac{(\mu_k - \mu_{k-1})g_i(x_k)}{\mu_k\mu_{k-1}}\nabla g_i(x_k).
\end{aligned}$$

Since the LICQ holds at x^*, this implies that

$$\nabla g_j^{\mathrm{T}}(x_k)\Delta_x \approx \left(1 - \frac{\mu_{k-1}}{\mu_k}\right) g_j(x_k), \qquad (j \in \mathcal{I} \cap \mathcal{V}(x^*)),$$

and hence,

$$g_j(x_k + \Delta_x) \approx g_j(x_k) + \nabla g_j^{\mathrm{T}}(x_k)\Delta_x \approx \left(2 - \frac{\mu_{k-1}}{\mu_k}\right) g_j(x_k).$$

But $2 - \mu_{k-1}/\mu_k < 0$ for $\mu_k < \mu_{k-1}/2$, and this shows that only a modest reduction of μ_k is possible in each iteration, because otherwise the Newton step applied at $y^{[0]} = x_k$ takes the iterate outside of the admissible domain and strong damping must be used to shorten the step size.

Of course, this only shows that using $y^{[0]} = x_k$ as a starting point for the inner loop is a bad choice; it can be shown that the formula for $y^{[0]}$ given in Algorithm PBM behaves much better.

3. The Primal-Dual Barrier Method. The bad scaling of the primal barrier method can be overcome by exploiting the fact that Lagrange multiplier estimates become available as μ is decreased. This leads to the *primal-dual barrier method* which we will describe next.

The relationship between the primal barrier method and the primal-dual barrier method is in some ways similar to the relationship between the quadratic penalty function method and the augmented Lagrangian method of Lectures 13 and 14.

The KKT conditions of (NLP) are as follows,

$$
\begin{aligned}
\nabla f(x) - \sum_i \lambda_i \nabla g_i(x) &= 0 && (3.1)\\
g_{\mathcal{E}}(x) &= 0 && (3.2)\\
\lambda_j g_j(x) &= 0 \qquad (j \in \mathcal{I}) && (3.3)\\
\lambda_{\mathcal{I}}, g_{\mathcal{I}}(x) &\geq 0. && (3.4)
\end{aligned}
$$

The primal-dual barrier method can be motivated by the following two ideas:

(i) guarantee that (3.4) holds through the application of line searches to prevent iterates to become inadmissible,

(ii) perturb the right hand side of the complementarity equations (3.3) by μ.

Thus, in each iteration of the algorithm one or several damped Newton steps are applied to the nonlinear system of equations

$$
\left.
\begin{aligned}
\nabla f(x) - \sum_i \lambda_i \nabla g_i(x) &= 0\\
g_{\mathcal{E}}(x) &= 0\\
\lambda_j g_j(x) &= \mu. \qquad (j \in \mathcal{I})
\end{aligned}
\right. \qquad (3.5)
$$

The corresponding Newton system is

$$
\begin{array}{rrl}
\left(D^2 f(x) - \sum_i \lambda_i D^2 g_i(x)\right)\Delta_x & -\left(g'_{\mathcal{I}\cup\mathcal{E}}(x)\right)^{\mathrm{T}}\Delta_\lambda & = -\nabla f(x) + \left(g'_{\mathcal{I}\cup\mathcal{E}}(x)\right)^{\mathrm{T}}\lambda\\
g'_{\mathcal{E}}(x)\Delta_x & & = -g_{\mathcal{E}}(x)\\
\mathrm{Diag}(\lambda) g'_{\mathcal{I}}(x)\Delta_x & +\mathrm{Diag}(g_{\mathcal{I}}(x))\Delta_\lambda & = \begin{bmatrix}\mu\\ \vdots\\ \mu\end{bmatrix} - \mathrm{Diag}(\lambda) g_{\mathcal{I}}(x)
\end{array}, \qquad (3.6)
$$

where $\mathrm{Diag}(v)$ denotes a diagonal matrix with a vector v on its diagonal. The algorithm is as follows:

Algorithm 3.1 (PDBM).

```
S0 Initialisation
       choose $\mu_{-1} > \mu_0 > 0$
       choose $\epsilon_0 > 0$
       choose $x_0$ strictly admissible
       choose $\lambda^{[0]}$ such that $\lambda^{[0]}_{\mathcal{I}} \geq 0$
       choose $\theta \in (0, 1)$
S1 For $k = 0, 1, 2, \ldots$ repeat
       $y^{[0]} = x_k$
       $\eta^{[0]} = \lambda^{[k]}$
       $l := 0$
       $\Delta_x, \Delta_\lambda = +\infty$
       until $\|\Delta_x, \Delta_\lambda\| \leq \epsilon_k$ repeat
           solve (3.6) for $(\Delta_x, \Delta_\lambda)$, setting $(x, \lambda, \mu) := (y^{[l]}, \eta^{[l]}, \mu_k)$
           $\alpha_{\max} := \max\{\alpha \geq 0 :\ g_{\mathcal{I}}(y^{[l]} + \alpha\Delta_x) \geq 0,\ (\eta^{[l]} + \alpha\Delta_\lambda)_{\mathcal{I}} \geq 0\}$
           choose $\alpha_l \in (0, \max\{1, 0.98 \times \alpha_{\max}\}]$
           $(y^{[l+1]}, \eta^{[l+1]}) = (y^{[l]}, \eta^{[l]}) + \alpha_l(\Delta_x, \Delta_\lambda)$
           $l \leftarrow l + 1$
       end
       $(x_{k+1}, \lambda^{[k+1]}) := (y^{[l]}, \eta^{[l]})$
       $\mu_{k+1} = \theta\mu_k$
       choose $\epsilon_{k+1} \in (0, \epsilon_k)$
   end
```

Note that we initialised the starting vector for the inner loop by $y^{[0]} = x_k$. In contrast to the primal barrier method, the primal-dual barrier method works fine with this choice, as a somewhat intricate analysis shows. In Lecture 16 we will analyse the primal-dual barrier method for linear programming in further detail.