A Superlinearly Convergent Trust Region Bundle Method

Andreas Grothey, Ken McKinnon

December 1998

MS-98-015

Supported by EPSRC research grant GR/K51587

Department of Mathematics and Statistics
University of Edinburgh, The King's Buildings, Edinburgh EH9 3JZ
Tel. (44) 131 650 5747 or 5042 E-Mail : agr or ken@maths.ed.ac.uk

A Superlinearly Convergent Trust Region Bundle Method

Andreas Grothey, Ken McKinnon

December 22, 1998

Abstract

Bundle methods for the minimization of non-smooth functions have been around for almost 20 years. Numerous variations have been proposed. But until very recently they all suffered from the drawback of only linear convergence. The aim of this paper is to show how exploiting an analogy with SQP gives rise to a superlinearly convergent bundle method. Our algorithm features a trust region philosophy and is expected to converge superlinearly even for non-convex problems.

1 Introduction

Our aim will be to present a superlinearly convergent algorithm to solve the problem

$$\min_{x \in IR^n} f(x), \tag{1}$$

where f is continuous and piecewise smooth. By this we mean that f is a composition of several smooth functions f_j each defined on a domain X_j that join along their boundaries in a continuous but non-differentiable fashion. We assume that $f_j \in C^2$ with bounded, Lipschitz continuous Hessians and that indeed at each x we can evaluate $f_j(x), \nabla f_j(x), \nabla^2 f_j(x)$ for one of the active faces. The problem is not required to be convex. The suggested method is an extension to the bundle concept introduced by Lemarechal[3], Kiwiel[4]. and will use many ideas from these methods. However by incorporating local second derivative information on f and exploiting a connection with sequential quadratic programming (SQP) we can obtain superlinear convergence. There have been several attempts in the past to include second order information into the bundle framework, most notably the recent work by Lukšan, Vlček[6] which is very similar to our approach. However all these methods use a line search, thus restricting their local search to a halfline. Also the need for convex subproblems introduced by a line search requires positive definite approximations of all Hessian matrixes. Our work is closer in spirit to the Bundle Trust Region method of Zowe, Schramm[5]. Like them we use a trust region but we can directly use the real Hessian information.

Note that although problem (1) is unconstrained it should be no problem to incorporate nonlinear (but smooth) constraints into the framework.

We start from the observation that if at a local solution x^* to (1) some functions $f_j, j \in J^*$ are active, then x^* is also a local minimum of

$$\bar{f}(x) = \max_{j \in J^*} \{f_j(x)\} \tag{2}$$

thus instead of (1) we can solve

$$\min_{x \in I\!R^n} \max_{j \in J^*} \{f_j(x)\}. \tag{3}$$

This problem can in turn be written as a constrained NLP

$$\min_{x,v} v, \qquad \text{subj. to} \quad v \geq f_j(x), j \in J^*. \tag{4}$$

This is now a smooth problem and could be solved for example by SQP which would involve solving the following subproblem at each iteration.

$$\begin{aligned} &\min_{d,v}\ v + \tfrac{1}{2} d^T [\sum_{j=1}^{m} \lambda_j^{(k-1)} \nabla^2 f_j(x_k)] d \\ &\text{subject to} \quad v \geq f_j(x_k) + \nabla f_j(x_k)^T d \end{aligned} \tag{5}$$

The $\lambda_j^{(k-1)}$ are the multiplier on the constraints in the previous QP. However doing this would require the knowledge of all f_j at any given point. In our case we only know one of the f_j at the current point, namely the one for which the maximum in (2) is attained.

Nevertheless the SQP subproblem features some desirable properties, such as superlinear (and under suitable conditions global) convergence of the resulting algorithm. Hence our aim will be to emulate the behaviour of SQP as much as possible.

The method will generates a sequence of iterates $\{x_i\}$. At each x_i we have knowledge of the function value and its derivatives of one of the f_j, say $f_{j(i)}$. Let y_j denote the most recent point of the sequence $\{x_i\}$ that led to information on f_j. Therefore at each iteration the information available is $\{f_j(y_j)\}$ together with first and second derivatives $\{\nabla f_j(y_j)\}, \{\nabla^2 f_j(y_j)\}$. We can therefore approximate the unknown information $f_j(x_k), \nabla f_j(x_k), \nabla^2 f_j(x_k)$ by a Taylor series based on y_j to give

$$\begin{aligned} f_j^{app}(x) =& f_j(y_j) + \nabla f_j(y_j)^T(x-y_j) + \frac{1}{2}(x-y_j)^T \nabla^2 f_j(y_j)(x-y_j) \\ \nabla f_j^{app}(x) =& \nabla f_j(y_j) + \nabla^2 f_j(y_j)(x-y_j) \\ \nabla^2 f_j^{app}(x) =& \nabla^2 f_j(y_j), \end{aligned} \tag{6}$$

and substitute these expressions into (5) to arrive at the conceptual subproblem

$$\begin{aligned} &\min_{d,v}\ v + \tfrac{1}{2} d^T [\sum_{j=1}^m \lambda_j^{(k-1)} \nabla^2 f_j^{app}(x_k)] d \\ &\text{subject to} \quad v \geq f_j^{app}(x_k) + \nabla f_j^{app}(x_k)^T d. \end{aligned} \tag{7}$$

There is still one problem with the use of this subproblem, namely that $f_j^{app}(x_k)$, the approximated value for $f_j(x_k)$, might overestimate the actual function value $f(x_k)$. This can lead to the premature convergence of the algorithm as can be seen in figures 1/2.

Note that this is due to the quadratic nature of the updates and would be the case even if f were convex.

In order to overcome this we will use *locality measures* β_j^k in a similar spirit as Kiwiel[4] which shift approximations of f_j down if used far from where they were generated.

We have experimented with two different choices of locality measures.

$$\beta_{j,kiw}^{(k)} = \max\{0, f_j^{app}(x_k) - f(x_k) + \gamma \|x_k - y_j\|^2\} \tag{8}$$

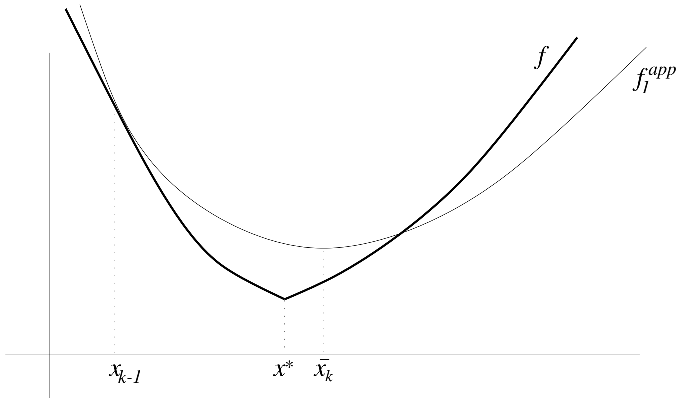

Figure 1: f_1^{app} *is the approximation of the left hand side of the function. If* $\lambda_j^{(k-1)} = 1$ *the subproblem (7) at* x_{k-1} *is equivalent to the minimization of* f_1^{app}*, hence leading to the point* x_k*.*

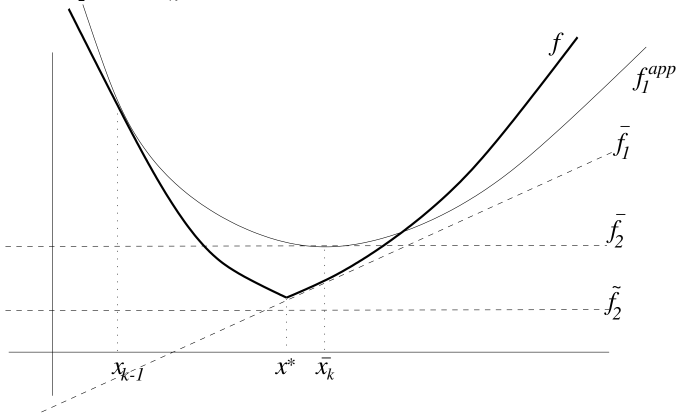

Figure 2: $\bar{f}_1$ *and* $\bar{f}_2$ *are linearizations of* f_1^{app} *and* f *around* x_k*. At* x_k *subproblem (7) minimizes the maximum of these linearizations plus a quadratic term. This minimization leads again to* x_k *since the gradient of* $\bar{f}_2$ *is zero. No progress is possible from* x_k*. Using locality measures the algorithm uses* $\tilde{f}_2$ *instead of* $\bar{f}_2$ *and will progress from* x_k*.*

as used by Kiwiel, and as an alternative

$$\beta_{j,alt}^{(k)} = \gamma \|x_k - y_j\|^2, \tag{9}$$

where in both cases $\gamma > 0$. We will also use a trust region to aid global convergence. With this our subproblem to be solved at iteration k becomes

$$\begin{aligned} \min_{d,v} \; v + \tfrac{1}{2} d^T [\sum_{j=1}^{m} \lambda_j^{(k-1)} \nabla^2 f_j^{app}(x_k)] d & \\ \text{subject to} \qquad v \geq & f_j^{app}(x_k) + \nabla f_j^{app}(x_k)^T d - \beta_j^{(k)} \\ \|d\| \leq & \rho. \end{aligned} \tag{10}$$

Note that for $\beta_{j,kiw}^{(k)}$ the approximation will always underestimate $f(x)$ at x_k. This is because

$$\begin{aligned} f_j^{app}(x_k) \;\; &+ \;\; \nabla f_j^{app}(x_k)^T d - \beta_{j,kiw}^{(k)} \\ &\leq \;\; f_j^{app}(x_k) - f_j^{app}(x_k) + f(x_k) - \gamma \|x_k - y_j\|^2 \\ &\leq \;\; f(x_k). \end{aligned}$$

In the second case observe that $\exists L > 0 : \|\nabla^2 f_j(x) - \nabla^2 f_j(y)\| \leq L \|x - y\|$ and therefore

$$\|f_j(y) - f_j(x) - \nabla f_j(x)(y-x) - \frac{1}{2}(y-x)\nabla^2 f_j(x)(y-x)\| \leq \frac{L}{12}\|y-x\|^3.$$

It follows that if $\gamma > L\|x_k - y_j\|/12$, then

$$\begin{aligned} f_j^{app}(x_k) \;\; &+ \;\; \nabla f_j^{app}(x_k)^T d - \beta_j^{(k)} \\ &= \;\; f_j^{app}(x_k) - \gamma \|x_k - y_j\|^2 \\ &\leq \;\; f_j(x_k) + \frac{L}{12}\|x_k - y_j\|^3 - \gamma \|x_k - y_j\|^2 \\ &\leq \;\; f_j(x_k) \leq f(x_k). \end{aligned}$$

The reason for considering this alternative is that in (7) the approximation $f_j^{app}(x_k) + \nabla f_j^{app}(x_k)^T d$ approximates $f_j(x_k + d)$ up to $O(\|x_k - y_j\|^2)$. This property will be important for the superlinear convergence of the algorithm and is preserved by the use of $\beta_j^{(k)}$. However the condition $\gamma > L\|x_k - y_j\|/12$ is necessary and we might have convergence to a non-optimal point if it does not hold. But if the sequence $\{x_i\}$ and hence $\{\|x_k - y_j\|\}$ remain bounded this condition can be guaranteed to hold for γ big enough.

In the following chapter we will formally state the proposed algorithm. The last chapter will present numerical results on a range of both academic test examples and real-life problems.

2 The Algorithm

The solution to subproblem (10) suggests a step d_k to the new point $y_{j(k+1)} = x_k + d_k$. In what follows we will use the index $_+$ to denote both $_{k+1}$ and $_{j(k+1)}$. The constraints in (10) can be interpreted as defining a "cutting-plane" model for the v-term in the objective function

$$f_{cp}^{(k)}(x) = \max_j \{f_j^{app}(x_k) - \beta_j(k) + \nabla f_j^{app}(x_k)^T (x - x_k)\} \tag{11}$$

which the subproblem is minimizing. A traditional trust region algorithm will accept a step as long as there is a sufficient similarity between the model and the objective function. The usual test for accepting steps (in bundle method terms a SERIOUS STEP) is that the *actual reduction* is at least a fixed percentage of the *predicted reduction*. In terms of our algorithm

$$f(y_+) \leq f(x_k) + m_1(v_k - f(x_k)), \quad 0 < m_1 < 1. \tag{12}$$

SQP will work with this kind of strategy since in SQP the model is guaranteed to be a "good" approximation of f at least in a small neighbourhood of x_k. This however is not the case with subproblem (10) (since cuts might be gathered far from the current point) and our algorithm has to take care of this.

Bundle methods overcome this by the introduction of NULL STEPS. If the actual decrease is not sufficient for (12), instead of reducing the trust region the information obtained at $y_{k+1} = x_k + d_k$ might be added to the model. In this case the constraint $v \geq f_+^{app}(x_k) + \nabla f_+^{app}(x_k)^T d - \beta_+^{(k)}$ is added to the subproblem and it is resolved.

However adding this constraint only makes sense if the model is changed sufficiently by it. This is the case if the condition

$$f_+^{app}(x_k) + \nabla f_+^{app}(x_k)^T d_k - \beta_+^{(k)} \geq v_k + m_2(f(x_k) - v_k) \tag{13}$$

holds for fixed $m_2 > 0$ such that $1 - m_1 - m_2 > 0$. In this case the solution to the unmodified subproblem (d_k, v_k) cannot be obtained anymore, hopefully leading to the selection of a different, more successful d_k after the resolve. Should the test (13) fail, the old rejected step is still optimal in the new subproblem (if $m_2 = 0$). In this case we have no other choice but to reduce the trust region and solve the old model again.

It is important to verify that using this strategy ρ cannot decreased forever: eventually either a NULL STEP or a SERIOUS STEP will have to be accepted. We will show that this is the case at the end of this section.

We present the suggested algorithm:

0. **Initialization**
   Choose $\rho > 0$, $\gamma > 0$, $\epsilon > 0$.
   Choose parameters $0 < m_1, m_2$ with $1 - m_1 - m_2 > 0$.
   Choose starting point x_1.
   Evaluate $f(x_1), \nabla f(x_1), \nabla^2 f(x_1)$
   and set $f^{app}_{j(1)}(x_1) = f(x_1), \nabla f^{app}_{j(1)}(x_1) = \nabla f(x_1), \nabla^2 f^{app}_{j(1)}(x_1) = \nabla^2 f(x_1)$.
   Set $\lambda_1^{(0)} = 0$, $k = 1$, $J = \{j(1)\}$.

1. **Solve subproblem:** solve subproblem (10) for (d_k, v_k) and multipliers on the constraints $\lambda_j^{(k)}$.

2. **Convergence test:** if $\|d_k\| \le \epsilon$ then stop, convergence.

3. **Evaluate objective function:** Set $y_+ = x_k + d_k$.
   Evaluate $f(y_+), \nabla f(y_+), \nabla^2 f(y_+)$. Set $f^{app}_+(x_k), \nabla f^{app}_+(x_k), \nabla^2 f_+(x_k)$ according to (6).

4. **Trust Region strategy**

   if $f(y_+) < f(x_k) + m_1(v - f(x_k))$ then SERIOUS STEP
      set $r := \frac{f(y_+) - f(x_k)}{v - f(x_k)}$.
      if $r > 0.75$ and $\|d_k\| = \rho$ then $\rho := 2\rho$.
      if $r < 0.25$ then $\rho := \rho/4$.
   else
      if $f^{app}_+(x_k) + \nabla f^{app}_+(x_k)^T d_k - \beta_+^{(k)} > v_k + m_2(f(x_k) - v_k)$ then
         NULL STEP
      else
         $\rho := \rho/4$ and back to step 1.

end if
end if

5. **Update cuts:**

   add $j(k+1)$ to J.
   if (Serious Step)
   $x_{k+1} = x_k + d_k$
   evaluate $f_j^{app}(x_{k+1}), \nabla f_j^{app}(x_{k+1}), \nabla^2 f_j^{app}(x_{k+1})$ for $j \in J$ according to (6).
   if (Null Step)
   $x_{k+1} = x_k$

6. **Delete obsolete cuts:**
   delete all constraints from (10) that have $\lambda_j^{(k)} = 0$.

7. $k := k+1$ and return to step 1.

REMARKS

- In the nondegenerate case there can be a maximum of $n+1$ constraints in (10) with nonzero multipliers. Even in the degenerate case this can be assured by the use of an appropriate QP solver. Hence our deletion strategy ensures that the subproblem has never more than $n+2$ constraints.

- In many problems (e.g those arising from decomposition) it is known to which f_j the information returned by the function evaluation corresponds. For the sake of generality the above algorithm does not rely on this knowledge. However if available it can be used in two ways:

  - reject steps if y_+ is on the same face as x_k but Serious Step condition fails.
  - only keep the most recent cut corresponding to each face f_j.

2.1 Finiteness of inner iteration

To justify what we are doing, we will show that the inner iteration (steps 1-4) of the above algorithm is finite. That is after a finite number of steps either a NULL STEP or a SERIOUS STEP is generated (or the convergence criterion is satisfied).

For this we need the following

Lemma 1 *Let* $(v(\rho), d(\rho))$ *be the solution to*

$$\min_{v,d} v + \tfrac{1}{2} d^T H d$$
$$\text{subject to } v \geq f_{cp}^{(k)}(x_k + d), \|d\| \leq \rho$$

then for fixed ρ_0 *and* $\rho < \rho_0$ *small enough*

$$f(x_k) - v(\rho) \geq \frac{\rho}{\rho_0}(f(x_k) - v(\rho_0)).$$

Proof:

$$\begin{aligned}
v(\rho) &= \arg\min_v \min_d \; v + \tfrac{1}{2} d^T H d \\
&\qquad \text{s.t. } v \geq f_{cp}^{(k)}(x_k + d), \|d\| \leq \rho \\
&\leq \arg\min_v \min_{t,d} \; v + \tfrac{1}{2} d^T H d \\
&\qquad \text{s.t. } v \geq f_{cp}^{(k)}(x_k + d), d = t d_0, 0 \leq t \leq \rho/\rho_0 \\
&= \arg\min_v \min_t \; v + \tfrac{1}{2} t^2 d_0^T H d_0 \\
&\qquad \text{s.t. } v \geq f_{cp}^{(k)}(x_k + t d_0), 0 \leq t \leq \rho/\rho_0 \\
&\leq \arg\min_v \min_t \; v + \tfrac{1}{2} t^2 d_0^T H d_0 \\
&\qquad \text{s.t. } v \geq f(x_k) + t(f_{cp}^{(k)}(x_k + d_0) - f(x_k)), 0 \leq t \leq \rho/\rho_0 \\
&= \arg\min_v \min_t \; v + \tfrac{1}{2} t^2 d_0^T H d_0 \\
&\qquad \text{s.t. } v \geq f(x_k) + t(v(\rho_0) - f(x_k)), 0 \leq t \leq \rho/\rho_0
\end{aligned}$$

where the first inequality is due to restricting the feasible choices for d and the second due to convexity of $f_{cp}^{(k)}$. The last problem is equivalent to

$$\min_{0 \leq t \leq \rho/\rho_0} f(x_k) + t(v(\rho_0) - f(x_k)) + \tfrac{1}{2} t^2 d_0^T H d_0$$

with a solution (since $v(\rho_0) - f(x_k) \leq 0$)

$$t_{min} = \begin{cases} \rho/\rho_0 & \text{if } d_0^T H d_0 \leq 0, \\ \min\{\frac{\rho}{\rho_0}, 2\frac{f(x_k)-v(\rho_0)}{d_0^T H d_0}\} & \text{if } d_0^T H d_0 > 0. \end{cases}$$

So for ρ small enough

$$t_{min} = \frac{\rho}{\rho_0}$$

and therefore

$$\begin{aligned} v(\rho) &\leq f(x_k) + t_{min}(v(\rho_0) - f(x_k)) \\ &= f(x_k) + \rho/\rho_0(v(\rho_0) - f(x_k)). \end{aligned}$$

□

Lemma 2 *For both choices of $\beta_+^{(k)}$ there exists an $C > 0$ so that if in the algorithm the* Serious Step *condition (12) is not satisfied we have*

$$f_+^{app}(x_k) + \nabla f_j^{app}(x_k)^T d_k - \beta_+^{(k)} \geq f(x_k) + m_1(v_k - f(x_k)) - C\|x_k - y_+\|^2.$$

Proof:

Since the Serious Step was rejected

$$f(y_+) \geq f(x_k) + m_1(v_k - f(x_k)).$$

case 1) $\beta_+^{(k)} = \gamma\|x_k - y_j\|^2$

In this case

$$\begin{aligned} \Rightarrow f_+^{app}(x_k) \; &+ \; \nabla f_+^{app}(x_k)^T d_k - \beta_+^{(k)} \\ &= \; f(y_+) - \tfrac{1}{2} d_k^T \nabla^2 f(y_+) d_k - \gamma d_k^T d_k \\ &\geq \; f(x_k) + m_1(v_k - f(x_k)) - d_k^T[\gamma I + \tfrac{1}{2}\nabla^2 f(y_+)]d_k. \end{aligned}$$

case 2a) $\beta_+^{(k)} = 0$ as above for $\gamma = 0$.

case 2b) $\beta_+^{(k)} = f_+^{app}(x_k) - f(x_k) + \gamma\|x_k - y_j\|^2$

Since $f_+ \in C^2$ we know

$$\|f_+(x_k) - f(y_+) - \nabla f(y_+)^T d_k\| \leq M\|x_k - y_+\|^2$$

$$\begin{aligned}
\Rightarrow -\nabla f(y_+)^T d_k &\geq f(y_+) - f_+(x_k) - M\|d_k\|^2 \\
&\geq f(x_k) + m_1(v_k - f(x_k)) - f_+(x_k) - M\|d_k\|^2 \\
&\geq m_1(v_k - f(x_k)) - M\|d_k\|^2
\end{aligned}$$

and then we have

$$\begin{aligned}
f_+^{app}(x_k) &+ \nabla f_+^{app}(x_k)^T d_k - \beta_+^{(k)} \\
&= f(x_k) - \gamma\|d_k\|^2 + (\nabla f_+(y_+) + \nabla^2 f_+(y_+) d_k)^T(-d_k) \\
&= f(x_k) - \nabla f(y_+)^T d_k - d_k^T[\gamma I + \nabla^2 f(y_+)]d_k \\
&\geq f(x_k) + m_1(v_k - f(x_k)) - d_k^T[(\gamma + M)I + \nabla^2 f(y_+)]d_k
\end{aligned}$$

and the lemma is proved for $C > \gamma + M + \|\nabla^2 f\|$. □

Combining the two lemmata we obtain

Theorem 1 *The inner iteration of the algorithm is finite*

Proof:

Assume that throughout the inner iteration the convergence criterion is not satisfied and no SERIOUS STEPS are accepted. Keeping in mind that the solution to the subproblem depends on ρ we will write $(v(\rho), d(\rho))$ for (v_k, d_k), and $\|x_k - y_+\| \leq \rho$. Thus

$$\begin{aligned}
&f_+^{app}(x_k) + \nabla f_+^{app}(x_k)^T d(\rho) - \beta_+^{(k)}(\rho) \\
&\geq f(x_k) + m_1(v(\rho) - f(x_k)) - C\|x_k - y_+\|^2 \\
&= v(\rho) + m_2(f(x_k) - v(\rho)) + (1 - m_1 - m_2)(f(x_k) - v(\rho)) - C\|x_k - y_+\|^2 \\
&\geq v(\rho) + m_2(f(x_k) - v(\rho)) + (1 - m_1 - m_2)(f(x_k) - v(\rho_0))\rho/\rho_0 - C\rho^2 \\
&\geq v(\rho) + m_2(f(x_k) - v(\rho))
\end{aligned}$$

for ρ small enough. Hence a NULL STEP has to be accepted for ρ small enough which proves the theorem. □

3 Numerical Results

We have tested our algorithm on a range of test-problems. In the first part of this chapter we will concentrate on academic test-problems, whereas the second part presents results on real life problems that arise from the decomposition of a Utility Optimization problem.

3.1 Academic Examples

The problems we used for this test are

- the chained Rosenbrock Problem

$$f(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (1 - x_i^2)] \tag{14}$$

  Rosenbrock's problem is of course smooth and we would hope that our method can duplicate the results of SQP in this case.

- n1actfs (2 examples: convex and nonconvex)

$$f(x) = \max\{g(-\sum x_i), g(x_i) : i = 1, \dots, n\} \tag{15}$$

  where $g(y)$ is one of

$$g(y) = \begin{cases} e^{|y|} - 1, & \text{convex} \\ \ln(|y| + 1), & \text{nonconvex.} \end{cases}$$

  The maximal number of active functions in (2) can be $n + 1$ where n is the dimension of the problem. This problem is designed so that indeed $n + 1$ functions are active at the solution.

- non-smooth chained Rosenbrock Problem

$$f(x) = \sum_{i=1}^{n-1} [g(x_{i+1} - x_i^2) + (1 - x_i^2)] \quad \text{where } g(y) = (|y|) + 1)^2 - 1 \tag{16}$$

n	chained Rosenbrock	non-smooth chained Rosenbrock	n1actfs convex	n1actfs non-convex
γ	1	4	1	1
2	17	25	17	23
5	26	35	32	41
10	36	84	57	84
20	65	130	94	141
30	92	183	244	207
50	150	309	270	338

Table 1: results for β_{alt} as in (9).

n	chained Rosenbrock	non-smooth chained Rosenbrock	n1actfs convex	n1actfs non-convex
γ	1	2	1	1
2	17	13	19	20
5	26	24	35	38
10	36	44	59	71
20	65	79	109	132
30	92	112	159	201
50	150	159	278	351

Table 2: results for β_{kiw} as in (8).

which is identical to the smooth version of the problem with the difference that the $100y^2$ term which is responsible for the "steep banana shaped valley" appearance of the function is replaced by the "non-smooth valley" term $g(y)$. At the solution n of $n+1$ possible functions are active.

We have tried both choices of $\beta_j^{(k)}$ for each function with a convergence tolerance of $\epsilon = 10^{-12}$. The results of our experiments are summarized in tables 1 and 2.

Superlinear convergence to the final solution was observed in each of these cases.

Using $\beta^{(k)}_{j,alt}$ the number of iterations depended more heavily on the value of γ and may fail for γ to small. The results stated here are for an "average" successful choice of γ

Kiwiel's choice $\beta^{(k)}_{j,kiw}$ on the other hands leads to convergence for all values of γ and is on average faster than our choice. However in a few cases convergence was of a slow linear speed for a long time in the solution process, before superlinear convergence began. But this was typically at a much later stage than with $\beta^{(k)}_{j,alt}$. Again results are for an average choice of γ although the variation is much less in this case.

3.2 Decomposition

We have tried our method on a real life example arising from the decomposition of a Utility Optimization problem. We used a model of a 2-turbine power generation cycle (Figure 3). A problem of this kind can be written as

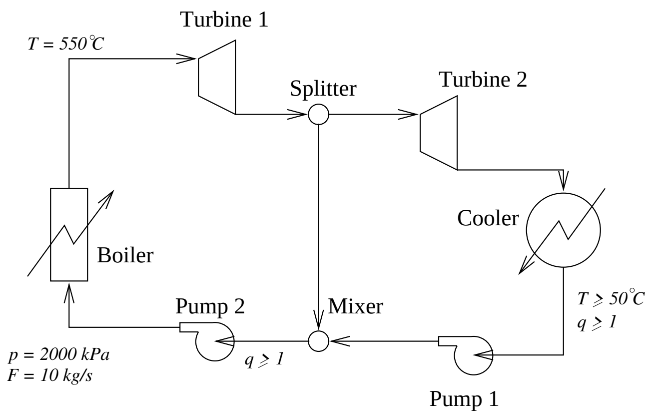

Figure 3: 2-turbine power generation cycle

$$\min_{y,y_i^{l,o},x_i} \sum_{i=1}^{m} f_i(x_i, y_i^{in}, y_i^{l,o}), \quad \text{s.t.} \quad \begin{aligned} c_i(x_i, y_i^{in}, y_i^{l,o}) &= 0, \\ y_i^{l,o} - y_i^{out} &= 0, \\ \underline{x_i} \leq x_i &\leq \overline{x_i} \\ \underline{y} \leq y &\leq \overline{y} \end{aligned} \tag{17}$$

where f_i, c_i are smooth nonlinear, non-convex functions. We decomposed (17) using an augmented Lagrangian on the $y_i^{l,o} - y_i^{out} = 0$ constraint and decomposing in y_i, λ_i to obtain

$$\max_{\lambda} v(\lambda) \tag{18}$$

$$v(\lambda) = \min_{y} \sum_{i=1}^{m} v_i(y_i^{in}, y_i^{out}, \lambda_i) \tag{19}$$

$$\begin{aligned} v_i(y_i^{in}, y_i^{out}, \lambda_i) = \min_{y_i^{l,o}, x_i} & f_i(x_i, y_i^{in}, y_i^{l,o}) + \lambda_i^T (y_i^{l,o} - y_i^{out}) \\ & + \theta (y_i^{l,o} - y_i^{out})^T (y_i^{l,o} - y_i^{out}) \\ \text{subject to } & c_i(x_i, y_i^{in}, y_i^{l,o}) = 0 \\ & \underline{x_i} \leq x_i \leq \overline{x_i}. \end{aligned} \tag{20}$$

The functions $v_i(y_i^{in}, y_i^{out}, \lambda_i)$ are non-smooth and satisfy the condition for the application of our method. We have therefore solved (19) by the suggested algorithm with our choice for $\beta_j^{(k)}$. Note that problems (18) and (20) are smooth and were therefore solved by an implementation of SQP (see Fletcher,Leyffer [2] for details). In our example the un-decomposed problem has 261 variables and 167 constraints. Taking the dependent variables p, h, F of each stream as complicating variables y_i yields 27 y-variables and decomposes the problem into 8 subproblems by tearing every connecting stream. Thus the medium level problem (19) which is solved by the bundle method has 27 variables. In the solution process (19) was called five times with the following details

call no	iter	cuts active
1	38	1
2	126	3
3	54	2
4	33	4
5	20	3

Again superlinear convergence was observed. Furthermore the algorithm was hot started from the solution of the respective last problem which explains the decrease in iterations in the final two calls. However no function approximations were kept between bundle calls.

References

[1] R. Fletcher, *Practical Methods of Optimization, 2^{nd} edition*, John Wiley, 1987.

[2] R. Fletcher, S. Leyffer, *Nonlinear Programming without a penalty function*, Numerical Analysis Report NA/171, Department of Mathematics, University of Dundee, September 1997.

[3] C. Lemaréchal, *Extensions diverses des méthodes de gradient et application*, Thèse d'Etat, Paris, 1980.

[4] K.C. Kiwiel, *Methods of Descent for Nondifferentiable Optimization*, Springer-Verlag, Berlin, New York, 1985.

[5] J. Zowe, H. Schramm, *A Version of the Bundle Idea for Minimizing a Nonsmooth Function: Conceptual Idea, Convergence Analysis, numerical Results*, SIAM Optimization, 2/1 (1992), pp. 121-152

[6] L. Lukšan, J. Vlček, *A bundle Newton method for nonsmooth unconstrained minimization*, Math. Prog. 83 (1998) pp. 373-391.