DECOMPOSITION FOR OPTIMAL POWER FLOWS

by

S.N. Talukdar, T.C. Giras * Y.K. Kalyan

December, 1932

DRC-13-52-3?

DECOMPOSITIONS FOR OPTIMAL POWER FLOWS

Sarosh N. TakJcdar Theo C. Giras Vibhu K. Kalyan

Power Engineering Program
Department of Electrical Engineering
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213

Abstract - The Han-Powell algorithm has proved to be extremely fast and robust for small optimum power flow problems (of the order of 100 buses). However, it balks at full size problems (of the order of 1000 buses). This paper develops a class of decompositions to break large problems down to sizes the Han-Powell algorithm can comfortably tackle. From this class we select one member - called the Super Hybrid - that seems to work best and describe it in detail.

I. INTRODUCTION

I.I. Optimum Power Flows

The general form of an optimum power flow problem is:

(OPF): Min f(u,x)
u,x

st: g(u,x) ≥ 0 (1)

h(u,x) » 0 (2)

where:

f is some cost of running the power system

U eR* is a vector of variables whose values are controlled by regulators. The set points of these regulators can be adjusted by the system operator, u consists largely of the real power outputs and voltage magnitudes of generators, transformer tap positions and loads that can be continuously managed, m is typically of order 100.

x eR is a set of state variables consisting largely of the reactive powers and voltage angles of generators and the voltages (magnitudes and angles) at non-generator buses, n is typically of order 1000.

g: R +R^q. The inequalities in (1) represent the systems operating constraints - equipment ratings and recommended practices. Typically g contains strongly nonlinear elements.

H: R™** -*n# The equalities in (2) are the system's power flow equations. They are nonlinear but only mildly so.

For further details on Optimum Power Flows see [1], [2].

1.2 More Notation

T denotes transpose

$z^T = [u^T, x^T]$

Au and Ax are changes in u and x.

Au and Ax are values of Au and Ax that define a local direction-of-movement that would be profitable to pursue in seeking an improvement to the incumbent estimate of the solution of (OPF).

f_u> f_x» G_u» G_x» H_u and H_x are first derivatives of f, g and h with respect to u^T and x^T.

a is a step length

y and X_are vectors of Lagrange multipliers

£ - f-y^Tg-X^Th, is the Lagrangian of (OPF)

f_z and $£_2$ a?e fi«t derivatives of f and £ with respect to z^T.

$V_u£$ is the reduced gradient of £ with respect to u^T.

Q is a positive definite approximation to the second derivative of £ with respect to z.

^uu'^ux' %ni aDd %x are Partitions of Q corresponding to u and x.

w (.) is a formula for updating the value of Q.

f^{u*}» x*l *• a Decomposition Point - a point used in converting a large quadratic programming problem to a smaller one.

a, B, C, d are coefficients of the reduced quadratic programming problem.

<D(.) Is a test function used in linear searches

N is the number of iterations needed to reach an optimal solution.

N is the value of N for the Han-Powell algorithm.

Subscripts: In dealing with iterations and the sequences of estimates they produce, we will primarily be concerned with a window from which one can see three successive estimates - the incumbent estimate, its immediate predecessor and its immediate successor. No subscript will be attached to the incumbent estimate. Its immediate predecessor will be identified by a "-" subscript, its immediate successor by a M+M. For example, the three estimates of u in order are: u_,u and u

X.3 The Han-Powell Algorithm

This is a Quasi Newton (Variable Metric) algorithm that was suggested by Han [3], [4] and refined by Powell [51, [61. As we shall see, it has some features that make it attractive for optimum power flows. First, however, we will outline the algorithm's steps.

In each iteration of the Han-Powell algorithm, the Incumbent estimate , $[u^T, x^T]$, to the solution of (OPF), is Improved by taking a step of length a in a direction-of-movement, $[\underline{Au}^T, \underline{Ax}^T]$. The new and Improved estimate is given by:

$[u^T, x^T] + a[\underline{Au}^T, \underline{Ax}^T]$

The direction-of-movement is found by solving a Quadratic Programming Problem. The objective of this problem is a second order approximation of f. The constraints are first order approximations to g and

h. Specifically, this problem is:

$$(QPP): \quad \underset{\Delta u, \Delta x}{\text{Min}} \; f_u \Delta u + f_x \Delta x + \tfrac{1}{2}[\Delta u^T, \Delta x^T] \; Q \; [\Delta u^T, \Delta x^T]^T \quad (3)$$

$$st: \quad g + G_u \Delta u + G_x \Delta x \geq 0 \quad (4)$$

$$h + H_u \Delta u + H_x \Delta x = 0 \quad (5)$$

In the first iteration Q is set equal to unity. Subsequently its value is updated with derivative information and a formula that maintains Q's positive definitnetness. The formula is given in detail in the Appendix. It has the form:

$$Q_+ = w(Q, z, z_-, \ell_z, \ell_{z_-}) \quad (6)$$

The step size, α, is chosen so that it produces a decrease in a test function, $\phi(\alpha)$, along the direction-of-movement. One form that this test function can take is given in the Appendix.

I.4 Strengths and Weaknesses

The advantages and most attractive features of the Han-Powell algorithm are:

1. It is fast. In tests on difficult problems it seems to converge much more quickly than competing methods [7], [8]. It tends to be especially fast on heavily constrained problems. In fact, as the number of active constraints, M, approaches the number of variables, m+n, the convergence approaches a quadratic rate. In the extreme circumstance of M = m+n the algorithm devolves to Newton's method for solving the active constraints. This close relationship to Newton's method is particularly desirable in view of the success that Newton's method enjoys in tackling power system equations.
2. It is robust. The Quadratic Programming Problem in each iteration tends to force convergence even under adverse circumstances such as profoundly infeasible starting points or problems whose Lagrangians have second derivatives matrices with large negative eigenvalves.
3. It is logically straightforward and therefore, is easy to program.
4. As with other Quasi-Newton methods, it automatically provides the information with which to determine the sensitivities of the optimal solution to variations in arbitrary parameters. For decision makers and analysts these sensitivities can be as useful as the optimal solution.

The disadvantages of the algorithm are:

1. If, for some reason, the algorithm stops before reaching an optimal solution, its last computed estimate may not even be feasible. The reason is that the algorithm often approaches the optimal solution from outside the feasible region.
2. The square matrix Q, which is used in (QPP), is nonsparse and its dimensions are equal to m+n, the number of variables in the problem. For problems with large numbers of variables (say 1000 or more) the Q-matrix is difficult, if not impossible, to deal with in (QPP).

The first disadvantage can be serious if very large amounts of computer time are involved. However, with power system problems we envision running times that are small enough so that the occasional need to rerun a program is not a matter for great concern.

The second disadvantage, however, prevents the Han-Powell algorithm from being used on full sized optimum power flow problems. The rest of this paper will be devoted to finding decompositions that eliminate the second disadvantage.

II. DECOMPOSITIONS

II.1 A General Framework

We consider here a class of decompositions that replacs (QPP), the quadratic programming problem in m+n variables, with a much smaller problem, (RQPP), in only m variables. The reduction is achieved by using the equality constraints in (OPF) to eliminate the state variables, x. The reduction is repeated at each iteration of the overall algorithm. Specifically, the steps involved in each iteration are:

Step 1: Select a point $[u_*^T, x_*^T]$ which will be called the Decomposition Point. This point could, but does not have to, be the same as the incumbent estimate $[u^T, x^T]$.

Step 2: Linearize the equality constraints about the Decomposition Point and express changes in x in terms of changes in u as follows:

$$\Delta x = - H_x^{-1} [H_u \Delta u + h] \quad (7)$$

Step 3: Select the coefficients, a,B,C,d of the reduced quadratic programming problem:

$$(RQPP): \quad \underset{\Delta u}{\text{Min}} \; a^T \Delta u + \tfrac{1}{2} \Delta u^T B \Delta u$$

$$st: \quad C\Delta u \geq -d$$

Step 4: Find the solution, $\underline{\Delta u}$, of (RQPP)

Step 5: Substitute $\underline{\Delta u}$ in (7) to get $\underline{\Delta x}$. The vector $[\underline{\Delta u}^T, \underline{\Delta x}^T]$ is the direction-of-movement. Now proceed to find a step length and a new estimate to the overall solution as in the full Han-Powell method.

Observe that variations among members of this class of decompositions are confined to the manner in which the Decomposition Point and the coefficients, a,B,C,d are chosen.

In the remainder of this section we will examine two existing decompositions and then proceed to synthesize an improved decomposition.

II.2 The BLW (Berna, Locke and Westerberg) Decomposition [8].

The necessary conditions for a solution of (QPP) are:

$$\begin{bmatrix} Q_{uu} & Q_{ux} & G_u^T & H_u^T \\ Q_{xu} & Q_{xx} & G_x^T & H_x^T \\ G_u & G_x & 0 & 0 \\ H_u & H_x & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta u \\ \Delta x \\ -\mu \\ -\lambda \end{bmatrix} \begin{matrix} = \\ = \\ \geq \\ = \end{matrix} \begin{bmatrix} -f_u \\ -f_x \\ -g \\ -h \end{bmatrix} \quad (8)$$

$$\mu^T [G_u \Delta u + G_x \Delta x + g] = 0$$

$$\mu \geq 0$$

If we eliminate Δx and λ from (8) we get:

$$\begin{bmatrix} B & C^T \\ C & 0 \end{bmatrix} \begin{bmatrix} \Delta u \\ -\mu \end{bmatrix} \begin{matrix} = \\ \geq \end{matrix} \begin{bmatrix} -a \\ -d \end{bmatrix} \quad (9)$$

where:

$$a^T = f_u - f_x H_x^{-1} H_u - h^T H_x^{-T} [Q_{xu} - Q_{xx} H_x^{-1} H_u] \quad (10)$$

$$B = Q_{uu} - H_u^T H_x^{-T} Q_{xu} - [Q_{ux} - H_u^T H_x^{-T} Q_{xx}] H_x^{-1} H_u \quad (11)$$

$$C - G_u - G_x H_x^{-1} H_u \quad (12)$$

$$d \ll g - G_x H_x^{-1} h \quad (13)$$

Notice that (9) has the sane structure as (8). Thus, (9) can be thought of as a set of necessary conditions for another and smaller quadratic programing problem, namely, (RQPP). This observation is the basis of the BLW Decomposition. Specifically, the Decomposition Point is chosen to be the same as the incumbent estimate and (BQPP)'s coefficients are calculated from expressions (10) - (13).

In exact arithmetic the BLV Decomposition produces the same results, iteration by iteration, as does the full Han-Powell algorithm. Most other decompositions converge more slowly, especially from distant starting points.

The main disadvantage of the BLV Decomposition is that it reduces but does not completely eliminate the usage of the Q-matrix. While this matrix does not appear in (RQPP), it is used in its entirety in calculating the coefficients of (RQPP). It would be better if one did not have to deal with the Q-matrix at all.

II.3 The Reduced Gradient Decomposition

This decomposition has been widely used in a variety of ways and with a variety of nonlinear programming methods, e.g. [10], 111]. In our context, the decomposition has two key features:

(a) The Decomposition Point is chosen to satisfy the equality constraints. This is done by setting u^*u and finding x_A so that

$$h(u,x^*) - 0 \quad (14)$$

(b) The coefficients of (RQPP) are calculated directly from reduced gradient information. The Q-matrix is not used.

The reduced gradients that we speak of here are first derivatives.with respect to u in an m-dimensional subspace of R^{**11}. Specifically, the subspace is the m-dimensional hyperplane that is tangent to the equality constraints at the Decomposition Point.

The intent of using reduced gradients is to effect a reduction in the size of (QPP) by projecting (QPP) into the hyperplane.

Points in the hyperplane are given by x • x^ + Ax and y * y* + Ay where:

$$H_u Au + H_x Ax - 0 \quad (15)$$

where H_u and H_x are evaluated at the Decomposition Point. The projection of (QPP) into the hyperplane is done in two steps. First, (15) is used to eliminate Ax from all but the quadratic term in (QPP)• Second, the quadratic term is replaced with the term: *h* Au^BAu. B is a positive definite approximation to the second derivative of £ in the hyperplane. However, instead of first calculating -Q and then projecting it into the hyperplane, B is obtained by direct updating with formula (6) and reduced gradient information. The results of these two steps yield expressions for the coefficients of (RQPP). The expressions are given below (except where otherwise specified, the quantities in the right hand sides must be evaluated at the Decomposition Point).

$$a^T = f_u * f_x H_x^{-1} H_u \sim f_x H_x^{-1} h \quad (16)$$

$$- f_u - f_x H_x^{*1} H_u \text{ because } h\ll 0 \quad (16a)$$

$$B \ll w(B_-, u, u_-, V_u £, \nabla_{u_-} £) \quad (17)$$

$$c - G_u - G_x H_x^{-1} H_u \quad , \quad (18)$$

$$d = S - G_x H_x^{-1} h \quad (19)$$

$$= g \text{ because } h - 0 \quad (19a)$$

The advantage of the Reduced Gradient Decomposition is that it eliminates all need to deal with the Q-matrix. However, it has two major disadvantages:

- It expends a good deal of effort in attempting to find Decomposition Points
- It can be slow to converge. Occasionally, it will fail to converge on problems that the full Han-Powell handles with ease. Consider, for instance, a problem in two variables with the constraint:

  $u^2 + x^2 \ll 4$ and a starting point of u«5, x-5.

  The first Decomposition Point sought by the Reduced Gradient Decomposition does not exist so the decomposition fails. Another example in five variables is given below [5]:

(EX): Min $x_1 x_2 x_3 u_1 u_2$

st: $x_1^2 + x_2^2 + x_3^2 + u_1^2 + u_2^2 - 10 = 0$

$x_2 x_3 - 5\, u_1 u_2 = 0$

$x_1^3 + x_2^3 + 1 = 0$

with a starting point of x_1 » -2, x« = 2, x_3 = 2, u_1 = -1, u_2 " -1. This is a difficult problem to solve and so is useful in testing methods. The Han-Powell algorithm solves it quite easily but the Reduced Gradient Decomposition fails.

II.4 Towards More Efficient Decompositions -

The total effort expended in solving (OPF) is determined by the product of J and N, where J is the average effort expended per iteration and N is the number of iterations needed to reach an optimal solution.

J can be kept small by ensuring that: (1) no effort is expended in separating the Decomposition Point from the Incumbent Estimate and (2) the Q-matrix is not used in calculating the coefficients of (RQPP).

N should not be much larger than N, the number of iterations required by the full Han-Powell. Otherwise, the decomposition loses much of its appeal as an alternative to the full Han-Powell.

In attempts to keep both J and N small, it has been suggested [12], [13] that the Decomposition Point be picked as in the BLV method (i.e. coincident with the Incumbent Estimate) and the coefficients of (RQPP) be calculated with expressions (16), (17), (18), (19) from the Reduced Gradient method. We will call the resulting decomposition the Hybrid. It works well on some problems but poorly on others, including (EX). The following observations provide some clues to why this happens and also contain the ingredients for a more robust decomposition.

(i) For the full Han-Powell algorithm to work well Q must be positive definite [14].

(il) In exact arithmetic and beginning with the same starting point, the full Han-Powell and the BLV decomposition produce identical sequences of estimates to a solution of (OPF).

(iii) By comparing expressions (10) - (13) with (16) - (19) we see that the BLV and Hybrid decompositions differ only in the values they use for a and B. These differences disappear when the following conditions are satisfied.

$$Q_{ux} - H_u^T H_x^{-T} Q_{xx} = 0 \quad (20)$$

$$\%u - {}^{B}_{Hybrid} + {}^{H}_{u} \left[{}^{H}_{x} \right] Q_{xu} \quad (21)$$

where B_{Hybrid} is the value of B computed from (17). Therefore, we can think of the Hybrid Decomposition as being a BLW Decomposition in which the Q-matrix has been chosen so that (20) and (21) are satisfied. But it may not be possible to make this Q-matrix positive definite. This may be why the Hybrid Decomposition sometimes performs poorly.

(iv) Both analysis [14] and empirical evidence [15] suggest that some pieces of the second derivative information in Q are more important than others. Suppose that z is partitioned into u' and x' so that the variables in x* are used to satisfy all the active constraints at the optimal solution of (OPF). Then the dimension of u' is the degrees of freedom left to minimize f. It happens that the diagonal block $Q_{u' u'}$ contains the information most important to the Han-Powell algorithm. The information is $Q_u?_x t$ and $Q^{\wedge},^{\wedge}$, is less important. The information in $Q_x»_x»$ is unimportant. (This is partially illustrated by the extreme case in which there are as many equality constraints as variables in z. Then x' * z; the Han-Powell algorithm devolves to Newton's method for solving the equalities; and Q exerts no influence on performance).

Since u' is a subset of u, we can arrange to preserve the most important second derivative information by retaining Q and discard the contents of the other parts of Q and replace them with entries that add no computational burden but keep Q positive definite. One way to do this is with a procedure we will call the Super-Hybrid Decomposition and describe below.

II.5 The Super Hybrid Decomposition

The essential steps of this decomposition are:

(a) Choose the Incumbent Estimate as the Decomposition Point.

(b) Set $Q_{xx} = 1$

$$Q_{xu} = Q_{ux}^T = 0$$

(c) Update Q_{uu} with either gradient or reduced gradient information (close to a solution the latter seems to work a little better).

(d) Evaluate a, B, C and d from expressions (10) - (13) and then proceed as with either the BLW or Reduced Gradient Decompositions.

The details of an algorithm that incorporates this decomposition are given in the Appendix.

The Super-Hybrid is attractive because its effort per iteration is low - the reasons are summarized in Table I - and because it seems to converge at least as fast, often faster, than other decompositions that do not use the full Q-matrix. For such decompositions, problem (EX) provides a very severe test. Notice, from the results given in Table II, that the Reduced Gradient and Hybrid decompositions fail when applied to (EX), but the Super-Hybrid manages to converge, albeit linearly. On less demanding problems, including optimum power flows, the Super-Hybrid seems to converge as fast as the full Han-Powell•

Finally, we note that the performance of a decomposition, particularly in regions far from a solution, seems to be much more sensitive to the manner in which the a-coefficient is calculated than the way in which B is updated. The best performance is obtained by using the BLW's expression, (10), for calculating a.

III. TEST RESULTS

III.l The Importance of Full Scale Testing

How many iterations will a nonlinear programming method take to solve a given problem? Usually, the best that theory can do in attempting to answer this question is tell us what the method's convergence rate is and whether it has quadratic termination. (The Han-Powell method is superlinearly convergent and does have quadratic termination). Convergence rates tell us how the method will behave close to a solution but not far away. Methods with quadratic termination will take at most M iterations to solve unconstrained problems with quadratic objective functions in M variables. In other words, we shouldn't be surprised if the number of iterations increases with problem size, even for simple unconstrained problems.

111.2 Some Results -

In [15] and [22] we demonstrated that the BLW Decomposition and certain variations on the Reduced Gradient Decomposition work very well on small OPFs. Since then we have developed the Super Hybrid Decomposition and tested it on a number of small and large systems. Some specimen results are shown in Table III. It seems that the number of iterations is not strongly affected by problem size. Infact, the Super-Hybrid often converges in about the same number of iterations as a Newton method takes to find a load flow solution (both procedures being started at the same point). Of course, a Super Hybrid iteration contains a load flow iteration and hence, involves more computations. Most of the additional effort goes into solving (RQPP). A rough estimate is that a Super Hybrid iteration requires from 2 to 10 times the effort involved in a Newton-load-flow-iteration. Numbers in the lower end of the range are obtained when the dimension of u is small in comparison to the dimension of x; numbers in the upper part of the range are obtained when the two dimensions are comparable.

111.3 Remarks

(i) Table IV compares some of the Super Hybrid's salient features with those of other OPF methods.

(ii) Newton-load-flow iterations become cheap relative to a Super Hybrid iteration when the number of retained variables (i.e. variables in u) is comparable to the number of eliminated variables (i.e. variables in x). In these circumstances, it makes sense to use a few more Newton-load-flow iterations wherever they can reduce the number of Super-Hybrid iterations. It happens that the greatest benefits from such additional iterations are obtained towards the end of the overall process, close to the optimal solution and not as intuition would suggest, at the starting point. The strategy governing additional iterations is to use them to keep the linear approximation to the eliminated variables at least as accurate as the estimate to the solution of the retained variables [22].

(iii) Suppose we want to consider not only the existing network configuration but also the configuration that could result from the occurrence of contingencies. This means that we would like to expand the constraints in an OPF formulation to include those of an existing configuration and also several additional configurations. Since the elimination of the state variables for each configuration can proceed independently, the eliminations can be processed n parallel on separate processors.

IV. CONCLUSIONS

This paper has identified a class of decompositions and from two of its existing members synthesized a third - called the Super-Hybrid - that is well suited to OPF problems. The algorithm obtained by combining the Super-Hybrid decomposition with the Han-Powell method of nonlinear programming has the following attractive features.

- Nonlinear objective functions and constraints can be accommodated directly, without tricks or intricate manoeuvres. This makes it easy to add complicated security constraints and to do real power dispatching, reactive power dispatching or both simutaneously. It also simplifies the coding, maintainance and updating of the algorithm.
- The decomposition works by eliminating some variables. It is convenient to choose these variables to be the same as the ones calculated in a load flow program. Then one iteration of a standard Newton-Load-Flow can be used to make the elimination.
- In tests on large problems the algorithm has proved to be fast. Often it will find an optimal solution in about as many iterations as a Newton method takes to find a load flow solution.
- The algorithm is robust. The starting point does not have to be feasible. Infact, the algorithm will force convergence from profoundly infeasible starting points.
- The information with which to calculate the sensitivity of the optimal solution to parameter variations is readily made available by the algorithm (though we have not yet taken advantage of this feature in the code we have written).

There are two principal factors that limit the type of problems that can be effectively handled by the algorithm. They are the numbers of retained variables (i.e. variables, in u) and inequality constraints. These factors determine the size of the quadratic programming problem that must be solved in each iteration of the algorithm.

The number of equality constraints is not a limiting factor because the equalities are eliminated by the decomposition. The elimination can be done with parallel processing when the equalities arise from several different network configurations as happens in contingency constrained optimum power flows [24], [25].

Available quadratic programming codes, e.g.[263» can efficiently handle about 300 retained variables. This is more than enough to accommodate all the regulated generator variables for most networks. However*, networks with large numbers of tap-changing transformers may boost the number of retained variables to 500 or so. Three possibilities for handling such situations are: ignore the less important tap changers or handle them in the second stage of a two-stage process; expand the capabilities of quadratic programming codes; use larger computers. We are not sure which of these possibilities is best. We suspect that the problem of how to handle large numbers of tap-changing transformers also remains to be suitably solved in other OPF packages.

V. ACKNOWLEDGEMENT

We are grateful to APS (Allegheny Power Service) and Duquesne Light for providing data with which to test our programs.

TABLE I: FEATURES AFFECTING THE EFFORT PER ITERATION OF FOUR DECOMPOSITIONS

Features	Decompositions			
	BLW	Reduced Gradient	Hybrid	Super-Hybrid
Effort expended to find DP	NONE, DP-IE	Considerable, DP is the solution of the equalities	NONE, DP-IE	NONE, DP-IE
Is the Q-matrix* of the eauivalent Han-Powell algorithm positive definite?	YES	Equivalent Han-Powell doesn't exist	NO	YES
Is the Q-matrix used in calculating the coefficients of (RQPP)?	YES	NO	NO	NO

DP: Decomposition Point
IE: Incumbent Estimate
*Even if the Q-matrix is not explictly used, it must be positive definite for the decomposition to work well.

TABLE II. RESULTS FOR PROBLEM (EX)

Iteration Number	Euclidian Norm of Error in Estimate				
	Han-Powell	BLW	Reduced Gradient	Hybrid	Super-Hybrid
0	0.6205	0.6205	0.6205	0.6205	0.6205
1	0.1409	0.1409			0.1409
2	0.1080	0.1080			0.0972
3	0.0874	0.0874			0.0791
4	0.0046	0.0046			0.0634
5	0.0001	0.0001			0.0507
6	0.0000	0.0000			0.0406
7			FAILS		0.0325
8					0.0260
9					0.0209

TABLE III: SPECIMEN RESULTS FOR THE SUPER-HYBRID ALGORITHM

Problem	AEP 30 Bus	APS 550 Bus	Hypothetical 1110 Bus
# of equalities	60	1110	2220
# of inequalities	44	160	320
# of elements in u	12	42	84
# of Newton Iterations to reach a Load Flow Solution	5	5	5
# of Super-Hybrid Iterations to reach an Optimal Power Flow Solution	7	6	9

TABLE IV: A REPRESENTATIVE SAMPLING OF NONLINEAR PROGRAMMING ALGORITHMS FOR OPF PROBLEMS [2].

Algorithm	Retained Variables	Directionrof Movement	Techniques for Handling Non-Linearities	Status	Comments
Dommael-Tinney [17]	u	Gradient	Penalty Functions	Many Production Programs	Probably the most coded algorithm. Works well when carefully tuned to a system. Detection of infeasi-bility is slow and the penalty functions are a disadvantage.
Generalized Reduced grad-ient (GRG) [18],[19]	Changed from one itera-tion to the next	Gradient	Variable Switching	At least one product-ion Program	Appears to be more robust than the Dommel-Tinney algorithm but the intricate variable ex-change mechanisms are a signifi-cant disadvantage.
Wu,Gross, Luini,Look, Gribik [20],[21]	Changed from one iter-ation to the next	Gradient	Variable Switching,Penalty Functions and Aug-mented Lagrangians	One Product*-ion Program	A modified GRG algorithm in which the variable switching is made compatible with a load flow pro-gram. A two stage approach is used. The solution of the first stage relaxed problem is used as a starting point for the second stage.
Super-Hybrid	u	Quasi-Newton	Linearization	Experi-mental	Simpler to program than other methods. Very fast and robust. Sensitivity data is readily avail-able. Can be decomposed for par-allel processing.
Burchett-Happ-Wirgau [23]	Changed from one iter-ation to the next	Quasi-Newton or Conjugate Gradient	Linearization and an Aug-mented Lagrangian	One Product-ion Program	The algorithm is similar to the Han-Powell except that in determining the direction-of-movement the Han-Powell uses a quadratic objective while the algorithm uses a more nonlinear objective. It seems that fhe algorithm requires about as many Iterations but more work per iteration than the Super-Hybrid.

REFERENCES

[1] B. Stott, 0. Alsac, J.L. Marinho, "The Optimal Power Flow Problem," Proceeding of the Int. Conf. on Electric Power Problems, the Mathematical Challenge, Seattle, 1980.

[2] Talukdar, Wu " Computer Aided Dispatch for Power Systems," invited paper, IEEE Proceedings, Vol. 69, No. 10 October, 1981.

[3] S.P. Han, "A Globally Convergent Method for Non-linear Programming", Technical Report 75-257, Cornell University, 1975.

[4] S.P. Han, "Superlinearly Convergent Variable Metric Algorithms for General Nonlinear Programming Algorithms", Mathematical Programming 11, 1976.

[5] M.J.D. Powell, "Algorithms for Nonlinear Constraints that Use Lagrangian Functions," Mathematical Programming 14, 1976.

[6] M.J.D.Powell, "A Fast Algorithm for Nonlinearly Constrained Optimization Calculations,"presented at the Dundee Conference on Numerical Analysis, 1977.

[7] T.C. Giras, S.N. Talukdar,"Quasi-Newton Method for Optimal Power Flows," Int. Journal of Electrical Power & Energy Systems Vol 3, No. 2 April 1981.

[8] T.J. Berna, M.H. Locke, and A.W. Westerberg, "A new approach to optimization of chemical processes" AIChE J., January 1980.

[9] R.H. Edahl, "Sensitivity Analysis in Nonlinear Programming: Applications to Comparative Statics and Algorithmic Construction,"Ph.D. thesis, Carnegie-Mellon University, Aug, 1982.

[10] L.M. Vidigal, S.W. Director, " A Design Centering Algorithm for Nonconvex Regions of Acceptability," IEEE Trans, on CAD, Vol. CAD-1, Jan 1982.

[11] R.L. Fox, "Optimization methods for engineering design", Edison-Wesley Publishing Co.,1971.

[12] M.H. Locke, R.H. Edahl, A.W. Westerberg, "An Im-Proved Successive Quadratic Programming Optimization Algorithm for Engineering Design Problems,"submitted for publication AIChE J, September 1982.

[13] S.N. Talukdar, "A Decomposition of the Han-Powell Algorithm,"CMU working paper, April, 1982.

[14] M.J.D. Powell, "The convergence of variable metric methods for nonlinearly constrainted optimization calculations" pres. at Nonlinear Pro. 3 Symp. Wisconsin.

[15] T.C. Giras,"A quasi-Newton method for optimal power flow solutions - a feasibility study,"MS Thesis, Carnegie-Mellon University, Pittsburgh, Pa. May 1980.

[16] M. Avriel," Nonlinear Programming; Analysis and Methods,"Englewood Cliffs, NJ: Prentice-Hall, 1976.

[17] H.W. Dommel, W.F. Tinney, "Optimal Power Flow Solutions," IEEE Jhans on PAS2 vol. PAS-87, pp. 1866-1876, 1968.

[18] J.Peschon, D.W. Bree, L.P. Hajdu, "Optimal solutions involving system security," in Proc. IEEE Pica Conf. (Boston, MA),pp.210-218, May 1971.

[19] J.Velghe, N. Peterson, "Optimal control of real and reactive power flow under constraints," in Proc. 4th PSCC (Grenoble, France), paper 2.1/4, Sept 1972.

[20] F.F. Wu, G. Gross,J.F. Luini, P.M. Look,"A two-state approach to solving large-scale optimal power flows," in Proc. PICA Conf., pp.126-136, 1979.

[211 P.R.Gribik and Pui Mee Look, "A Hybrid Technique for Solving Large-Scale Optimal Power Flow Problems", The International Congress on Tech. and Tech. Exchange, May 3-6, 1982.
[22] S.N. Talukdar ana T.C. Giras, "A Fast and Robust Variable Metric Method for Optimum Power Flows," IEEE Transactions on PAS, PAS-101, 1, February 1982, pp.415-425.
[23] R.C. Burchett, H.H. Happ, K.A. Wirgau," Large Scale Optimal Power Flows," IEEE Trans on PAS., Oct 1982, p.3722.
[24] S.N. Talukdar, S.S. Pyo, T.C.Giras, "Asynchronous Procedures for Parallel Processing" submitted to PICA-83.
[25] S.N. Talukdar, R.Mehrotra, N.Tyle, "A Framework for automatic security maintenance procedures", submitted to PICA-83.
[26] P.E.Gill, W.Murray, M.A.Saunders and M.H.Wright, Documentation for SOL/QPSOL: a Fortran package for Quadratic Programming, "Systems Optimization Laboratory, Stanford University.

APPENDIX; A SUPER HYBRID ALGORITHM

Step 0: Initialization

i. Choose u and x so that H_x is nonsingular for all values of u and x that are of interest. In the case of power systems choose x to be the variables solved for by a Newton-load-flow.

ii. Choose the starting point $[u^T, x^T]$

Step 1; Evaluate the first derivatives f_u, f_x, G_u, G_x, H_u, H_x and the residues h and g. These evaluations must be made at the incumbent estimate $[u^T, x^T]$.

Step 2; Evaluate a, C and d, coefficients for (RQPP), from:

$$a^T = f_u - f_x H_x^{-1} H_u + h^T H_x^{-T} H_x^{-1} H_u$$

$$C = G_u - G_x H_x^{-1} H_u$$

$$d = g - G_x H_x^{-1} h$$

Step 3: Update B. In the first iteration set B-l and E«l. (E is a dummy variable that makes writing the formulas a little easier.) In subsequent iterations update B and E using the formulas:

i. $$E = E_- - \frac{E_- \delta_u \delta_u^T E_-}{[\delta_u^T E_- \delta_u + \delta_x^T \delta_x]} + \frac{\eta_u \eta_u^T}{\delta^T \eta}$$

where $\delta = z - z_-$

$\delta_u = u - u_-$ and $\delta_x = X - X_-$

$\eta = \theta a + (1-\theta) E_- \delta$

$\theta = 1$ if $\delta^T y \geq 0.2\ \delta^T E_- \delta$

$$= \frac{0.8\delta^T E_- \delta}{[\delta^T E_- \delta - \delta^T y]} \quad \text{otherwise}$$

η_x is a vector of the last n elements of η

η_u is a vector of the first m elements of η

$$\gamma^T = \frac{\delta \ell(z,\mu,\lambda)}{\delta z^T} - \frac{\delta \ell(z_-,\mu,\lambda)}{\delta z^T}$$

$$\ell(z,V,\lambda) = f - y^T g - X^T h$$

y and X are calculated in the previous iteration in Steps 4 and 5.

ii. $$B = E + H_u^T H_x^{-T} H_x^{-1} H_u$$

(These formulas were obtained by making two sets of modifications to the BFGS updating formula [16]. The first modification was made by Powell [5] to ensure that the updates remained positive definite. The second modification was made by us to adapt the formulas to updating only the Q_{uu} partition of the Q-matrix . We note that the formulas given above update Q_{uu} with gradient information. Close to a solution, the use of reduced gradient information seems to work a little better. To include reduced gradient information one would replace Y^T with $V_u I - V_u \ell_-$.

Step 4: Solve the reduced quadratic programming problem:

$$\text{(RQPP):} \quad \min_{\Delta u} \ a^T \Delta u + \tfrac{1}{2} \Delta u^T B \Delta u$$

$$\text{st:} \quad B\Delta u \geq -d$$

Let Δu be the solution and y the Lagrange multiplier associated with the inequality constraints.

Step 5: Calculate Δx and X from:

$$\Delta x = -H_x^{-1} [H_u \Delta u + h]$$

$$\lambda = H_X^{-T} [f_X - G_X^T y + A^T \lambda]$$

(These expressions are obtained by rearranging the second and fourth equations in (8)).

Step 6: Find a step size, a, such that

$$0 < a \leq 1$$

and

where $\phi(a) < \phi(0)$

$$\phi(\alpha) = f(u(\alpha), x(a)) + \hat{y}^T |g(u(a),x(a))| + \hat{X}^T |h(u(a), x(a))|$$

$$u(a) = u + a\Delta u$$

$$x(a) = x + a\Delta x$$

|g| and |h| are vectors of the absolute values of g and h

$$\hat{U} = \max \{|y|, \ldots\} + \ldots$$

$$\hat{X} = \max \{|X|, \ldots\} + \ldots$$

If no such a can be found stop. Otherwise set:

$$u_+ = u + a\Delta u$$

$$x_+ = x + a\Delta x$$

(The test function ϕ used in this step has been given the form of a penalty function. This is not the only possibility. The Lagrangian works well too and other as yet undiscovered forms may work even better. Another observation is that the algorithm seems to work best when a is unity or close to it. Therefore, one should not perform a line search to seek out the smallest value of the test function along the direction-of-movement. Rather, one should seek the largest value of a that is unity or less and produces a decrease in the test function.)

Step 7: Check for convergence. If $a^2 \Delta u_+^T \Delta u_+ + \beta > \psi$, where δ is a norm of the constraint violations and ε is a tolerance, go to Step 1. Otherwise, stop.