On the implementation of augmented Lagrangian algorithms

José Mario Martínez*

March 2, 1998

Abstract

A family of augmented Lagrangian algorithms for nonlinear programming is described. The algorithms are implemented using trust-region box constraint optimization software for solving the subproblems. In particular, an implementation of a modified exponential multiplier method is introduced. Numerical experiments are presented.

*Institute of Mathematics, University of Campinas, CP 6065, 13081-970 Campinas SP, Brazil. This work was supported by PRONEX, FAPESP (grant 90-3724-6), FINEP, CNPq, FAEP-UNICAMP.

1 Introduction

We are concerned with the nonlinear programming problem

$$\text{Minimize } f(x)$$

$$\text{subject to} \quad h(x) = 0, \;\; g(x) \leq 0, \;\; x \in \Omega,$$

where $f : I\!R^n \to I\!R$, $h : I\!R^n \to I\!R^m$, $g : I\!R^n \to I\!R^p$ are differentiable and Ω is a simple closed and convex set. In general, $\Omega = \{x \in I\!R^n \mid \ell \leq x \leq u\}$. Although all the arguments in this work apply to the general case, we will restrict ourselves, for the sake of simplicity, to the case where no equality constraints are present:

$$\text{Minimize } f(x) \tag{1}$$

$$\text{subject to} \quad g(x) \leq 0, \;\; x \in \Omega. \tag{2}$$

The main step of an augmented Lagrangian method for solving (1)-(2) is

$$\text{Minimize (approximately) } \; L(x, \rho, \mu) \; \text{ subject to } \; x \in \Omega, \tag{3}$$

where $\rho \in I\!R_{++}$ is a penalty parameter associated to the constraints $g(x) \leq 0$ and $\mu \in I\!R^p_+$ is a vector of Lagrange multiplier estimates.

(Throughout this work, $I\!R_+ = \{t \in I\!R \mid t \geq 0\}$, $I\!R_{++} = \{t \in I\!R \mid t > 0\}$ and $[v]_i$ is the $i-$th component of the vector v.)

The method can also be formulated with p different penalty parameters, one for each component of $g(x)$. The description for this situation is a straightforward variation of the one that we are going to present here. However, it does not seem to have numerical advantages in practical cases.

The objective function of (3) will be called an augmented Lagrangian if the following properties take place:

P1. For all fixed $\mu \in I\!R^p_+$ (except, perhaps, $\mu = 0$) the method defined by repeated applications of (3) with ρ going to ∞ is a *penalty method* (see [1, 8, 9] among others). This implies that, assuming that the feasible region of (1–2) is nonempty and that x_k is an exact global minimizer of $L(x, \rho_k, \mu)$, every limit point of $\{x_k\}$ is a global solution of (1–2).

P2. If x_* is a regular stationary point of (1–2) and $\mu_* \in I\!R^p_+$ is the vector of Lagrange multipliers then, for all fixed $\rho \in I\!R_{++}$, x_* is a stationary point of

$$\text{Minimize } L(x, \rho, \mu_*) \; \text{ subject to } \; x \in \Omega. \tag{4}$$

An augmented Lagrangian algorithm consists of repeated applications of (3) followed by the updating of the penalty parameter and the Lagrange multiplier estimates. Generally speaking, the penalty parameters are increased between different iterations if the progress measured in terms of gains of feasibility and complementarity ($\sum_{i=1}^p [\mu]_i [g(x)]_i$ must be zero at a solution) is not satisfactory.

The form of *separable* augmented Lagrangian functions is

$$L(x, \rho, \mu) = f(x) + \sum_{i=1}^{p} R(\rho, \mu_i, g_i(x)). \tag{5}$$

A general way in which suitable augmented Lagrangian schemes can be obtained (see [1]) is defining

$$R(\rho, \mu_i, g_i(x)) = \frac{1}{\rho}\theta(\rho g_i(x), \mu_i) \tag{6}$$

where $\theta : I\!R \times I\!R_+ \to I\!R$ is a function with the following properties:

$$\lim_{\rho\to\infty} \frac{1}{\rho}\,\theta(\rho z, \mu) = \infty \quad \text{for all} \quad z > 0, \mu > 0 \tag{7}$$

$$\lim_{\rho\to\infty} \frac{1}{\rho}\,\theta(\rho z, \mu) = 0 \quad \text{for all} \quad z < 0, \mu > 0 \tag{8}$$

$$\frac{\partial}{\partial z}\,\theta(0, \mu) = \mu \quad \text{for all} \quad \mu > 0. \tag{9}$$

$$\frac{\partial}{\partial z}\,\theta(z, \mu) \geq 0 \quad \text{for all} \quad z \in I\!R, \mu > 0. \tag{10}$$

Many variations of this general scheme were introduced and exploited in the literature. In particular, see [3]. Convergence properties of augmented Lagrangian algorithms for convex problems were recently surveyed in [16]. Due to (7–8), property **P1** holds for augmented Lagrangians defined by (6). On the other hand, the identity (9) guarantees that property **P2** takes place.

The most classical (Powell-Hestenes-Rockafellar) augmented Lagrangian method corresponds to

$$\theta_{PHR}(z, \mu) = \frac{(z+\mu)^2}{2} \quad \text{for } z \geq -\mu\ ,\ 0 \ \text{ otherwise.}$$

The main drawback of the PHR augmented Lagrangian method is that second derivatives of θ_{PHR} are discontinuous, so that methods for solving (3) based on quadratic approximations of the objective function tend to be inefficient.

The exponential-multiplier form of the augmented Lagrangian (see [1]) overcomes this difficulty defining

$$\theta(z, \mu) = \mu\, e^z \quad \text{for all} \quad \mu > 0, z \in I\!R.$$

In this paper we describe an implementation of the augmented Lagrangian method based on (7–10), we propose a modification of the exponential Lagrangian method that enhances its efficiency, we show numerical experiments and we suggest the main lines for future research.

2 Updating the penalty parameters and the multipliers

Assume that problem (3) has been solved for some $\mu_k \in I\!R^p_+, \rho_k \in I\!R_{++}$ and call x_k the approximate solution obtained. By Property **P2**, if we fixed μ_k and let $\rho_k \to \infty$, the sequence of solutions $\{x_k\}$ would tend to a minimizer of (1-2). This means that, when we solve problem (3) we expect some progress in relation to the previous approximation, both in terms of feasibility and optimality. If this progress is not satisfactory, the penalty parameter should be increased, in general by multiplication by a fixed positive factor γ.

Now, after the resolution of (3) we have enough information to define a new approximation for the vector of Lagrange multipliers. In fact, at a regular solution x_* of (1–2) we have that

$$\langle \nabla f(x_*) + \sum_{i=1}^{p} [\mu_*]_i \nabla g_i(x_*), x - x_* \rangle \geq 0 \quad \text{for all} \quad x \in \Omega, \tag{11}$$

while, at a solution of (3) we have that, approximately,

$$\langle \nabla f(x_k) + \sum_{i=1}^{p} \theta'(g_i(x^k), [\mu_k]_i) \nabla g_i(x_k), x - x_k \rangle \geq 0 \quad \text{for all} \quad x \in \Omega. \tag{12}$$

Comparison between (11) and (12) suggests that a suitable new estimate for the Lagrange multiplier $[\mu_*]_i$ is

$$[\mu_{k+1}]_i = \theta'(g_i(x^k), [\mu_k]_i). \tag{13}$$

(So, according to (10), $\mu_{k+1} \geq 0$.) Therefore, by (12), we have that

$$\langle \nabla f(x_k) + \sum_{i=1}^{p} [\mu_{k+1}]_i \nabla g_i(x_k), x - x_k \rangle \geq 0 \quad \text{for all} \quad x \in \Omega. \tag{14}$$

is satisfied, approximately, after each outer iteration of an augmented Lagrangian method.

Assume, for a moment, that (14) holds up to a user-given small precision $\varepsilon > 0$. If, in addition, we have that

$$g_i(x_k) \leq \varepsilon \quad \text{for all} \quad i = 1, \ldots, p \tag{15}$$

and

$$[\mu_{k+1}]_i \leq \varepsilon \quad \text{whenever} \quad g_i(x_k) < -\varepsilon \tag{16}$$

we say that x_{k+1} is an approximate solution of the original problem. (In fact, it is an approximate stationary point of (1–2).)

In order that, eventually, approximate stationary points can be reached, we require that the precision $\varepsilon_k > 0$ that define (3) be such that $\varepsilon_k = \varepsilon$ after a finite number of iterations.

Taking into account to the observations above, a practical augmented Lagrangian algorithm can be defined by:

Augmented Lagrangian algorithm

Step 1 *Initialization*

Choose $\tau \in (0, 1), \gamma > 1$, $\rho_1 > 0$, $[\mu_1]_i > 0$ for all $i = 1, \ldots, p$, $\varepsilon_1 > 0$, $\sigma_0 = \infty$.

Step 2 *Solve the subproblem*

Solve (3) up to precision ε_k.

Step 3 *Update Lagrange multipliers*

Compute μ_{k+1} according to (13).

Step 4 *Stopping criterion*

If $\varepsilon_k = \varepsilon$ and, in addition, (15) and (16) hold, declare "convergence" and terminate the execution of the algorithm.

Step 5 *Update penalty parameter and precision*

Define ε_{k+1} and

$$\sigma_k = \max \{| \min \{[\mu_k]_i, -g_i(x_k)\}|, \; i = 1, \ldots, p\}.$$

If $\sigma_k \leq \tau \sigma_{k-1}$, define $\rho_{k+1} = \rho_k$. Else, define $\rho_{k+1} = \gamma \rho_k$.
Step 6 *Increase iteration number*
Replace k by $k+1$ and go to **Step 2.**

In our experiments we used $\tau = 0.1, \gamma = 10$, $\rho_1 = 10$, $[\mu_1]_i = 1$ for the exponential Lagrangian method and $[\mu_1]_i = 0$ for PHR.

3 Solving the subproblems

Assume that Ω is an n-dimensional box, given by

$$\Omega = \{x \in I\!R^n \;\; | \;\; \ell \leq x \leq u\}.$$

So, (3) consists on finding an approximate solution of

$$\text{Minimize}_x \;\; L(x, \rho, \mu) \text{ subject to } \ell \leq x \leq u. \tag{17}$$

Augmented Lagrangian algorithms with approximate solutions of the subproblems were analyzed in [4, 7, 14, 15].

Subproblem (17) is solved, at each outer iteration, using the method introduced in [11], called BOX from here on. This is an iterative method which, at each iteration, approximates the objective function by a quadratic and minimizes this quadratic model in the box determined by the natural constraints $\ell \leq x \leq u$ and an auxiliary box that represents the region where the quadratic approximation is reliable (trust region). If the objective function is sufficiently reduced at the (approximate) minimizer of the quadratic, the corresponding trial point is accepted as new iterate. Otherwise, the trust region is reduced. The main algorithmic difference between BOX and the method LANCELOT (used in [4]) is that in [11] the quadratic is explored on the whole intersection of the natural box and the trust region, while in [4] only the face determined by an "approximate Cauchy point" is explored. A comparison between these two methods for box-constrained minimization can be found in [6].

The augmented Lagrangian algorithm is designed in order to cope large-scale problems. For this reason, no factorization of matrices are used at any stage. The quadratic solver used to deal with the subproblems of the box-constraint algorithm (called QUACAN from now on) visits the different faces of its domain using conjugate gradients on the interior of each face and "chopped gradients" as search directions to leave the faces. See [10], [11] and [2] for a description of the 1998 implementation of QUACAN. At each iteration of this quadratic solver, a matrix-vector product of the Hessian approximation and a vector is needed. Since Hessian approximations are usually cumbersome to compute, we use the "Truncated Newton" approach, so that each *Hessian* $\times$ *vector* product is replaced by an incremental quotient of ∇L along the direction given by the vector.

The augmented Lagrangian subroutine has many parameters that influentiate its practical performance. In this study we adjusted the most sensitive parameters using a typical problem with 103 variables and 78 nonlinear constraints, called “the Icosahedron problem”. (This is problem 1 (3, 12) described below, with the inequality constraints replaced by equality constraints by means of the introduction of slack variables.) Below we comment the decision taken on the main sensitive parameters based on this problem.

3.1 Termination criteria for the box-constraint solver

Each outer iteration finishes when some stopping criterion for the algorithm that solves (17), is fulfilled. We consider that the box-constraint algorithm BOX converges when

$$\|g_P(x)\|_2 \leq \varepsilon_k,$$

where $g_P(x)$ is the “continuous projected gradient” of the objective function of (17) at the point x. This vector is defined as the difference between the projection of $x - \nabla L(x, \rho, \mu)$ on the box and the point x. The tolerance ε_k may change at each outer iteration. In fact, we tested (with the Icosahedron problem) a strategy that defines dynamically ε_k depending of the degree of feasibility of the current iterate against a constant choice $\varepsilon = 10^{-5}$. Although not conclusive, the results for constant ε in the typical problem were better. So, we adopted this choice in the experiments. The box-constraint code admits other stopping criteria. For example, the execution also stops when the radius of the trust region becomes too small (less than 10^{-8} in our experiments) or when the number of iterations exceed a user-given value (300 in our experiments). Moreover, execution can stop if the progress between different iterations is not good during some consecutive steps. However, best results were obtained inhibiting this alternative stopping criterion.

3.2 Parameters for the Quadratic Solver

The algorithm QUACAN, which minimizes a (not necessarily convex) quadratic with bounds on the variables, plays a crucial role in the overall behavior of the augmented Lagrangian method. Therefore, its main parameters must be carefully chosen. A very important one is the parameter used to declare convergence of the algorithm. If the projected gradient of the quadratic is null, the corresponding point is stationary. According to this, convergence is declared if the norm of this projected gradient is less than a fraction of the corresponding norm at the initial point. Here we use “non-continuous projected gradients”, in which the projections are not computed on the feasible box but on the affine subspace defined by the active constraints. After testing the fractions 1/10, 1/100 and 1/100000 on the Icosahedron Problem, we observed that the first was the best one, so it was the one employed in the numerical experiments. The number of iterations allowed to the quadratic solver is also important because, sometimes, a lot effort is invested in solving subproblems without a close relation to the original problem. We found that 100 is a suitable value for “maximum of iterations” in this case. Other non-convergence stopping criteria were inhibited in the the resolution of the quadratic subproblem.

The radius of the trust region determines the size of the domain of the auxiliary box used in QUACAN. The nonlinear programming algorithm is sensitive to the choice of δ, the first trust region radius. In the experiments presented in this paper we used $\delta = 10$.

A very important parameter of the quadratic solver is $\eta \in (0,1)$. According to this parameter, it is decided whether the next iterate must belong to the same face as the current one, or not. Roughly speaking, if η is small the algorithm tends to leave the current face when a mild decrease of the quadratic is detected. On the other hand, if $\eta \approx 1$, the algorithm only abandons the current face when the current point is close to a stationary point of the quadratic on that face. A rather surprising result was that the conservative value $\eta = 0.95$ was better than smaller values of η for the Icosahedron Problem.

When the quadratic solver hits the boundary of its feasible region a extrapolation step can be tried, according to the value of an extrapolation parameter $\kappa \geq 1$. If κ is large new points will be tried at which the number of active constraints can be considerably increased. On the other hand, if $\kappa = 1$, no extrapolation is intended. Here, we finished up deciding that $\kappa = 10$ is suitable for the Icosahedron problem.

4 Modification of the exponential Lagrangian method

One of the computational difficulties associated to the exponential penalty function is related to the rapid growth of this function, which can cause overflow and numerical instability.

Sometimes stopping by overflow can be avoided without further consequences if the compiler has the capability of replacing the undesirable quantity by the largest possible machine number. This is the case of the problem corresponding to Table 4 below. In these cases the quantities associated to overflow correspond to trial points that are going to be rejected and the decision on the trust region size taken by BOX is independent of the magnitude of the objective function value at rejected trial points. Therefore, nothing changes if the large quantity is replaced by ∞. However, as we are going to see in other examples, the influence of large quantities on the behavior of the algorithm is more subtle.

Roughly speaking, the level sets of the function $e^{x_1+\ldots+x_n}$ are similar to those of $\max\{x_1,\ldots,x_n\}$ when one component is dominant. Moreover, when many components are similar and large the exponential tends to assume a typical "nearly-nonsmooth" shape. In fact, for any smooth function $f(x_1,\ldots,x_n)$, given a point z where the gradient is not null, the probability of obtaining a point x such that $f(x) < f(z)$ in a neighborhood of radius $\delta > 0$ tends to $\frac{1}{2}$ when $\delta \to 0$. But taking $n = 100$, $z = (100,\ldots,100)$ and $\delta = 1$ we obtain that the probability of obtaining $e^{x_1+\ldots+x_n} < e^{z_1+\ldots+z_n}$ with $\|x-z\|_\infty < 1$ is around 0.006. On the other hand, the probability of having $\|x\|_2 < \|z\|_2$ with $\|x-z\|_\infty < 1$ is around 0.49 and, of course, the probability of having $\|x\|_\infty < \|z\|_\infty$ with $\|x-z\|_\infty < 1$ is 2^{-100}.

Finally, our algorithm, as many other algorithms for nonlinear programming, is based on quadratic approximations. Unfortunately, the quadratic (Taylor) model of e^x is a poor approximation of this function if $x > 0$ is large. If $z = 10$ and $\delta = 1$ the error of replacing e^x by its second order approximation in $|x-z| \leq \delta$ is around 3600, but for $z = 50$, $\delta = 1$ this error exceeds 10^{20}. This probably indicates that methods based on quadratic models cannot be very efficient when the objective function involves several exponentials with not very small arguments.

These observations lead us to suggest a modification of the exponential Lagrangian method which consists on replacing the exponential by a quadratic, if the argument exceeds a given value, in such a way that the function, the first and second derivatives are continuous.

Therefore, the modification consists on defining

$$\theta(z, \mu) = \mu \operatorname{axp}(z),$$

where

$$\operatorname{axp}(z) = e^z \text{ if } z \le \beta,$$
$$\operatorname{axp}(z) = e^\beta + e^\beta(z-\beta) + e^\beta(z-\beta)^2/2 \text{ if } z > \beta.$$

It is easy to see that Properties **P1** and **P2** hold for this definition.

5 Numerical experiments

We tested the classical PHR augmented Lagrangian method and the exponential Lagrangian method with the modification suggested in the previous section using some typical nonlinear programming problems:

Problem 1: Find *npun* points on the unitary sphere of $\mathbb{R}^{ndim}$ such that maximum scalar products between them is minimum. (This is equivalent to say that the minimum distance is maximum.) The nonlinear programming problem has been defined as

$$\text{Minimize } z$$

$$\text{subject to}$$

$$\|x_k\|_2^2 = 1, k = 1, npun,$$

$$z \ge \langle x_i, x_j \rangle \text{ for all } i \ne j.$$

$$x_k \in \mathbb{R}^{ndim}, k = 1, \dots, npun.$$

The solution of this problem is the set of vertices of the polyhedron showed in Picture 1.

As initial approximation we took $[x_k]_i$ and z randomly in $[-1, 1]$. In Tables 1, 2 and 3 we show the performance of the augmented Lagrangian PHR method and the modified exponential Lagrangian method for $(ndim, npun) = (3, 24), (3, 30)$ and $(4, 25)$ respectively and for different choices of β. The configuration of the tables for all the problems is similar:

"Outer" is the number of augmented Lagrangian iterations (number of times in which (3) is solved;

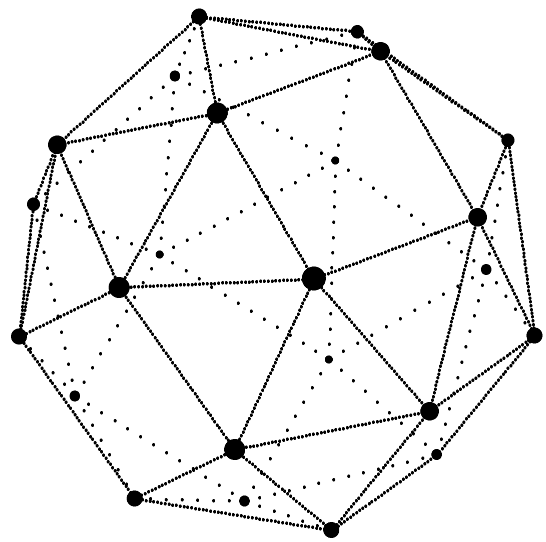
"Inner" is the number of iterations performed by BOX;

"Evaluations" is the number of times in which the augmented Lagrangian was evaluated;

"Q.It." is the number of iterations of QUACAN;

"MVP" is the number of "matrix vector products", which in this case involve an additional augmented Lagrangian gradient evaluation;

"Time" is the CPU time (seconds) of the execution using Microsoft double precision Fortran 77 in a Pentium with 90 MHz.



Picture 1: Solution of Problem 1 (3, 24)

For this problem, the last column "dist" is the minimum distance between points at the solution obtained. (In fact, to prevent small violations of equality constraints, the solution computed by the algorithm was first normalized so that all the points "really" belong to the unitary sphere.) This problem has *npun* equality constraints. Since for this class of constraints the classical PHR augmented Lagrangian scheme does not present discontinuity problems, we dealt with them using that standard procedure.

Problem 2: We took 132 points in the classical map of America (taken from the New York Times) with 17 countries and we formulated the problem of finding the closest possible 132 points in the plane such that the area of each country is the true one with a 1 percent precision. Therefore, if the 132 data points are $y_1, \ldots, y_{132} \in I\!R^2$, the objective function is $\frac{1}{2}\sum_{k=1}^{132} \|x_k - y_k\|_2^2$ while the 34 constraints of the problem are of the form

$$0.99 \ \times \ \text{True area} \leq \text{Computed area} \leq 1.01 \ \times \ \text{True area} \ .$$

for each one of the 17 countries considered. The solution of this problem is the map of America drawn in Picture 2.

As initial approximation we took $x_k = y_k, k = 1, \ldots, npun$. The final column of Table 4 represents the objective function value at the solution obtained.

Problem 3: This problem has been suggested by C. Gonzaga [13] to test sensitivity with respect to almost coincident constraints. It is a very simple problem which, on purpose, is not formulated in the best possible way. (Bound constraints are treated as explicit constraints $g_i(x) \leq 0$ instead of being included in Ω.) The problem is

$$\text{Minimize} \sum_{i=1}^{n} \frac{[x]_i}{i} \text{ subject to } [x]_i \geq 0,\ [x]_i \geq 0.001, i = 1, \ldots, n.$$

Clearly, its solution is $(0.001, \ldots, 0.001)$. We used $n = 1000$ (so $p = 2000$). The coordinates of the initial approximation were taken randomly between -10 and 10. The last column in Table 5 is the logarithm of the $\infty-$norm of the error.

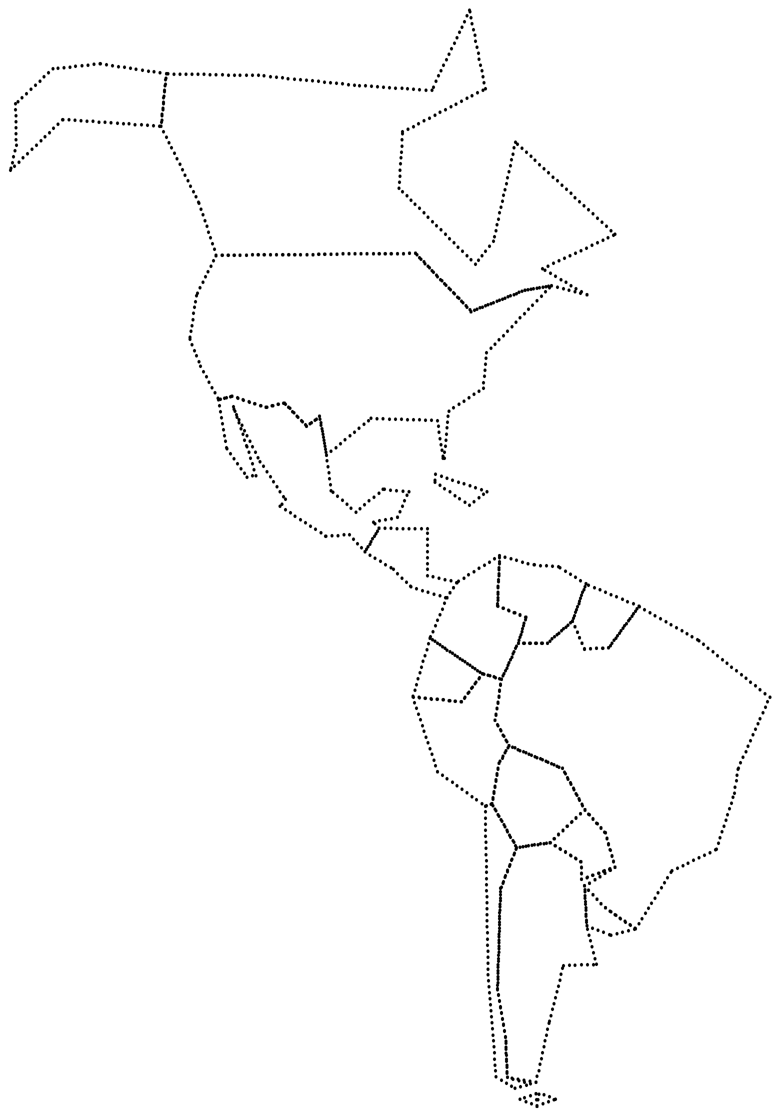
Problem 4: This problem consists on finding *npun* points in $I\!R^3$ such that the distance between any pair of them is not less than 1 and the maximum distance is as close to 1 as possible:

$$\text{Minimize } z$$

$$\text{subject to}$$

$$1 \leq \|x_i - x_j\|_2^2 \leq 1 + z$$

for all $i \neq j,\ i, j = 1, \ldots, npun$. The coordinates of the initial approximation were taken randomly between -10 and 10. After solving the problem using the augmented Lagrangian method, we computed the effective distances $\|x_i - x_j\|$. If any of them is (of course, slightly) smaller than 1, we replaced each x_k by " factor $\times x_k$" in such a way that the smallest distance is exactly equal to 1. For this normalized points we computed the maximum deviation of the distances with respect to 1. This number is the one of the last column of Tables 6 and 7 and reflects the quality of the solution obtained in practice.



Picture 2: Solution of Problem 2

Table 1: **Problem 1 (3, 24) (n = 72, m = 24, p = 276)**

Method	Outer	Inner	Evaluations	Q.It.	MVP	Time	dist
PHR	7	503	5069	15742	43976	316.0	.743830
$\beta = 0.$	6	48	78	1115	1396	69.5	.74 4206
$\beta = 0.1$	5	51	78	1125	1419	70.7	.744 206
$\beta = 1.$	5	55	85	1336	1703	84.6	.74 4206
$\beta \geq 10$	5	61	94	1453	1843	92.0	.744206

Table 2: **Problem 1 (3, 30) (n = 90, m = 30, p = 435)**

Method	Outer	Inner	Evaluations	Q.It.	MVP	Time	dist
PHR	8	434	3945	26990	47418	420.8	.657374
$\beta = 0.$	5	125	193	6057	6995	489.8	.660981
$\beta = 0.1$	5	124	187	5558	6446	456.9	.660981
$\beta = 1.$	5	138	216	5999	6770	479.0	.66 0981
$\beta = 10.$	5	160	242	6621	7593	532.9	.660981
$\beta = 20.$	5	157	233	6206	7202	507.0	.6 60981
$\beta = 100.$	5	157	233	6206	7202	507.0	. 660981

Table 3: **Problem 1 (4, 25) (n = 100, m = 25, p = 600)**

Method	Outer	Inner	Evaluations	Q.It.	MVP	Time	dist
PHR	7	635	4726	34528	64374	658.4	.957825
$\beta = 0.$	9	2451	5406	132620	156111	8927.8	.78 7241
$\beta = 0.1$	10	2762	6496	107417	128496	7416.5	.6 63185
$\beta = 1.$	6	202	314	14404	15561	952.5	.96 1487
$\beta = 10.$	6	237	380	15609	17192	1067.5	.9 61489
$\beta = 20.$	6	258	398	15772	17291	1074.0	.9 61487
$\beta = 100.$	6	258	398	15772	17291	1073.6	. 961487
$\beta = 500.$	6	258	398	15772	17291	1073.5	. 961487

Table 4: Problem 2 (n = 264 , p = 34)

Method	Outer	Inner	Evaluations	Q.It.	MVP	Time	f
PHR	7	110	931	3508	8505	162.4	3.050771
$\beta = 0.$	6	56	87	1493	2083	79.9	3.049630
$\beta = 0.1$	6	57	88	1367	1910	74.3	3.049630
$\beta = 1.$	6	64	98	1485	2183	85.1	3.049630
$\beta = 10.$	6	72	109	1287	1880	74.0	3.049630
$\beta = 20.$	6	80	115	1227	1717	68.5	3.049630
$\beta \in [100., 696.]$	6	109	139	973	1451	59.0	3.04 9630
$\beta \geq 697.$							overflow

Table 5: Problem 3 (n = 1000 , p = 2000)

Method	Outer	Inner	Evaluations	Q.It.	MVP	Time	log(Error)
PHR	6	331	964	9193	21446	1333.0	0.83
$\beta = 0.$	6	94	139	9578	13087	1400.1	-8
$\beta = 0.1$	6	104	158	10935	14969	1609.2	-8
$\beta = 1.$	6	113	171	11523	16329	1736.9	-8
$\beta = 10.$	6	136	198	10430	15071	1634.8	-8
$\beta = 20.$	6	149	217	11318	16325	1754.4	-8
$\beta = 100.$	6	224	287	12236	17646	1899.8	-8

Table 6: Problem 4 (3, 10) (n = 31 , p = 90)

Method	Outer	Inner	Evaluations	Q.It.	MVP	Time	deviation
PHR	9	143	369	1910	4070	19.7	.776874
$\beta = 0.$	3	48	70	565	722	31.4	.77 3190
$\beta = 0.1$	3	52	79	692	905	38.9	.773 183
$\beta = 1.$	2	45	68	500	683	29.8	.76 2397
$\beta = 10.$	3	67	96	743	961	41.9	.7 73175
$\beta = 20.$	3	132	195	1741	2151	92.4	.8 03061
$\beta = 100.$	3	161	248	994	1459	65.6	. 773175

Table 7: **Problem 4 (3, 20) (n = 61 , p = 380)**

Method	Outer	Inner	Evaluations	Q.It.	MVP	Time	deviation
PHR	6	314	844	5035	11532	120.1	1.537385
$\beta = 0.$	4	75	112	2020	2409	711.1	1.44 6678
$\beta = 0.1$	4	63	89	1498	1733	512.7	1.4 46662
$\beta = 1.$	4	95	138	2416	2875	857.4	1.44 6650
$\beta = 10.$	4	88	132	2399	2725	811.3	1.4 46650
$\beta = 20.$	4	108	161	2161	2598	781.6	1.4 46650
$\beta = 100.$	4	175	268	2249	2718	839.0	1. 446650

6 Conclusions

A qualitative analysis of the numerical results presented in the tables 1–8 shows that:

1. The exponential Lagrangian method with $\beta = 0$ is the best method in Problem 1 (3,24). The difference with $\beta = 0.1$ is, however, marginal. The performance of PHR is much poorer in this problem.

2. In Problem 1 (3,30) the best performance is the one of the exponential Lagrangian method with $\beta = 0.1$. PHR uses less computer time but the quality of the solution is worse. In spite of using less computer time, the number of matrix-vector products and the other quantitative indicators is much larger in the case of PHR than in exponential Lagrangian methods. This phenomenon is due, partially, to the higher cost of exponentials with respect to quadratics. However, more influent in the computer time is the fact that, in PHR, it is not necessary to evaluate the gradient of a constraint at a point where it is strongly satisfied, while in exponential Lagrangian algorithms all the constraint gradients must be evaluated independently of their degree of fulfillment.

3. In Problem 1 (4, 25), the exponential Lagrangian method with $\beta = 1$ is the best one. PHR uses less computer time, but the solution is worse and the number of MVP is four times larger than the one of the winner. In this case $\beta = 0$ and $\beta = 0.1$ give low-quality solutions while the solution for $\beta > 1$ has the same quality as that of $\beta = 1$.

4. In Problem 2, the solution has the same quality for all the exponential Lagrangian methods but is marginally worse in PHR. In this problem the computer time decreases as β increases, but the number of inner iterations goes in the opposite direction. This means that, when β grows, the quadratic subproblems become easier. That is to say, less MVP are necessary to achieve the required precision by QUACAN. As a result, more BOX iterations are necessary, but savings on MVP turn these alternatives more economic.

5. The most remarkable fact in Problem 3 is the poor quality of the solution obtained by PHR. All the exponential Lagrangian methods behaved similarly, with some advantage for $\beta = 0$. In fact, although the exponential Lagrangian methods were able to find the solution, their behavior in terms of quantitative indicators was far from being satisfactory. Although this problem has been introduced with the aim of testing the influence

of almost coincident constraints ($[x]_i \geq 0$ and $[x]_i \geq 0.001$) additional experiments using only $[x]_i \geq 0$ gave similar results. The key point seems to be the large number of active constraints at the solution, combined with the very different influence of the variables on the variation of the objective function. Both features make quadratic models inadequate. On the other hand, if the problem is scaled so that the variable $[x]_i$ is replaced by $[x]_i/i$ all the algorithms behave very well. Moreover, the algorithms (especially PHR) also behave very well if the initial approximation belongs to $[-20, -10]^n$ because in this case all the inner iterates tend to lie in a region that is infeasible with respect to all the constraints, where there exists a well conditioned quadratic model that represents well the true objective function of the subproblems. Recall that other obvious situation in which the problem is trivial is when the bounds are included in Ω, so that QUACAN deals with them.

   As a matter of fact, it seems that the minimization of $\sum_{i=1}^{1000} [x]_i/i$ subject to $x \geq 0$ is a very simple and surprisingly challenging problem for testing augmented Lagrangian methods.

6. In Problem 4 (3,10) the best solution is obtained by the exponential Lagrangian method with $\beta = 1$, which also uses the lowest computer time among exponential Lagrangians. PHR uses less computer time (although more MVP) but the quality of its solution is marginally worse.

7. Finally, in Problem 4 (3,20) PHR is the method that uses less computer time, but its solution is worse than all the solutions obtained by exponential Lagrangians. Among these, $\beta = 1$ is the best in terms of computer time, its solution being very marginally worse than that of $\beta > 1$.

Summing up, it appears from the numerical experiments that the solutions obtained by PHR are, in general, worse than those obtained by the exponential Lagrangian algorithms. This defficiency is related to lack of continuity in second derivatives. In fact, when second derivatives change abruptly, the model approximation ceases to be second-order, fast convergence rate is lost and BOX tends to stop with diagnostics of "small trust region", instead of "small projected gradient". The consequence is that a really small projected gradient and thus a good solution sometimes fails to be reached.

The expensiveness of exponential Lagrangian inner iterations and matrix-vector products with respect to the same indicators of PHR is also very impressive. As we mentioned above, the main reason is that one MVP in the case of the exponential Lagrangian involves necessarily all the constraints, while in PHR it involves only those which are not strongly satisfied.

The experiments presented in the previous section also reveal that the modification suggested here for the exponential Lagrangian method has advantages over the original algorithm. Although we cannot determine accurately the best possible value for β, it seems that for reasonably scaled problems some value around the unity is the best one.

In the next table, we show the number of matrix-vector products (so, auxiliary gradient evaluations) per inner iteration for the best choice of β at each problem. We also show the quotient of this number over the number of variables.

Table 8: Gradient evaluations

Table	MVP	Inner	MVP/Inner	MVP/Inner/n
1	1396	48	29	0.40
2	6995	25	56	0.62
3	15561	202	77	0.77
4	1451	109	13	0.05
5	13087	94	139	0.14
6	683	45	15	0.48
7	1733	63	28	0.46

This table shows that a considerable number of gradient evaluations per iteration is used in the truncated Newton approach for all the problems, except for the problem of Table 4, in which this number is very moderate. In one case (Table 3) the number of gradient evaluations per iteration is 77 percent of the number that should be used by a discrete Newton method.

The observation above shows that there is a lot of place for the development of quasi-Newton methods for problems with the described structure since, essentially these methods use only one gradient evaluation per iteration. Of course, we do not expect to reproduce the number of inner iterations of a truncated Newton method using quasi-Newton but it seems that even losing in terms of number of iterations, a quasi-Newton method could be more efficient in terms of overall performance. A little bit disappointing is the fact that we have tested the use of classical BFGS and Symmetric Rank-One corrections for these problems with results much poorer than the ones of the truncated Newton method.

References

[1] D. P. Bertsekas [1982], *Constrained optimization and Lagrange multiplier methods*, Academic Press, New York.

[2] R. H. Bielschowsky, A. Friedlander, F. M. Gomes, J. M. Martínez and M. Raydan [1995], An adaptive algorithm for bound constrained quadratic minimization, Technical Report, Institute of Mathematics, University of Campinas, to appear in *Investigación Operativa.*

[3] R. A. Castillo [1998], *Métodos de Lagrangeano aumentado usando penalidades generalizadas para programação não linear*, Tese de Doutorado, COPPE, Universidade Federal do Rio de Janeiro.

[4] A. R. Conn, N. I. M. Gould and Ph. L. Toint [1991], A globally convergent augmented Lagrangean algorithm for optimization with general constraints and simple bounds, *SIAM Journal on Numerical Analysis* 28, pp. 545 - 572.

[5] J. E. Dennis and R. B. Schnabel [1983], *Numerical methods for unconstrained optimization and nonlinear equations*, Prentice-Hall, Englewood Cliffs, N. J.

[6] M. A. Diniz-Ehrhardt, M. A. Gomes-Ruggiero and S. A. Santos [1997], Comparing the numerical performance of two trust-region algorithms for large-scale bound-constrained

minimization, in *International Workshop of Numerical Linear Algebra and Optimization*, edited by R. J. B. Sampaio and J. Y. Yuan, Department of Mathematics, Universidade Federal do Paraná, Brazil, pp. 23-24.

[7] Z. Dostál, A. Friedlander and S. A. Santos [1997], Analysis of semicoercive contact problems using symmetric BEM and augmented Lagrangians, to appear in *Engineering Analysis with Boundary Elements.*

[8] A. V. Fiacco and G. P. Mc Cormick [1968], *Nonlinear programming: sequential unconstrained minimization techniques*, John Wiley, New York.

[9] R. Fletcher [1987]: *Practical Methods for Optimization*, John Wiley and Sons, Chichester.

[10] A. Friedlander and J. M. Martínez [1994], On the maximization of a concave quadratic function with box constraints, *SIAM Journal on Optimization* 4, pp. 177-192.

[11] A. Friedlander, J. M. Martínez and S. A. Santos [1994], A new trust region algorithm for bound constrained minimization, *Applied Mathematics and Optimization* 30, pp. 235-266.

[12] P. E. Gill; W. Murray; M. H. Wright [1981], *Practical optimization*, Academic Press, London.

[13] C. C. Gonzaga [1997], Private communication.

[14] W. W. Hager [1985], Approximations to the multiplier method, *SIAM Journal on Numerical Analysis* 22, pp. 16–46.

[15] W. W. Hager [1987], Dual techniques for constraint optimization, *Journal of Optimization Theory and Applications* 55, pp. 37–71.

[16] A. N. Iusem [1997], Augmented Lagrangian and proximal point methods for convex optimization, Technical Report, Institute of Pure and Applied Mathematics, Rio de Janeiro, Brazil.