Some notes on inexact SQP methods  
(not authorized, for internal OPTEC use only)  

Moritz Diehl  
April 20, 2010  

1 Introduction  
We are concerned with solving the nonlinear optimization problem  
\[ \min_{x \in \mathbb{R}^n} f(x) \quad \text{s.t.} \quad \begin{cases} g(x) = 0 \\ h(x) \geq 0. \end{cases} \]  
with Lagrangian  
\[ \mathcal{L}(x, \lambda, \mu) = f(x) - \lambda^T g(x) - \mu^T h(x) \]  
SQP methods successively approximate this problem by a QP  
\[ \min_{x \in \mathbb{R}^n} f_{\text{LIN}}(x) \quad \text{s.t.} \quad \begin{cases} g_{\text{LIN}}(x) = 0 \\ h_{\text{LIN}}(x) \geq 0. \end{cases} \]  
where the approximating functions are usually defined at some linearization point \( \bar{x}, \bar{\lambda}, \bar{\mu} \) as follows, (using the convention \( \nabla g(x) = \frac{\partial g}{\partial x}(x)^T \))  
\[ f_{\text{LIN}}(x) = f(\bar{x}) + \nabla f(\bar{x})^T (x - \bar{x}) + \frac{1}{2} (x - \bar{x})^T \nabla^2 \mathcal{L}(\bar{x}, \bar{\lambda}, \bar{\mu})(x - \bar{x}), \]  
\[ g_{\text{LIN}}(x) = g(\bar{x}) + \nabla g(\bar{x})^T (x - \bar{x}), \]  
and  
\[ h_{\text{LIN}}(x) = h(\bar{x}) + \nabla h(\bar{x})^T (x - \bar{x}). \]  
We know that this “exact Hessian” SQP method has locally quadratic convergence. In practical SQP methods often the Hessian matrix is inexact, e.g. approximated by BFGS, kept constant, or approximated by a Gauss-Newton Hessian, and nice convergence theory exists for these cases as well. But what happens if we also replace the constraint Jacobians \( \nabla g(\bar{x})^T \) and \( \nabla h(\bar{x})^T \) by something different than an exact derivative?  
In this note we propose to use instead of the exact linearization at a given point some other approximating functions of the form  
\[ f_{\text{APP}}(x) = f_0 + f_1^T (x - \bar{x}) + \frac{1}{2} (x - \bar{x})^T F(x - \bar{x}), \]  
\[ g_{\text{APP}}(x) = g_0 + G(x - \bar{x}), \]
and
\[ h_{\text{APP}}(x) = h_0 + H(x - \bar{x}). \]  

These functions might mix different evaluation points for the vectors and matrices. No one forces us to use a derivative at the current point (even though this has the advantage that we describe an exact tangent).

Particularly interesting is the case where we re-evaluate in each iteration \( g_0 := g(\bar{x}) \) and \( h_0 := h(\bar{x}) \) (and possibly also the gradient of the objective or Lagrangian by reverse differentiation), but keep the matrices \( F, G, H \) fixed to an old linearization or approximate them via an inexact derivative e.g. using an integrator with sensitivities with low accuracy, or some Broyden type update procedures.

This could have the following advantages, which are particularly interesting when a series of similar but parameter dependent problems needs to be solved, e.g. in homotopy methods, in multi objective optimization, real-time optimization, and nonlinear model predictive control:

- We could save time needed to compute the matrices if this is costly.
- We could re-use existing QP factorizations in the subsequent QPs using e.g. qpOASES that can exploit the fact that only vectors but not QP matrices change.
- We could impose some favourable structure (like positive definiteness of the Hessian, sparsity of Hessian or Jacobians)

The question is if the method converges to something or not. If it converges, does it converge to a local minimum or something different?

In this note we will briefly introduce into the local convergence theory of inexact SQP methods. More material can be found in [2, 3, 1].

## 2 Inexact Newton methods

Let us for simplicity start with a Newton method applied to the root finding problem
\[ F(x) = 0 \]  
which computes, when at the current linearization point \( \bar{x} \), the new iterate \( x \) as solution of the linear system
\[ F(\bar{x}) + \frac{\partial F}{\partial x}(\bar{x})(x - \bar{x}) = 0. \]  

It is well known that we can replace the Jacobian by something else, which we might call matrix \( M \), and still recover convergence to a root of \( F(x) = 0 \) if we evaluate \( F(\bar{x}) \) exactly in each iteration, i.e. if we solve
\[ F(\bar{x}) + M(x - \bar{x}) = 0. \]  

One sufficient condition for local convergence is to guarantee that during the iterations \( x_{k+1} = x_k + \Delta x_k \) a bound with Lipschitz constant \( \omega < \infty \) is guaranteed:
\[ \|M_{k+1}^{-1}(J(x_k + t \Delta x_k) - J(x_k))\Delta x_k\| \leq \omega t\|\Delta x_k\|^2 \]  

(13)

as well as a compatibility condition with compatibility constant \( \kappa < 1 \):

\[ \|M_{k+1}^{-1}(M_k - J(x_k))\Delta x_k\| \leq \kappa\|\Delta x_k\|. \]  

(14)

It is important to keep in mind that the first condition is basically satisfied if the matrix \( M \) is invertible and the function \( F \) twice differentiable, i.e., in most practical cases, and that the second condition is satisfied if \( M \) and \( J \) are not too different. Thus, if \( M \) is a decent approximation of \( J \), the inexact Newton method converges locally, with linear contraction rate \( \kappa \).

### 3 Inexact Newton Lagrange methods

When we apply the above reasoning to the equality constrained optimization problem

\[
\min_{x \in \mathbb{R}^n} f(x) \quad \text{s.t.} \quad g(x) = 0
\]  

(15)
i.e. the root finding problem

\[
\begin{bmatrix} \nabla L(x, \lambda) \\ g(x) \end{bmatrix} = 0
\]  

(16)

we immediately see that we have to evaluate in each iteration the gradient of the Lagrangian

\[ \nabla L(x, \lambda) = \nabla f(x) - \nabla g(x)\lambda \]  

(17)

and if the Hessian is approximated \( F \) and the Jacobian by \( G \) we immediately get the following linear system to solve in each iteration:

\[
\begin{bmatrix} \nabla L(\bar{x}, \bar{\lambda}) \\ g(\bar{x}) \end{bmatrix} + \begin{bmatrix} F & -G^T \\ G & 0 \end{bmatrix} \begin{bmatrix} x - \bar{x} \\ \lambda - \bar{\lambda} \end{bmatrix} = 0
\]  

(18)

or equivalently,

\[
\begin{bmatrix} \nabla L(\bar{x}, \bar{\lambda}) + G^T\bar{\lambda} \\ g(\bar{x}) \end{bmatrix} + \begin{bmatrix} F & -G^T \\ G & 0 \end{bmatrix} \begin{bmatrix} x - \bar{x} \\ \lambda \end{bmatrix} = 0
\]  

(19)

It is interesting to note that this last system in variables \( x \) and \( \lambda \) is equivalent to the KKT optimality conditions of the following QP:

\[
\min_{x \in \mathbb{R}^n} f_{\text{APP}}(x) \quad \text{s.t.} \quad g_{\text{APP}}(x) = 0
\]  

(20)

with

\[
f_{\text{APP}}(x) = f(\bar{x}) + \frac{1}{2}(x - \bar{x})^T F(x - \bar{x}) + \frac{1}{2}(x - \bar{x})^T F(x - \bar{x}), \]  

(21)

and

\[ g_{\text{APP}}(x) = g(\bar{x}) + G(x - \bar{x}). \]  

(22)
The interesting term here is the vector
\[ f_1 = \nabla f(\bar{x}) + (G^T - \nabla g(\bar{x}))\bar{\lambda} \] (23)
which we might call the “modified gradient” as it is nothing else than the gradient of the objective corrected by a term that accounts for the Jacobian inexactness. Note that for an exact Jacobian SQP method would hold \( f_1 = \nabla f(\bar{x}) \). Also note that we will never evaluate the full matrix \( \nabla g \) but compute \( f_1 \) as
\[ f_1 = \nabla L(\bar{x}, \bar{\lambda}) + G^T \bar{\lambda} \] (24)

4 Inexact SQP methods

The above reasoning for the Newton Lagrange method is easily extendable to inequality constrained problems. Here, we simply have to use the modified gradient
\[ f_1 = \nabla f(\bar{x}) + (G^T - \nabla g(\bar{x}))\bar{\lambda} + (H^T - \nabla h(\bar{x}))\bar{\mu} \] (25)
which, of course, we evaluate most efficiently as
\[ f_1 = \nabla L(\bar{x}, \bar{\lambda}, \bar{\mu}) + G^T \bar{\lambda} + H^T \bar{\mu} \] (26)
using reverse differentiation for \( \nabla L \). If with this \( f_1 \) we now solve in each SQP iteration the QP
\[
\min_{x \in \mathbb{R}^n} f_{\text{APP}}(x) \quad \text{s.t.} \quad \begin{cases} \quad g_{\text{APP}}(x) &= 0 \\ \quad h_{\text{APP}}(x) &\geq 0. \end{cases} \tag{27}
\]
with functions
\[ f_{\text{APP}}(x) = f_1^T (x - \bar{x}) + \frac{1}{2} (x - \bar{x})^T F (x - \bar{x}), \] (28)
\[ g_{\text{APP}}(x) = g(\bar{x}) + G(x - \bar{x}), \] (29)
and
\[ h_{\text{APP}}(x) = h(\bar{x}) + H(x - \bar{x}). \] (30)
and if the matrices \( F, G, H \) are sufficiently close to \( \nabla^2 L, \nabla g^T, \nabla h^T \), then the inexact SQP method converges to a local minimum of the original problem, with linear rate. This method might also be called an adjoint based inexact SQP method.

4.1 Simplification for linear inequality constraints

If the inequalities are linear, the Hessian \( \nabla^2 L \) does not depend on the inequality multipliers, and we can assume that we have computed \( H \) exactly, i.e. \( h(x) = h_{\text{APP}}(x) \). In this case the iterates in the inequality multipliers \( \mu \) do not matter in our SQP method, and the primal variables and the steps in the equality multipliers, \( x \) and \( \Delta \lambda \) can be obtained by solving the QP
\[
\min_{x \in \mathbb{R}^n} f_{\text{APP}}^{\text{simpl}}(x) \quad \text{s.t.} \quad \begin{cases} \quad g_{\text{APP}}(x) &= 0 \\ \quad h_{\text{APP}}(x) &\geq 0. \end{cases} \tag{31}
\]
with
\[ f_{\text{simpl}}^{\text{APP}}(x) = (\nabla f(\bar{x}) - \nabla g(\bar{x}, \bar{\lambda}))^T (x - \bar{x}) + \frac{1}{2} (x - \bar{x})^T F(x - \bar{x}) \] (32)

where the gradient of the simplified Lagrangian \( f(x) - \lambda^T g(x) \) can be evaluated by reverse differentiation. Note that in contrast to the previous iSQP method, we obtain the step in the multipliers only, but we have avoided the matrix vector product \( G^T \bar{\lambda} \). We can also avoid the other products by using \( \Delta x = x - \bar{x} \) as QP variable.

5 Feasibility improvement iterations

In the above inexact SQP method we have to evaluate in each iteration the gradient of the Lagrangian, which is more costly than just evaluating the constraint residual functions \( g, h \). We might ask ourselves what happens if we keep the objective approximation constant, but re-evaluate the constraint functions in each iteration?

It turns out that such a method can also converge to something, but it will not converge to an optimum. Instead, if it converges, it converges just to a feasible point as an “approximate solution”. From the point of view of many applications, this might be just as good. But how can we characterize the distance of the approximate solution \( \tilde{x}^* \) from the true one \( x^* \)? It is kept as an exercise to derive the perturbed problem of which this point is the solution.

Tipp: the problem we solve is identical to the original NLP but with a gradient perturbation which itself depends on the location of \( \tilde{x}^* \).

References

