Working Paper 01–09
Statistics and Econometrics Series 06
January 2001

Departamento de Estadística y Econometría
Universidad Carlos III de Madrid
Calle Madrid, 126
28903 Getafe (Spain)
Fax (34) 91 624 9849

A DECOMPOSITION PROCEDURE BASED ON APPROXIMATE NEWTON DIRECTIONS

A.J. Conejo, F.J. Nogales and F.J. Prieto*

Abstract

The efficient solution of large-scale linear and nonlinear optimization problems may require exploiting any special structure in them in an efficient manner. We describe and analyze some cases in which this special structure can be used with very little cost to obtain search directions from decomposed subproblems. We also study how to correct these directions using (decomposable) preconditioned conjugate gradient methods to ensure local convergence in all cases. The choice of appropriate preconditioners results in a natural manner from the structure in the problem. Finally, we conduct computational experiments to compare the resulting procedures with direct methods, as well as to study the impact of different preconditioner choices.

Keywords: Conjugate gradient; preconditioner; local convergence; nonlinear optimization

* Conejo, E.T.S. Ingenieros Industriales, Univ. de Castilla-La Mancha, Ciudad Real, Spain, e-mail: aconejo@ind-cr.uclm.es; Nogales, E.T.S. Ingenieros Industriales, Univ. de Castilla-La Mancha, Ciudad Real, Spain, e-mail: fjnogale@ind-cr.uclm.es; Prieto, Dept. of Statistics and Econometrics, Univ. Carlos III de Madrid, Spain, e-mail: fjp@est-econ.uc3m.es

A decomposition procedure based on approximate Newton directions

A.J. Conejo, F.J. Nogales, F.J. Prieto

E.T.S. Ingenieros Industriales
Univ. de Castilla-La Mancha, Ciudad Real, Spain
aconejo@ind-cr.uclm.es fjnogale@ind-cr.uclm.es

Dept. of Statistics and Econometrics
Univ. Carlos III de Madrid, Spain
fjp@est-econ.uc3m.es

ABSTRACT

The efficient solution of large-scale linear and nonlinear optimization problems may require exploiting any special structure in them in an efficient manner. We describe and analyze some cases in which this special structure can be used with very little cost to obtain search directions from decomposed subproblems. We also study how to correct these directions using (decomposable) preconditioned conjugate gradient methods to ensure local convergence in all cases. The choice of appropriate preconditioners results in a natural manner from the structure in the problem. Finally, we conduct computational experiments to compare the resulting procedures with direct methods, as well as to study the impact of different preconditioner choices.

AMS: 90C26, 90C30, 49M27

1 Introduction

Traditionally, the optimization literature has been interested in studying and developing procedures that solve optimization problems by treating small parts of these problems separately. The growth in the size and complexity of optimization models during the past years, and the increasing availability of hardware and software for distributed computation, has led to an expansion of this interest. These procedures should be able to exploit any special structures that may be present in the original model, such as network constraints, integer and continuous variables, dynamic dependencies, etc. Even in those cases where no special structure is present, decomposition techniques may allow the efficient use of distributed computation resources for the solution of very large problems.

Many different decomposition techniques have been proposed during the past forty years. Early proposals were based on linear programming and theoretical results from convex analysis, such as the well-known Dantzig-Wolfe decomposition [11] and its dual variant, Benders decomposition [3]. Another frequently used technique is the Lagrangian Relaxation procedure [16, 22, 25], based on convex analysis theory; it is related to the preceding ones in the sense that it becomes equivalent to the Dantzig-Wolfe decomposition for linear problems if the cutting plane method [4] is used to update the multipliers. The Lagrangian Relaxation technique requires the use of nondifferentiable optimization procedures to maximize the dual function, and this may lead to slow and oscillatory convergence. Relaxation techniques based on Augmented Lagrangian functions [10, 5] attempt to address these difficulties and allow for extensions to nonconvex cases by combining penalization methods with local duality theory. The additional penalization terms may not have the separability properties of Lagrangian Relaxation transformations; therefore, additional problem manipulations are required to enforce these properties, such as for example those proposed by [29, 23, 24].

In some cases, the procedures based on relaxation techniques may present drawbacks: difficulties to converge to an optimal solution (in the absence of convexity assumptions), and convergence rates that are very sensitive to the choice of values for the parameters [25, 19]. Moreover, these techniques may require the computation of a very precise solution of the subproblems resulting from each set of values of the parameters in each iteration. As a consequence, the computational effort may be quite high, particularly for large-scale problems.

There exist many other decomposition techniques that take the structure of the problem into account to specialize classical optimization procedures, such as for example those of [6, 9, 8]. These techniques achieve a high level of efficiency, at the price of being strongly dependent on the particular form of the problem.

Our area of interest is the application of decomposition techniques to the solution of large-scale nonlinear, and possibly nonconvex, problems with special structure. The recent development of interior-point techniques for the solution of linear and quadratic problems has implied that these problems can be efficiently solved by transforming them into nonlinear (although generally convex) problems. In this sense, the methods that we will describe should be useful for the solution of general (linear and nonlinear) continuous optimization problems with special structure.

In this paper we will assume that this special structure will fit one of the two following patterns:

- Complicating constraints. Problems with complicating constraints arise frequently in numerous applications, including multicommodity network flows (in scheduling and transportation models), allocation of scarce resources among competing activities [1], design and management of water supply systems and electric power network analysis [31], logistics, econometric data fitting and statistics [2], etc. These models can be written in the general form

$$\begin{aligned} \text{minimize} \quad & f(x_1, \ldots, x_N) && (1)\\ \text{subject to} \quad & h_0(x_1, \ldots, x_N) \leq 0 && (2)\\ & c_j(x_j) \leq 0 \quad j = 1, \ldots, N, && (3) \end{aligned}$$

  where the constraints (2) are referred to as *complicating constraints*; if they were removed, the resulting problem would be separable.

- Complicating variables. Problems with complicating variables appear in scenario analysis and stochastic optimization [26], financial planning under uncertainty [21] and structural optimization, among others. The corresponding models can be written as

$$\begin{aligned} \text{minimize} \quad & f(x_0, x_1, \ldots, x_N) && (4)\\ \text{subject to} \quad & h_j(x_0, x_j) \leq 0 \quad j = 1, \ldots, N && (5)\\ & c_0(x_0) \leq 0, && (6) \end{aligned}$$

  where the variables x_0 in (4)-(6) are referred to as *complicating variables*; if they were fixed to constant values, the resulting problem would be separable.

In both cases $x_j \in \mathbb{R}^{n_j}$, and f and each h_j, c_j are smooth functions for $j = 0, \ldots, N$, with $f : \mathbb{R}^n \to \mathbb{R}$, $h_j : \mathbb{R}^n \to \mathbb{R}^{m_{h_j}}$ and $c_j : \mathbb{R}^{n_j} \to \mathbb{R}^{m_{c_j}}$, where $n = \sum_j n_j$.

Standard optimization techniques for the solution of the preceding problems use variants of Newton's method to find solutions of the system of nonlinear equations that defines the first-order conditions for the corresponding problem. These methods solve in each iteration a system of linear equations (the KKT system) with a coefficient matrix defined in terms of the first and second derivatives of the functions in the problem. The proposals presented in this paper will depend crucially on the observation that the special structures described above are also apparent both in the derivatives of the functions and in the coefficient matrices of the resulting linear systems.

The decomposition method we propose is related to Lagrangian decomposition techniques in that it solves decomposed subproblems obtained after fixing some of the primal and dual variables in the Lagrangian function. Nevertheless, the updating rules for these fixed variables differ from those standard in Lagrangian decomposition methods. Alternatively, the proposed method can be seen as being based on exploiting the problem structure present in the KKT systems in an efficient manner to approximate them by separable linear systems. In this sense, it is related to the many proposals in the literature for the distributed solution of linear systems of equations, see for example [27]. The proposed techniques are also adapted to the fact that we wish to solve a sequence of related linear systems, one for each iterate. We do not need a very precise solution for each one of these systems, as a sufficient approximation to Newton's direction is enough to ensure reasonable convergence properties in the algorithm. In this

regard, the proposed procedure is also related to optimization methods based on the inexact solution of the subproblems, see for example [12].

The resulting decomposition methodology is simple, having very few parameters to consider, efficient and very well suited for its implementation in a distributed computation environment. It is also very general, as it is able to handle both complicating constraints and complicating variables in a similar manner. On the other hand, the approximations introduced to decompose the problem imply that the superlinear rate of convergence of a pure Newton approach will in general become a linear rate of convergence for the decomposition algorithms. This reduction in the convergence rate is often in practice more than offset by the computational gains achieved through the solution of the smaller problems obtained from the decomposition. The linear rate of convergence can be improved using a preconditioned conjugate gradient (PGC) method [18, 17] for the solution of the linear systems of equations. In particular, we will use a generalized minimal residual (GMRES) procedure [28, 18], as the coefficient matrix of the system of equations will not be positive definite in general.

The remaining of this paper is organized as follows: In Section 2 we will motivate the proposed procedure as a modified Lagrangian relaxation technique. Section 3 describes the proposed procedure and studies its local convergence properties. In Section 4 we present different preconditioner choices for the proposed algorithm. We also describe in Section 5 practical implementation details and computational results from the application of this procedure to the solution of both nonlinear and linear problems. Finally, Section 6 states some conclusions.

2 A modified Lagrangian relaxation procedure

In this section we try to motivate the proposed procedure, by considering a particular implementation of Lagrangian relaxation for the complicating constraints case, (1)-(3). In Section 3 we will show that this implementation is a particular case of the more general decomposition procedure we describe in this paper.

To simplify our discussion, we consider the case in which we have only two groups of variables ($N = 2$) and additionally all constraints are equality ones. The simplified problem will have the form

$$\begin{aligned} \text{minimize} \quad & f(x_1, x_2) && (7)\\ \text{subject to} \quad & h_1(x_1, x_2) = 0 && (8)\\ & h_2(x_1, x_2) = 0 && (9)\\ & c_j(x_j) = 0 \quad j = 1, 2, && (10) \end{aligned}$$

where we have introduced some partition of the constraints h_0 into h_1 and h_2.

The basic Lagrangian procedure applied to this problem considers the auxiliary problem

$$\begin{aligned} \text{minimize} \quad & f(x_1, x_2) - \bar{\lambda}_1^T h_1(x_1, x_2) - \bar{\lambda}_2^T h_2(x_1, x_2) && (11)\\ \text{subject to} \quad & c_j(x_j) = 0 \quad j = 1, 2, && (12) \end{aligned}$$

defined in terms of multiplier estimates $\bar{\lambda}_1$ and $\bar{\lambda}_2$. Problem (11)-(12) can be solved by fixing the values of some of the variables ($\bar{x}_2$ and $\bar{x}_1$) to obtain the subproblems

$$\begin{aligned} \text{minimize} \quad & f(x_1, \bar{x}_2) - \bar{\lambda}_1^T h_1(x_1, \bar{x}_2) && (13)\\ \text{subject to} \quad & c_1(x_1) = 0, \end{aligned}$$

and

$$\begin{aligned} \text{minimize} \quad & f(\bar{x}_1, x_2) - \bar{\lambda}_2^T h_2(\bar{x}_1, x_2) && (14)\\ \text{subject to} \quad & c_2(x_2) = 0. \end{aligned}$$

Once the solutions for these subproblems have been computed, the multipliers of the complicating constraints can be updated, using for example a subgradient technique,

$$(\bar{\lambda}_1)_{k+1} = (\bar{\lambda}_1)_k - \alpha h_1(x_1, x_2), \quad (\bar{\lambda}_2)_{k+1} = (\bar{\lambda}_2)_k - \alpha h_2(x_1, x_2). \tag{15}$$

Note that the convergence of the procedure requires that the solutions for the subproblems should be computed up to a certain degree of accuracy.

The procedure proposed in this paper follows a similar approach when applied to problem (7)-(10). As in the preceding case, to decompose problem (11)-(12) we require some separable approximation for both f, h_1 and h_2. We also fix some of the variables in these functions to their last computed values, to obtain

$$\begin{array}{ll} \text{minimize} & f(x_1, \bar{x}_2) - \bar{\lambda}_2^T h_2(x_1, \bar{x}_2) \\ \text{subject to} & h_1(x_1, \bar{x}_2) = 0 \\ & c_1(x_1) = 0, \end{array} \tag{16}$$

and

$$\begin{array}{ll} \text{minimize} & f(\bar{x}_1, x_2) - \bar{\lambda}_1^T h_1(\bar{x}_1, x_2) \\ \text{subject to} & h_2(\bar{x}_1, x_2) = 0 \\ & c_2(x_2) = 0, \end{array} \tag{17}$$

where $\bar{x}_1$ and $\bar{x}_2$ denote the values of the corresponding variables at the last iterate. To reduce the computational cost, we perform a single iteration for each subproblem before updating the parameters $\bar{x}_1$ and $\bar{x}_2$.

This procedure is not very different from a standard Lagrangian approach, except for performing a single iteration for each subproblem, but it presents one significant advantage: it provides efficient information to update the multiplier estimates $\bar{\lambda}_1$ and $\bar{\lambda}_2$. The multipliers corresponding to the subproblem constraints (16) and (17), $\Delta\lambda_1$ and $\Delta\lambda_2$, have the property that, if the values of $\bar{x}_1$ and $\bar{x}_2$ would be the optimal ones, the best values for λ_1 and λ_2 would be given by $\bar{\lambda}_1 + \Delta\lambda_1$ and $\bar{\lambda}_2 + \Delta\lambda_2$. These updated values can be used for the next iteration.

The resulting procedure is very simple to implement, uses few easily updated parameters and does work well in practice for certain classes of problems (see Section 5.2). As a consequence, it seems reasonable to study conditions under which this simplified Lagrangian decomposition scheme would converge. In the following sections we will present a decomposition scheme that generalizes the procedure described above to solve different classes of problems, and we analyze its convergence properties.

3 Proposed algorithm

The algorithm we briefly described in the preceding section uses specific procedures to update the values of the variables and multipliers from one iteration to the next. In this section we wish to expand this procedure into a more general decomposition algorithm that is efficient to implement and requires as few parameters as possible, while preserving the basic schemes introduced in the preceding section, that is, to approximate the functions in the problem by fixing some of the variables to attain separability.

To simplify our presentation, the description of this algorithm will be based on the complicating constraints model (1)-(3). At the end of the section we will indicate how to extend the corresponding results to the complicating variables model (4)-(6).

We will assume that problem (1)-(3) will be solved using interior-point techniques. These procedures simplify the description of the algorithm by transforming the problem into an equality constrained one; they are also computationally very efficient, particularly for large-scale problems. Thus, the optimization problem will be solved through a sequence of barrier problems in which the inequality constraints have been rewritten as equalities using slack variables, and the bounds on the variables have been removed via the addition of barrier terms to the objective function. For a description of the use of these techniques in the general nonconvex case see for example [32, 13, 33, 14].

The resulting model, to be used as our reference in the description of the decomposition procedure, will have the general form

$$\text{minimize} \quad f(x_1, \ldots, x_N; \mu) \tag{18}$$

$$\text{subject to} \quad h(x_1, \ldots, x_N) = 0 \tag{19}$$

$$c_j(x_j) = 0 \quad j = 1, \ldots, N, \tag{20}$$

where we assume $x_j \in \mathbb{R}^{n_j}$, $f : \mathbb{R}^n \to \mathbb{R}$, $h : \mathbb{R}^n \to \mathbb{R}^{m_h}$, $c_j : \mathbb{R}^{n_j} \to \mathbb{R}^{m_{c_j}}$, $n = \sum_j n_j$. The function f includes the barrier terms corresponding to the bounds on the variables, and μ denotes the corresponding barrier parameter. These terms are not relevant for the description of the decomposition procedure, as in the barrier terms we take into account only simple bounds on the variables (all other constraints are transformed into equalities), and the resulting terms are trivially separable.

An implicit assumption for decomposition procedures is that the number of complicating constraints is not very large, that is, $m_h \ll \sum_j m_{c_j}$. The description of the algorithm does not require that this condition holds, but the efficiency of our procedure will depend on these values, as we will discuss in Section 5.1.

In this paper we are mostly interested in the analysis of the local convergence of the proposed decomposition procedure. As a consequence, the algorithm introduced in this section will not consider mechanisms to ensure global convergence, such as line searches or trust regions. The scheme of this basic algorithm is presented in Figure 1, where the values σ_j and λ will denote the multiplier estimates for the constraints $c_j(x_j) = 0$ and $h(x) = 0$, respectively. We will denote by x the vector of all (primal) variables, $x = \begin{pmatrix} x_1^T & \dots & x_N^T \end{pmatrix}^T$, while σ denotes the vector $\sigma = \begin{pmatrix} \sigma_1^T & \dots & \sigma_N^T \end{pmatrix}^T$. In Figure 1, the positive constant ϵ represents a termination tolerance and the function L denotes the Lagrangian function of problem (18)-(20).

Choose initial values x_0, σ_0 and λ_0
Let $k \leftarrow 0$
while $\|\nabla L(x_k, \sigma_k, \lambda_k)\| + \sum_j \|c_j(x_{j_k})\| + \|h(x_k)\| > \epsilon$
  Compute a search direction Δ_k for variables and multipliers, *(Inner iteration)*

$$\Delta_k = \begin{pmatrix} \Delta x_k \\ \Delta \sigma_k \\ \Delta \lambda_k \end{pmatrix}$$

  Update $x_{k+1} \leftarrow x_k + \Delta x_k$
  Update $\sigma_{k+1} \leftarrow \sigma_k + \Delta \sigma_k$
  Update $\lambda_{k+1} \leftarrow \lambda_k + \Delta \lambda_k$
  $k \leftarrow k + 1$
end while

Figure 1: Basic algorithm

No explicit procedure is given for the update of the barrier parameter μ. We will assume that any of the procedures proposed in the literature, see [32, 13] for example, is used to update this parameter. In this sense, we will be mostly concerned with the (local) convergence of the procedure for a fixed value of μ.

The decomposition procedure depends crucially on the definition of the inner iteration, that is, the procedure to compute the search direction Δ_k. Our method of reference for this inner iteration is Newton's method; in this setting, in each outer iteration k the search direction is computed by solving the following system of linear equations:

$$\begin{pmatrix} \nabla^2 L(x, \sigma, \lambda) & \nabla c^T(x) & \nabla h^T(x) \\ \nabla c(x) & 0 & 0 \\ \nabla h(x) & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ -\Delta \sigma \\ -\Delta \lambda \end{pmatrix} = - \begin{pmatrix} \nabla L(x, \sigma, \lambda) \\ c(x) \\ h(x) \end{pmatrix}, \tag{21}$$

where

$$\nabla c(x) = \begin{pmatrix} \nabla c_1(x) & & \\ & \ddots & \\ & & \nabla c_N(x) \end{pmatrix}, \tag{22}$$

and all quantities are evaluated at the current iterate (the subscript k corresponding to the iteration has been omitted to simplify the notation).

Observe that, within an interior point framework, this representation would correspond to a primal approach. Primal-dual approaches are considered to be more efficient than primal ones, but for a primal-dual approach the resulting linear system would be equivalent to (21), after simplifying the update for the dual variables. Both systems would only differ in the diagonal terms for the first block of coefficients. Again, this does not affect the decomposition procedure and we will ignore these details in the description of the inner iteration.

In Section 2 we introduced an approach for solving problems of this kind, based on Lagrangian relaxation. That procedure proceeded by fixing the values of some of the variables to obtain decomposable subproblems, and then used the solutions of these subproblems to update the fixed variables. This approach is equivalent to replacing the Newton system of linear equations for problem (18)-(20) by a separable system that approximates it. In particular, fixing variables in the Lagrangian relaxation approach is equivalent to approximating the matrix $\nabla^2 L$ in (21) by a block diagonal matrix, while the matrix ∇h is replaced by a matrix having separable blocks.

To generalize this approach, note that the special structure in the original problem appears also in the constraints, in the sense that the linear system would be separable if two conditions were met: i) the constraints $h(x) = 0$ would be separable (would have a separable Jacobian matrix) or would not exist; and ii) the matrix $\nabla^2 L$ would be separable, that is, it would be block diagonal with blocks corresponding to the different groups of variables; this would be the case if both f and h are separable. In general, these conditions are not satisfied, although in many practical cases the departures from them are not large. The proposed method replaces (21) with a related and separable system, of the form

$$\begin{pmatrix} H & \nabla c^T(x) & A^T \\ \nabla c(x) & 0 & 0 \\ A & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ -\Delta\sigma \\ -\Delta\lambda \end{pmatrix} = - \begin{pmatrix} \nabla L(x,\sigma,\lambda) \\ c(x) \\ h(x) \end{pmatrix}, \tag{23}$$

where H and A are approximations to the Hessian matrix $\nabla^2 L$ and the Jacobian ∇h, respectively, having the property of being separable in the variables x_j. We will use the abbreviated notation $K\Delta_N = -g$ for system (21), and $\bar{K}\Delta_m = -g$ for the modified system (23).

The matrices H and A can be obtained by setting to zero a sufficient number of elements in the preceding matrices to ensure separability. For example, the terms in $\nabla^2 L$ corresponding to cross derivatives for variables belonging to different blocks could be replaced by zeros, while the constraints in ∇h could be divided into groups associated with each block of variables; the nonzero coefficients in each set of constraints not associated with the corresponding block of variables could then be set to zero. This is very similar to the approach used in Section 2, where the zeros were introduced by fixing the values of some of the variables in the subproblems.

We now analyze sufficient conditions (on the search directions) to guarantee the local convergence of the method shown in Figure 1.

3.1 A simple inner iteration

If system (21) is solved exactly to compute Δ_N, we have the standard Newton algorithm, that has assured local convergence under certain assumptions on the problem. As system (21) is not separable, we will not use this approach. Alternatively, consider a simplified inner iteration where at iteration k we define

$$\Delta_k \equiv \Delta_{m,k} = -\bar{K}_k^{-1} g_k. \tag{24}$$

In some cases Δ_m may be close enough to Δ_N to preserve some of the local convergence properties of Newton's method. This will happen if $\bar{K}_k$ is sufficiently close in some sense to K_k, that is, if the modification required in the system to make it separable is small. In these cases the problem can be directly decomposed by ignoring some of the elements in $\nabla^2 L$ and ∇h in the computation of the search directions, without taking any additional correction steps and without losing local convergence. We now give a result based on classical results from the theory of iterative methods for linear systems of equations, see for example [30], that provides sufficient conditions to ensure local convergence in this simplified setting.

For the remainder of this section, we will assume that problem (18)-(20) satisfies the following hypotheses: For a given second-order KKT point of problem (18)-(20), $y^* = (x^*, \sigma^*, \lambda^*)$, it holds that:

$A.1$ The functions f, c_j, h have Lipschitz-continuous second derivatives in an open set containing y^*.

$A.2$ The Jacobian of the constraints, $\begin{pmatrix} \nabla c_1^T(x_1) & \dots & \nabla c_N^T(x_N) & \nabla h^T(x) \end{pmatrix}^T$, has full row rank at y^*.

$A.3$ The sufficient second-order optimality conditions for problem (18)-(20) are satisfied at y^*.

We will also require the approximations $\bar{K}_k$ to satisfy the following condition:

$C.1$ The matrices $\bar{K}_k$ are nonsingular for any y_k, and the sequence $\{\bar{K}_k\}_k$ converges to a nonsingular matrix $\bar{K}^*$ as $y_k \to y^*$.

This condition is imposed to ensure that the solution of the system (24) should be well defined at all iterations, and its behavior should be reasonable near the solution. Any factorization approach that controls the ill-conditioning in the system should ensure its satisfaction.

In the following theorem K^* denotes the matrix K evaluated at y^*, and for any given matrix A, $\rho(A)$ denotes its spectral radius.

Theorem 1 *Under assumptions $A.1$ to $A.3$ and condition $C.1$, if at the second-order KKT point y^* it holds that*

$$\rho^* = \rho\left(I - (\bar{K}^*)^{-1}K^*\right) < 1, \tag{25}$$

then the procedure using (24) converges locally to y^* with linear rate at least equal to ρ^*.

Proof. Let y_k be an iterate that is sufficiently close to y^* so that assumption $A.1$ holds and K_k is nonsingular. This last property follows from the assumptions.

From the corresponding Taylor series expansion,

$$g_k = g^* + K^*(y_k - y^*) + o(\|y_k - y^*\|),$$

where $g^* = 0$. Also, by definition in the algorithm $y_{k+1} = y_k + \Delta_k$ and $\Delta_k = -\bar{K}_k^{-1}g_k$. Consequently,

$$y_{k+1} - y^* = y_k + \Delta_k - y^* = (I - \bar{K}^{-1}K^*)(y_k - y^*) + o(\|y_k - y^*\|)$$

and taking norms

$$\|y_{k+1} - y^*\| \leq \|I - \bar{K}_k^{-1}K^*\|\|y_k - y^*\| + o(\|y_k - y^*\|).$$

This result together with the condition on the spectral radius implies that the sequence $\{y_k - y^*\}$ converges to zero. Dividing by $\|y_k - y^*\|$ and taking limits, the result on the rate of convergence follows. $\square$

A consequence of this result is that certain problems, even if they are not directly separable, can be solved efficiently by computing a search direction from a separable system of equations (the one defined by $\bar{K}$). In particular, the proposed method based on the Lagrangian relaxation technique that we described in Section 2 would fit this framework and would be locally convergent under the conditions of Theorem 1.

Note that the convergence criterion in Theorem 1, $\rho^* < 1$, is the classical condition for the convergence of iterative methods for the solution of systems of linear equations, such as iterative refinement, Jacobi or Gauss-Seidel. It is not easy in general to check in advance if this condition will hold for a given problem. But the convergence condition of the decomposition algorithm and that for the convergence of the iterative refinement method are the same, and this fact would provide a simple way to check the convergence condition at a given iteration as the algorithm progresses. If iterative refinement generates convergent iterates for the KKT system at a sequence of iterations of the decomposition procedure, then the full algorithm should also converge.

3.2 A locally convergent algorithm

Our next step will be to develop a decomposition algorithm that is locally convergent, independently of any conditions on the approximating matrix $\bar{K}_k$. To attain this goal we will use a modification of the simple inner iteration proposed above, based on adding correction steps to the direction $\Delta_{m,k}$.

To compute $\Delta_{m,k}$ we need some factorization of the matrix $\bar{K}_k$, and this is a computationally expensive procedure. It would seem reasonable to obtain these correction steps through procedures that do not

require any additional factorizations. A simple way to do this is to apply an appropriate version of the preconditioned conjugate gradient (PCG) method (see [18]) to system (21), as the solution of this system has the desired local convergence properties. A natural choice for the preconditioner is the matrix $\bar{K}_k$, whose factors would be already available. As the matrix K is not positive definite (except perhaps in the unconstrained convex case), we have chosen to use GMRES (see [28]), a variant of the PCG procedure that does not impose any condition on the coefficient matrix of the system.

In each (outer) iteration k, the movement direction Δ_k is defined from the inner iteration indicated in Figure 2. In this inner iteration t_k denotes a positive termination tolerance to be specified later on; for

```
Solve K̄_k Δ_{m,k} = −g_k
Let Δ̃_0 ← Δ_{m,k} and i ← 0
while ‖K_k Δ̃_i + g_k‖ > t_k‖g_k‖
    Apply one iteration of GMRES using K̄_k as preconditioner
      to compute Δ̃_{i+1}
    i ← i + 1
end while
Δ_k ← Δ̃_i
```

Figure 2: Inner iteration

the convergence analysis we only require $0 < t_k \leq \bar{t} < 1$ for all k and some constant $\bar{t}$. Note that, within the inner iteration, all systems of equations have $\bar{K}$ as their coefficient matrix; in consequence they can be solved in a distributed manner.

The termination condition for the inner iteration is given in terms of the residuals of the system of Newton equations. This criterion has been chosen as it is easy to implement, it enforces the local convergence of the overall procedure, as we will see below, and it is efficient in practice. We now study the local convergence of the modified method. Note first that, as GMRES solves the system of equations $K_k\Delta_k = -g_k$, from the convergence of GMRES, the termination condition for the inner iteration is satisfied in a finite number of (inner) iterations.

We now establish that, close to a second-order KKT point y^* for problem (18)-(20), the method is locally convergent. As a first step, we start by relating the sizes of g_k and Δ_k.

Lemma 1 *Under assumptions A.1 to A.3, if y_k is close enough to y^*, then there exist constants r_1 and r_2 such that*

$$\|g_k\| \leq r_1\|\Delta_k\|, \qquad \|g_k\| \geq r_2\|\Delta_k\|.$$

Also, there exists a constant r_3 such that

$$\|\Delta_k\| \leq r_3\|y_k - y^*\|.$$

Proof. From the termination condition we must have

$$\|g_k + K_k\ \Delta_k\| \leq t_k\|g_k\|. \tag{26}$$

Also, from assumption $A.1$ it will hold that $\|K_k\| \leq \bar{r}$ for some constant $\bar{r}$ and all y_k close enough to y^*. Consequently, we will have that

$$t_k\|g_k\| \geq \|g_k\| - \bar{r}\|\Delta_k\| \Rightarrow \|g_k\| \leq \frac{\bar{r}}{1-\bar{t}}\ \|\Delta_k\|.$$

From assumptions $A.1$ to $A.3$ and condition $C.1$ it must follow that, for all y_k close enough to y^*, the smallest singular value of K_k is bounded away from zero. Let $\tilde{r} > 0$ be such a bound. Then

$$\|g_k + K_k\ \Delta_k\| \geq \|K_k\ \Delta_k\| - \|g_k\| \geq \tilde{r}\|\Delta_k\| - \|g_k\|,$$

and from (26),

$$(1+\bar{t})\|g_k\| \geq \tilde{r}\|\Delta_k\|,$$

implying the second inequality.

Finally, from this second inequality and

$$g_k = g^* + O(\|y_k - y^*\|) = O(\|y_k - y^*\|),$$

we obtain the third inequality. $\square$

We are now ready to prove a local convergence result for this algorithm.

Theorem 2 *Under assumptions A.1 to A.3 and condition C.1, the sequence $\{y_k\}$ generated using the algorithm in Figure 2 converges locally to y^* with linear rate of convergence no larger than $\bar{t}$.*

Proof. If y_k is close enough to y^*, from Lemma 1 and assumption $A.1$

$$g_{k+1} = g_k + K_k(y_{k+1} - y_k) + o(\|y_{k+1} - y_k\|) = g_k + K_k\Delta_k + o(\|\Delta_k\|).$$

Taking norms and using the termination condition for the inner iteration (26) and Lemma 1,

$$\|g_{k+1}\| \leq \|g_k + K_k\ \Delta_k\| + o(\|g_k\|) \leq t_k\|g_k\| + o(\|g_k\|).$$

From this inequality it follows that $g_k \to 0$, and from Lemma 1 this implies $y_k \to y^*$. Dividing by $\|g_k\|$ and taking limits it follows from Lemma 1 that the rate of convergence is at least equal to $\bar{t}$. $\square$

From this result, the choice of t_k should offer a compromise between the rate of convergence of the algorithm, and consequently the number of outer iterations required for convergence, and the computational cost of each outer iteration. In section 5.1 additional information will be provided on how to choose t_k from a practical point of view.

3.3 The complicating variables case

The preceding paragraphs have analyzed problems with complicating constraints. We consider now the complicating variables case (4)-(6) in a similar manner. If we apply a barrier approach to the solution of this problem, we obtain the resulting model

$$\begin{array}{lll} \text{minimize} & f(x_0, x_1, \ldots, x_N; \mu) & (27) \\ \text{subject to} & h_j(x_0, x_j) = 0 \quad j = 1, \ldots, N & (28) \\ & c_0(x_0) = 0, & (29) \end{array}$$

where the objective function includes the relevant barrier terms. If we apply Newton's method, the corresponding KKT system of equations (at a given iteration) would have the form

$$\begin{pmatrix} \nabla^2 L(x,\sigma,\lambda) & \nabla h^T(x) & \nabla c_0^T(x_0) \\ \nabla h(x) & 0 & 0 \\ \nabla c_0(x_0) & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ -\Delta\sigma \\ -\Delta\lambda \end{pmatrix} = - \begin{pmatrix} \nabla L(x,\sigma,\lambda) \\ h(x) \\ c_0(x_0) \end{pmatrix}, \tag{30}$$

where now $x = \begin{pmatrix} x_0^T & x_1^T & \cdots & x_N^T \end{pmatrix}^T$, σ_j denotes the multiplier for $h_j(x_0, x_j)$, $\sigma = \begin{pmatrix} \sigma_1^T & \cdots & \sigma_N^T \end{pmatrix}^T$, λ is the multiplier for $c_0(x_0) = 0$ and

$$\nabla h(x) = \begin{pmatrix} \nabla_{x_0} h_1 & \nabla_{x_1} h_1 & & \\ \vdots & & \ddots & \\ \nabla_{x_0} h_N & & & \nabla_{x_N} h_N(x) \end{pmatrix}.$$

The procedures described for problem (18)-(20), and in particular the construction of the approximate system (23), can be applied in the same manner to approximate this KKT system, to obtain

$$\begin{pmatrix} H & A^T & \nabla c_0^T(x_0) \\ A & 0 & 0 \\ \nabla c_0(x_0) & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ -\Delta\sigma \\ -\Delta\lambda \end{pmatrix} = - \begin{pmatrix} \nabla L(x,\sigma,\lambda) \\ h(x) \\ c_0(x_0) \end{pmatrix}, \tag{31}$$

for some approximations H and A to $\nabla^2 L$ and ∇h respectively. These approximations should be separable in the variables x_0, x_1, ..., x_N. For example, the blocks that correspond to the variables x_0 in ∇h could be made equal to zero (if that would not imply a loss of rank), as well as the blocks in $\nabla^2 L$ associated with cross derivatives for x_0 and x_i with $i \neq 0$. The algorithms described in Figures 1 and 2 could be applied with only the obvious modifications to adapt them to (30) and (31).

4 Preconditioning

The efficiency of the procedures described in Section 3 depends crucially on an appropriate choice of the preconditioner $\bar{K}_k$ for the KKT system in each iteration of the proposed method. More specifically, the convergence of the procedure in Figure 2 will depend on the proximity of $\bar{K}_k^{-1}K_k$ to the identity matrix I. If we write

$$\bar{K}_k^{-1}K_k = I + B_k, \tag{32}$$

then we have two criteria to select a good preconditioner in terms of the properties of the matrix B_k in (32).

- The matrix B_k has a small spectral radius. From Theorem 1 convergence would follow if $\sigma_{\max}(B_k) < 1$, where $\sigma_{\max}(B_k)$ denotes the largest singular value of B_k. The rate of local convergence would depend on the magnitude of this value, that is, it would be faster as $\sigma_{\max}(B_k)$ becomes smaller.

- The matrix B_k is a perturbation matrix of small rank. If exact arithmetic is used, the number of inner (GMRES) iterations required to find the exact solution of the system of linear equations (the Newton direction) is at most $r = \text{rank}(B_k) + 1$, see [17] for example; more precisely, it is at most equal to the number of different singular values of B_k. In practice, if these singular values are clustered in a small number of groups (for example, if many of them are equal to zero) then the preconditioned GMRES procedure would behave efficiently. This is the case for example if only a few of the rows and columns of K_k are modified to obtain $\bar{K}_k$.

Note also that the preconditioner must satisfy condition *C.1*, and in particular it must be a full-rank matrix.

Based on these remarks, we now present several preconditioner choices for the algorithm in Figure 2. All these choices are based on replacing some of the entries in the KKT matrix, K_k, by zeros. If these entries are chosen appropriately, the resulting system can be trivially decomposed.

1. The complicating constraints can be decomposed by introducing zeros in ∇h to obtain a separable matrix. For example, by partitioning h into two subsets of constraints h_1 and h_2, we could define the matrix A in (23) as (again assuming $N = 2$ for simplicity)

$$A = \begin{pmatrix} \nabla_{x_1} h_1 & 0 \\ 0 & \nabla_{x_2} h_2 \end{pmatrix}.$$

   If this can be done without reducing the rank of A below that of ∇h, then the matrix $\bar{K}_k^{-1}K_k$ has at most $2m_h$ eigenvalues different from one, and the rest of eigenvalues are equal to 1. The rank of matrix B_k in (32) is $2m_h$.

   We have found that this procedure works reasonably well for problems with complicating constraints, but it usually leads to reduced-rank preconditioner matrices in problems with complicating variables.

2. An alternative approach that complements the preceding one if there are difficulties with the rank of the preconditioner, proceeds by replicating the complicating constraints (19) N times, if the resulting number of equations would not exceed the number of variables. The KKT matrix, K_k, would have the form (we assume $N = 2$ for simplicity)

$$\begin{pmatrix} \nabla^2 \tilde{L} & \nabla^T \tilde{h} \\ \nabla \tilde{h} & 0 \end{pmatrix},$$

   where

$$\nabla^2 \tilde{L} = \begin{pmatrix} \nabla^2 L & \nabla c^T(x) \\ \nabla c(x) & 0 \end{pmatrix}$$

   and

$$\nabla \tilde{h} = \begin{pmatrix} \nabla_{x_1} h & \nabla_{x_2} h & 0 \\ \nabla_{x_1} h & \nabla_{x_2} h & 0 \end{pmatrix}.$$

The right-hand side would be replicated in a similar manner. The preconditioner, $\bar{K}$, would have the form

$$\bar{K} = \begin{pmatrix} \tilde{H} & \tilde{A}^T \\ \tilde{A} & 0 \end{pmatrix},$$

where

$$\tilde{A} = \begin{pmatrix} A_1 & 0 & 0 \\ 0 & A_2 & 0 \end{pmatrix},$$

and $\tilde{H}$, A_1 and A_2 are approximations to the Hessian $\nabla^2 \tilde{L}$ and the Jacobians $\nabla_{x_1} h$ and $\nabla_{x_2} h$, respectively. It is important to select these matrices, A_1 and A_2, in such a way that the resulting preconditioner matrix $\bar{K}$ has full rank. Observe that the KKT matrix K_k is singular, but the system is compatible.

If $\tilde{H}$, A_1 and A_2 can be chosen to coincide exactly with the previous Hessian and Jacobian matrices, the matrix $\bar{K}_k^{-1} K_k$ would have $2m_h$ eigenvalues equal to zero, $2m_h$ eigenvalues equal to 2, and the remaining eigenvalues are equal to one. The resulting matrix B_k in (32) has rank $4m_h$.

This approach seems to work well in practice in problems with complicating variables, but is less adequate for problems with complicating constraints, as we will comment in Section 5.

3. We may consider other preconditioners based on the structure of the problem. Note though that it is not easy to construct simple (and efficient) preconditioners, as the KKT matrix, K_k, is not positive definite and in particular has a block of zeros in the diagonal. Nevertheless, this block of zeros can be handled via the Schur complement, for example. We now describe a preconditioner based on that of [8] that is based on this remark. We will use the notation $K\Delta_N = -g$ to represent (21). To solve this system we can compute the Schur complement of K with respect to its zero blocks. If the constraints are ordered so that the complicating constraints are the last ones, the resulting Schur complement has the form

$$S = \begin{pmatrix} \nabla c(x) \\ \nabla h(x) \end{pmatrix} \nabla^2 L(x, \sigma, \lambda)^{-1} \begin{pmatrix} \nabla c(x)^T & \nabla h(x)^T \end{pmatrix} = \begin{pmatrix} B & C^T \\ C & D \end{pmatrix}$$

where the matrix $B = \nabla c \; \nabla^2 L^{-1} \; \nabla c^T$ inherits the special structure of the original problem, at least if the objective function is separable. Using this property, the Schur complement of S with respect to D could be computed in a reasonably efficient manner; this new Schur complement is typically dense and a diagonal (or polynomial) preconditioner is used to approximate it. See [8] for additional details.

5 Numerical results

In this section we describe the implemented versions of the algorithm, and present numerical results obtained by applying several variants of the proposed decomposition algorithm to a set of test problems.

5.1 Practical implementation

Several versions of the algorithm (using different preconditioners $\bar{K}$) have been implemented in Matlab to test its behavior on linear and nonlinear problems with special structure. The implementation is based on the description in Figures 1 and 2, but it includes a few additional details that are included in Figure 3.

The following issues are of particular interest:

- The proposed algorithm carries out several iterative refinement iterations, while the norm of the residuals is decreasing. In practice, this is a cheap way to improve the quality of the computed direction, and it works quite well in many problems for which it holds that $\rho(\bar{K}^{-1}K) < 1$.

- The initial value for the termination tolerance for GMRES, t_k,is chosen in terms of the behavior of the problem in the first iteration.

Choose initial values x_0, σ_0 and λ_0
Set $r_0 = 0$ and let $k \leftarrow 0$
while $\|\nabla L(x_k, \sigma_k, \lambda_k)\| + \sum_j \|c_j(x_{j_k})\| + \|h(x_k)\| > \epsilon$
  Solve $\bar{K}_k\ \Delta_m = -g_k$ *(Inner iteration)*
  Compute $\omega_0 = \|g_k + K_k\ \Delta_m\|$
  Solve $\tilde{\Delta}_1 = \Delta_m - \overline{K}_k^{-1}(g_k + K_k\ \Delta_m)$
  Compute $\omega_1 = \|g_k + K_k\ \tilde{\Delta}_1\|$
  if $k = 0$
    $t_0 = 0.8\omega_0/\omega_1$
  end if
  Let $\tilde{\Delta}_0 \leftarrow \Delta_m$ and $j \leftarrow 0$
  while $\omega_{j+1} < \omega_j$ and $j < J$
    compute $\tilde{\Delta}_{j+1} = \tilde{\Delta}_j - \overline{K}_k^{-1}(g_k + K_k\ \tilde{\Delta}_j)$
    $j \leftarrow j + 1$
    $\omega_{j+1} = \|g_k + K_k\ \tilde{\Delta}_j\|$
  end while
  Let $\tilde{\Delta}_0 \leftarrow \tilde{\Delta}_j$ and $i \leftarrow 0$
  while $\|\bar{K}_k^{-1}(g_k + K_k\ \tilde{\Delta}_i)\| > t_k\|\bar{K}_k^{-1}g_k\| = t_k\|\Delta_m\|$
    Do one iteration of GMRES, using $\bar{K}_k$ as preconditioner, to compute $\tilde{\Delta}_{i+1}$
    $i \leftarrow i + 1$
  end while
  $\Delta_k \leftarrow \tilde{\Delta}_i$
  Update $x_{k+1} \leftarrow x_k + \Delta x_k$
  Update $\sigma_{k+1} \leftarrow \sigma_k + \Delta\sigma_k$
  Update $\lambda_{k+1} \leftarrow \lambda_k + \Delta\lambda_k$
  Compute $r_k = \|g_k + K_k\ \Delta_k\|$ and choose t_{k+1} as

$$t_{k+1} = \begin{cases} \min\{1.25t_k, 0.95\} & \text{if } r_{k-1}/r_k > 1, \\ \min\{0.25t_k, 0.95\} & \text{if } r_{k-1}/r_k < 0.99, \\ \min\{0.75t_k, 0.95\} & \text{otherwise,} \end{cases}$$

  $k \leftarrow k + 1$
end while

Figure 3: Decomposition algorithm

- The termination tolerance t_k is updated dynamically taking into account the rate of reduction in the residual norm r_k. This strategy is very similar to that used in trust-region methods to adjust the size of the region and works well in practice.

5.2 Test problems and results

We have tested the algorithm on a set of test problems from two models that present both coupling patterns described in Section 1. The first model, corresponding to the complicating constraints case, is a multi-area optimal power flow problem (OPF). In [31] a single-area formulation of the OPF problem is discussed. It is an important problem for the secure and economic operation of interconnected power systems; it determines, in a precise way, the active and reactive power that each generation unit in the system must generate, to ensure that all demand and security constraints for the system are satisfied, at a minimal cost for all interconnected areas. The resulting multi-area OPF problem is a large-scale non-convex optimization problem.

A model for this problem in compact form has the structure given in (1)-(3), where x_j would be

Table 1: Main characteristics of test problems.

Case	N	c	n	m
MOPF1	2	10	24	27
MOPF2	3	35	72	101
MOPF3	2	15	224	167
MOPF4	2	50	128	191
MOPF5	3	25	336	251
MOPF6	3	50	1032	1344
MOPF7	6	110	2064	2972
SP1	6	151	2845	1520
SP2	6	188	1820	686
SP3	16	151	7335	3900
SP4	16	188	4540	1726

the variables for each area j of the global system. Equations (2) represent the power flow equations and transmission capacity limits for those buses and lines interconnecting different areas. Constraints (3) include the power flow equations and transmission capacity limits, only for those lines and buses lying within a given area, and limits over dependent and control variables. The sets of equations (2)-(3) represent both equality and inequality constraints. In these models, the objective function (1) is the total operation cost for the system, a quadratic function of x_j for all j.

The second model, corresponding to the complicating variables case, is a two-stage stochastic programming model [7]. It is a large-scale linear program that minimizes the cost of first-period decisions plus the expected cost of second-period recourse decisions, while satisfying some first-period constraints. The stochastic problem is solved as a deterministic one by considering a discrete distribution with associated probabilities. Under these approximations, the *deterministic equivalent* form of the problem is

$$\underset{x,y_1,\ldots,y_N}{\text{minimize}} \quad c^T x + \sum_{i=1}^{N} \pi_i \; (q_i^T \; y_i) \tag{33}$$

$$\text{subject to} \quad Ax = b \tag{34}$$

$$B_k x + W_i y_i = h_i, \quad \text{for } i = 1, 2, \ldots, N \tag{35}$$

$$x \geq 0, \; y_i \geq 0, \quad \text{for } i = 1, 2, \ldots, N, \tag{36}$$

where x denotes the first-period variables, y_i denotes variables corresponding to second-period decisions and (34),(35) are the first-period and second-period constraints, respectively.

Table 1 shows the most relevant characteristics for a series of instances corresponding to each one of the two models. The first column gives the problem name; the second column indicates the number N of areas/scenarios; a third column shows the number of complicating constraints/variables; the fourth and fifth columns present the total number of variables and functional constraints, respectively.

The first seven cases are multi-area OPF models; their description can be found in reference [15]. Case I corresponds to the IEEE 9 bus system with two areas. Case II is based on the IEEE 30 bus system, replicated three times using seven interconnecting lines. Cases III and V are based on the IEEE RTS 24 bus system; it has been duplicated for case III, and replicated three times in case V. Case IV is based on the IEEE 57 bus system, divided into two areas with eleven interconnecting lines; the division has been chosen to force a large value for the spectral radius. Finally, cases VI and VII are based on the IEEE 118 bus system, replicated three times for case VI, and six times for case VII.

The next four problems correspond to a subset of POSTS, a portable stochastic programming test set, a small test set of stochastic programming recourse problems designed to highlight different qualities of general linear recourse problems [20].

The first set of results shown in Table 2 corresponds to the implementation described in Section 3.1 for a simplified inner iteration, corresponding to a special case of the Lagrangian decomposition algorithm.

Table 2: Numerical results using a simple inner iteration.

Case	I_d	T_d
MOPF1	30	$3.632\ 10^{-2}$
MOPF2	59	$1.975\ 10^{-1}$
MOPF3	38	$6.155\ 10^{-1}$
MOPF4		
MOPF5	58	$1.198\ 10^{0}$
MOPF6	39	$2.367\ 10^{-1}$
MOPF7	52	$8.245\ 10^{0}$
SP1		
SP2		
SP3		
SP4		

Table 3: Numerical results with proposed preconditioners.

Case	I_g	I_d	T_g	T_d	I_{CG}
MOPF1	13	13	$3.000\ 10^{-2}$	$0.000\ 10^{0}$	0
MOPF2	22	23	$2.200\ 10^{-1}$	$3.715\ 10^{-1}$	74
MOPF3	21	21	$5.130\ 10^{-1}$	$3.509\ 10^{-1}$	10
MOPF4	23	42	$5.000\ 10^{-1}$	$2.385\ 10^{0}$	391
MOPF5	26	26	$1.051\ 10^{0}$	$9.932\ 10^{-1}$	57
MOPF6	36	35	$8.473\ 10^{0}$	$7.108\ 10^{0}$	15
MOPF7	43	39	$3.585\ 10^{1}$	$2.513\ 10^{0}$	9
SP1	20	20	$7.117\ 10^{1}$	$9.857\ 10^{1}$	43
SP2	27	27	$1.857\ 10^{2}$	$4.410\ 10^{0}$	145
SP3	21	22	$5.928\ 10^{2}$	$9.103\ 10^{2}$	58
SP4	28	28	$1.183\ 10^{2}$	$1.086\ 10^{1}$	291

As in all other cases in this section, a primal-dual interior point procedure has been used to generate the equality constrained problems (18)-(20). The second and third columns of table 2 indicate the total number of iterations required by the decomposition procedure and the CPU time in seconds needed to solve the linear systems, respectively. The blank entries correspond to cases in which the algorithm did not converge, as the corresponding problems did not satisfy condition (25). It is remarkable to note that so many of the MOPF problems did in fact satisfy this condition, and the simple algorithm was able to obtain solutions for them.

Table 3 presents a comparison of the results obtained from applying a direct approach (Newton method) to the test problems, and the decomposition procedure described in figure 3. Both procedures have been initialized using the same starting point, and the stopping tolerances have been the same for both procedures. The choice of preconditioner for the decomposition procedure has been made to ensure that the resulting preconditioner satisfies condition $C.1$ (the preconditioner has full rank). For problems with complicating constraints, the preconditioner described in item 2 of Section 4 tends to produce matrices $\bar{K}_k$ that are rank deficient; as a consequence, for the MOPF problems we have used the preconditioner described in item 1 of Section 4. For problems with complicating variables, the situation is the opposite one, that is, preconditioner 1 tends to yield matrices $\bar{K}_k$ that are rank deficient; for the SP problems we have used the preconditioner described in item 2 of Section 4.

The second and third columns of table 3 indicate the total number of iterations required by the direct method (I_g) and the decomposition procedure I_d, respectively, for each one of the problems. The fourth and fifth columns show the CPU time in seconds needed to solve the linear systems in the direct approach and the decomposition procedure, respectively. Finally, the sixth column provides the total number of iterations performed by the GMRES subroutine. The results show the good behavior of the proposed procedure and its preconditioner: there is a reduction in running times for nearly all cases. Note that these results have been obtained in a sequential computation environment. When comparing these results with the ones in Table 2 it is also interesting to note that the simple procedure covered in that table does not produce much lower running times for those cases when it converges. The improvement in the

Table 4: Numerical results with other preconditioners.

Case	Schur preconditioner			Diagonal preconditioner		
	I_d	T_d	I_{CG}	I_d	T_d	I_{CG}
MOPF1	13	$1.400\ 10^{-1}$	23	13	$0.000\ 10^{0}$	0
MOPF2	22	$1.681\ 10^{0}$	128			
MOPF3	21	$4.024\ 10^{0}$	132	21	$8.430\ 10^{-1}$	29
MOPF4	23	$6.439\ 10^{0}$	257			
MOPF5	26	$1.189\ 10^{1}$	313			
MOPF6	36	$1.357\ 10^{3}$	246			
MOPF7	43	$1.708\ 10^{4}$	593	41	$6.068\ 10^{1}$	606
SP1	21	$3.832\ 10^{1}$	239			
SP2	27	$4.220\ 10^{1}$	184			
SP3	96	$5.915\ 10^{2}$	2785			
SP4	29	$1.064\ 10^{3}$	265			

number of major iterations compensates for the additional effort required by the procedure in Table 3 in each iteration.

Table 4 gives the results for the same problems when other preconditioners are used in the decomposition procedure; its format is similar to Table 3. Missing entries correspond to cases in which the corresponding algorithm failed to converge. The Schur preconditioner, described in item 3 of Section 4, required a large number of GMRES iterations, and consequently running times were higher than those in Table 3 in nearly all cases. The use of a simple diagonal preconditioner led to a reasonably fast algorithm in those few cases in which it converged, but the number of failures was very high. This result emphasizes the importance for convergence of fast and accurate results in the computation of the search direction.

6 Conclusions

We have discussed a decomposition algorithm motivated on a particular case of a Lagrangian relaxation procedure, that proceeds by computing an approximate solution to the KKT equations using a preconditioned conjugate gradient procedure. The natural choice of preconditioner based on a decomposable approximation to the system works very well in practice, better than other preconditioners and even that the direct solution of the system in many cases when this is feasible.

Another important issue we have considered is the termination criterion for the inexact computation of the search directions. We propose using a dynamic update of the termination tolerance based on the quality of the preceding directions. This approach results in a small number of conjugate gradient iterations, and also in a reduced number of Newton steps.

References

[1] M. S. Bazaraa, J. J. Jarvis, and H. F. Sherali. *Linear Programming and Network Flows.* John Wiley and Sons, New York, second edition, 1990.

[2] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms.* John Wiley, New York, second edition, 1993.

[3] J. F. Benders. Partitioning procedures for solving mixed variables programming problems. *Numer. Math.*, 4:238–252, 1962.

[4] D. P. Bertsekas. *Nonlinear Programming.* Athena Scientific, Belmont, Massachusetts, 1995.

[5] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods.* Prentice-Hall Inc., 1989.

[6] J. R. Birge and D. F. Holmes. Efficient solution of two-stage stochastic linear programs using interior point methods. *Computational Optimization and Applications*, 1:245–276, 1992.

[7] John R. Birge and François Louveaux. *Introduction to Stochastic Programming.* Springer-Verlag, New York, 1997.

[8] J. Castro. A specialized interior–point algorithm for multicommodity network flows, 1998. Acepted for publication in *SIAM Journal on Optimization.*

[9] Y. Censor and S. A. Zenios. *Parallel Optimization: Theory, Algorithms, and Applications.* Numerical Mathematics and Scientific Computation. Oxford University Press, 1997.

[10] G. Cohen. Auxiliary problem principle and decomposition of optimization problems. *Journal of Optimization Theory and Applications*, 32(3), Nov 1980.

[11] G. B. Dantzig and P. Wolfe. Decomposition principle for linear programming. *Operations Research*, 8:101–111, 1960.

[12] R. S. Dembo and U. Tulowitzki. On the minimization of quadratic functions subject to box constraints. Working Paper Series B 71, School of Organization and Management, Yale University, New Haven, Conn., 1983.

[13] A. S. El–Bakry, R. A. Tapia, T. Tsuchiya, and Y. Zhang. On the formulation and theory of Newton interior–point method for nonlinear programming. *Journal of Optimization Theory and Applications*, 89(3):507–541, Jun 1996.

[14] A. L. Forsgren and P. E. Gill. Primal–dual interior methods for nonconvex nonlinear programming. *SIAM Journal on Optimization*, 8:1132–1152, 1998.

[15] L. L. Freris and A. M. Sasson. Investigation of the load flow problem. *Proceedings of the IEE*, 115:1459–1470, 1968.

[16] A. M. Geoffrion. Lagrangean Relaxation for integer programming. *Mathematical Programming Study*, 2:82–114, 1974.

[17] G. H. Golub and C. F. Van Loan. *Matrix Computations.* The Johns Hopkins University Press, London, third edition, 1996.

[18] M. R. Hestenes and E. Steifel. Methods of conjugate gradients for solving linear systems. *Journal of Research National Bureau of Standards*, 49:409–436, 1954.

[19] J. B. Hiriart–Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms*, volume I and II. Springer Verlag, Berlin, 1996.

[20] D. Holmes. A (po)rtable (s)tochastic programming (t)est (s)et (posts). http://www-personal.umich.edu/ jrbirge/dholmes/post.html.

[21] P. Kall and S. W. Wallace. *Stochastic Programming.* John Wiley, Chichester, UK, 1994.

[22] J. E. Kelley. The cutting–plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8:703–712, 1960.

[23] R. de Leone, R. R. Meyer, S. Kontogiorgis, A. Zakarian, and G. Zaker. Coordination in coarse–grained decomposition. *SIAM Journal on Optimization*, 4(4):777–793, 1994.

[24] K. Park and Y. Shin. Iterative bundle–based decomposition for large–scale nonseparable convex optimization. *European Journal of Operations Research*, 111:598–616, 1998.

[25] B. T. Polyak. *Introduction to Optimization.* Optimization Software, Inc., New York, 1987.

[26] R.T. Rockafellar and Roger J.-B. Wets. Scenarios and policy aggregation in optimization under uncertainty. *Math. Oper. Res.*, 16(1):119–147, 1991.

[27] Y. Saad. *Iterative Methods for Sparse Linear Systems.* PWS Publishing, New York, 1996.

[28] Y. Saad and M. H. Schultz. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistic Computations*, 7:856–869, 1986.

[29] G. L. Schultz and R. R. Meyer. An interior point method for block angular optimization. *SIAM Journal on Optimization*, 1(4):583–602, 1991.

[30] R. S. Varga. *Matrix Iterative Analisys*. Prentice–Hall, Englewood Cliffs, NJ, 1962.

[31] A. J. Wood and B. F. Wollenberg. *Power Generation, Operation and Control*. John Wiley and Sons, New York, second edition, 1996.

[32] S. J. Wright. *Primal–Dual Interior–Point Methods*. SIAM Publications, SIAM, Philadelphia, PA, USA, 1996.

[33] H. Yamashita and H. Yabe. Superlinear and quadratic convergence of some primal–dual interior point methods for constrained optimization. *Mathematical Programming*, 75:377–397, 1996.