MATPOWER's Extensible Optimal Power Flow Architecture

Ray D. Zimmerman, *Member, IEEE,* Carlos E. Murillo-Sánchez, *Member, IEEE,*
and Robert J. Thomas, *Fellow, IEEE*

Abstract—This paper describes the optimal power flow (OPF) architecture implemented in MATPOWER, an open-source Matlab power system simulation package. It utilizes an extensible architecture that allows the user to easily add new variables, constraints and costs to the standard OPF problem formulation while preserving the structure needed to use pre-compiled solvers. A software object is used to encapsulate the definition of the problem formulation, manage the corresponding named sets of variables, constraints and costs, and handle all of the tedious index maintenance tasks. The software design has the advantage of minimizing the coupling between variables, constraints and costs, making it possible, for example, to add variables to an existing model without having to explicitly modify existing constraints or costs to accommodate them. The example of adding joint co-optimization of reserves to the OPF, based on fixed zonal reserve requirements, is used to illustrate the capabilities of MATPOWER's extensible OPF architecture.

Index Terms—Load flow analysis, Optimization methods, Power engineering, Power engineering education, Power generation dispatch, Power systems, Power system economics, Power system simulation, Software packages, Software tools

I. INTRODUCTION

THIS paper describes the optimal power flow (OPF) architecture implemented in MATPOWER [1], an open-source Matlab power system simulation package. It is used widely in research and education for power flow and optimal power flow simulation. It also includes tools for calculating power transfer and line outage distribution factors (PTDF's and LODF's) and for running OPF-based auction markets.

The initial motivation for the development of the Matlab-based power flow and optimal power flow code that eventually became MATPOWER came from the computational requirements of the PowerWeb platform [2], [3]. PowerWeb is a web-based market simulation platform used to test electricity markets. It requires a "smart market" auction clearing software that uses an optimal power flow to compute the allocations and pricing. Having the clear potential to be useful to other researchers and educators, the software was released via the Internet as an open-source power system simulation package.

Even beyond its initial release, while some unrelated new features were added, the ongoing development of MATPOWER continued to be driven primarily by the needs of the PowerWeb project. This at least partially explains the lack of a graphical user interface used by some related tools such as PSAT [4].

As new OPF solvers were evaluated for improving MATPOWER's OPF performance, it became clear that MEX-based approaches, using Matlab extensions implemented in compiled languages such as C or Fortran, offered the potential for significant performance advantages. On the other hand, the frequent need to run a variation of the OPF with modifications and additions to the standard problem formulation would mean recompiling a new version of any MEX-based solver and modifying significant portions of Matlab code for each variation. The result was an effort to develop a general OPF architecture that allows modification of the formulation via optional input data instead of requiring extensive changes to the existing code.

This paper highlights MATPOWER's extensible OPF architecture, designed to allow the user to easily add new variables, constraints and costs to the standard OPF problem formulation while retaining the ability to utilize pre-compiled solvers. The next section describes the AC and DC versions of the standard OPF formulation as implemented in MATPOWER. The extensions to this formulation that form the basis for the extensible OPF architecture are detailed in Section III, with examples of how these capabilities are used internally shown in Section IV. Section V covers the currently available OPF solvers, Section VI describes the software architecture and Section VII illustrates the use of this extensible architecture to implement an OPF that jointly optimizes the allocation of energy and reserves.

II. STANDARD OPTIMAL POWER FLOW

MATPOWER includes code to solve both AC and DC versions of the optimal power flow problem. The standard version of each takes the following form.

$$\min_x f(x) \tag{1}$$

subject to

$$g(x) = 0 \tag{2}$$

$$h(x) \leq 0 \tag{3}$$

$$x_{\min} \leq x \leq x_{\max} \tag{4}$$

A. Standard AC OPF

The AC version of the standard OPF problem is a general non-linear constrained optimization problem, with both non-linear costs and constraints. In a system with n_b buses, n_g generators and n_l branches, the optimization variable x is defined in terms of the $n_b \times 1$ vectors of bus voltage angles Θ and magnitudes V and the $n_g \times 1$ vectors of generator real and reactive power injections P and Q as follows.

$$x = \begin{bmatrix} \Theta \\ V \\ P \\ Q \end{bmatrix} \tag{5}$$

The objective function (1) is simply a summation of individual polynomial cost functions f_P^i and f_Q^i of real and reactive power injections, respectively, for each generator.

$$\min_{\Theta,V,P,Q} \sum_{i=1}^{n_g} f_P^i(p_i) + f_Q^i(q_i) \tag{6}$$

The equality constraints (2) consist of two sets of n_b non-linear nodal power balance equations, one for real power and one for reactive power.

$$g_P(\Theta, V, P) = 0 \tag{7}$$
$$g_Q(\Theta, V, Q) = 0 \tag{8}$$

The inequality constraints (3) consist of two sets of n_l branch flow limits as non-linear functions of the bus voltage angles and magnitudes, one for the *from* end and one for the *to* end of each branch.

$$h_f(\Theta, V) \le 0 \tag{9}$$
$$h_t(\Theta, V) \le 0 \tag{10}$$

The variable limits (4) include an equality limited reference bus angle and upper and lower limits on all bus voltage magnitudes and real and reactive generator injections.

$$\theta_{\text{ref}} \le \theta_i \le \theta_{\text{ref}}, \qquad i = i_{\text{ref}} \tag{11}$$
$$v_i^{\min} \le v_i \le v_i^{\max}, \qquad i = 1 \ldots n_b \tag{12}$$
$$p_i^{\min} \le p_i \le p_i^{\max}, \qquad i = 1 \ldots n_g \tag{13}$$
$$q_i^{\min} \le q_i \le q_i^{\max}, \qquad i = 1 \ldots n_g \tag{14}$$

Here i_{ref} denotes the index of the reference bus and θ_{ref} is the reference angle.

B. Standard DC OPF

When using DC network modeling assumptions, the standard OPF problem above can be simplified to a quadratic program, with linear constraints and a quadratic cost function. In this case, the voltage magnitudes and reactive powers are eliminated from the problem completely and real power flows are modeled as linear functions of the voltage angles. The optimization variable is

$$x = \begin{bmatrix} \Theta \\ P \end{bmatrix} \tag{15}$$

and the overall problem reduces to the following form.

$$\min_{\Theta,P} \sum_{i=1}^{n_g} f_P^i(p_i) \tag{16}$$

subject to

$$g_P(\Theta, P) = A \begin{bmatrix} \Theta \\ P \end{bmatrix} + b = 0 \tag{17}$$

$$h_f(\Theta) = \quad B\Theta + c \le 0 \tag{18}$$
$$h_t(\Theta) = -B\Theta + d \le 0 \tag{19}$$

$$\theta_{\text{ref}} \le \theta_i \le \theta_{\text{ref}}, \qquad i = i_{\text{ref}} \tag{20}$$
$$p_i^{\min} \le p_i \le p_i^{\max}, \qquad i = 1 \ldots n_g \tag{21}$$

Here A and B are matrices and b, c and d are vectors, all constants arising from the DC network equations.

III. MATPOWER'S EXTENSIBLE OPF FORMULATION

For a research oriented tool, it is desirable to allow the end user the flexibility to modify or augment the problem formulation without rewriting the portions that are shared with the standard OPF formulation. On the other hand, when using pre-compiled solvers it is not generally feasible to employ the most general strategy of allowing the end user to supply their own code for evaluating cost functions, constraints and their derivatives.

As a tradeoff, MATPOWER pre-defines an extensible structure in which the modifications and additions to the standard OPF are specified in terms of optional input parameters. This allows for the use of pre-compiled solvers designed for this architecture while retaining the ability for the end user to modify the problem formulation.

The extensible OPF in MATPOWER modifies the standard formulation by introducing additional optional user-defined costs f_u, constraints, and variables z. The extended formulation can be written in the following form.

$$\min_{x,z} f(x) + f_u(x, z) \tag{22}$$

subject to

$$g(x) = 0 \tag{23}$$
$$h(x) \le 0 \tag{24}$$
$$x_{\min} \le x \le x_{\max} \tag{25}$$
$$l \le A \begin{bmatrix} x \\ z \end{bmatrix} \le u \tag{26}$$
$$z_{\min} \le z \le z_{\max} \tag{27}$$

A. User-defined Costs

The user-defined cost function f_u is specified in terms of parameters H, C, N, $\hat{r}$, k, d and m. All of the parameters are $n_w \times 1$ vectors except the symmetric $n_w \times n_w$ matrix H and the $n_w \times (n_x + n_z)$ matrix N. The cost takes the form

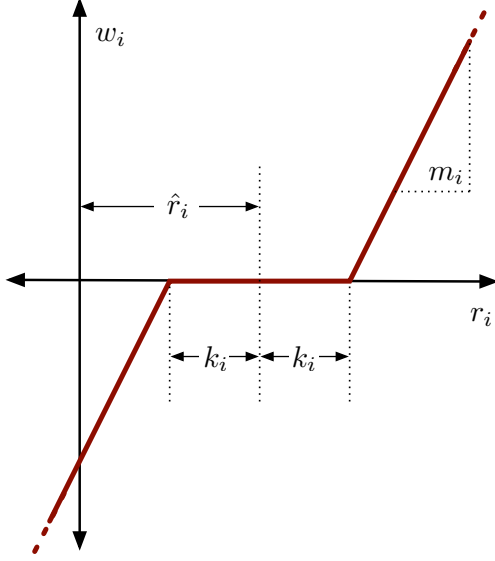$$f_u(x, z) = \frac{1}{2} w^{\mathsf{T}} H w + C^{\mathsf{T}} w \tag{28}$$

Fig. 1.   Relationship of w_i to r_i for $d_i = 1$ (linear option)

where w is defined in several steps as follows. First, a new vector u is created by applying a linear transformation N and shift $\hat{r}$ to the full set of optimization variables

$$r = N \begin{bmatrix} x \\ z \end{bmatrix}, \tag{29}$$

$$u = r - \hat{r}, \tag{30}$$

then a scaled function with a "dead zone" is applied to each element of u to produce the corresponding element of w.

$$w_i = \begin{cases} m_i f_{d_i}(u_i + k_i), & u_i < -k_i \\ 0, & -k_i \le u_i \le k_i \\ m_i f_{d_i}(u_i - k_i), & u_i > k_i \end{cases} \tag{31}$$

Here k_i specifies the size of the "dead zone", m_i is a simple scale factor and f_{d_i} is a pre-defined scalar function selected by the value of d_i. Currently, MATPOWER implements only linear and quadratic options

$$f_{d_i}(\alpha) = \begin{cases} \alpha, & \text{if } d_i = 1 \\ \alpha^2, & \text{if } d_i = 2 \end{cases} \tag{32}$$

as illustrated in Fig. 1 and Fig. 2, respectively.

This form for f_u provides the flexibility to handle a wide range of costs, from simple linear functions of the optimization variables to scaled quadratic penalties on quantities, such as voltages, lying outside a desired range, to functions of linear combinations of variables, inspired by the requirements of price coordination terms found in the decomposition of large loosely coupled problems encountered in our own research.

Some limitations are imposed on the parameters in the case of the DC OPF since MATPOWER uses a generic quadratic programming (QP) solver for the optimization. In particular, $k_i = 0$ and $d_i = 1$ for all i, so the "dead zone" is not considered and only the linear option is available for f_{d_i}. As a result, for the DC case (31) simplifies to $w_i = m_i u_i$.
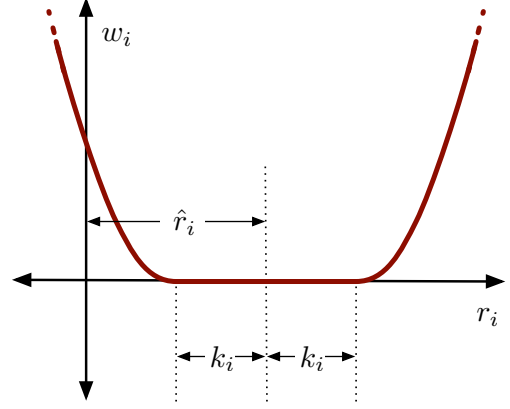

Fig. 2.   Relationship of w_i to r_i for $d_i = 2$ (quadratic option)

B. User-defined Constraints

The user-defined constraints (26) are general linear restrictions involving all of the optimization variables and are specified via matrix A and lower and upper bound vectors l and u. These parameters can be used to create equality constraints ($l_i = u_i$) or inequality constraints that are bounded below ($u_i = \infty$), bounded above ($l_i = \infty$) or bounded on both sides.

C. User-defined Variables

The creation of additional user-defined z variables is done implicitly based on the difference between the number of columns in A and the dimension of x. The optional vectors $z_{\min}$ and $z_{\max}$ are available to impose lower and upper bounds on z, respectively.

IV. STANDARD EXTENSIONS

In addition to making this extensible OPF structure available to end users, MATPOWER also takes advantage of it internally to implement several additional capabilities.

A. Piecewise Linear Costs

The standard OPF formulation in (1)-(4) does not directly handle the non-smooth piecewise linear cost functions that typically arise from discrete bids and offers in electricity markets. When such cost functions are convex, however, they can be modeled using a constrained cost variable (CCV) method. The piecewise linear cost function $c(x)$ is replaced by a helper variable y and a set of linear constraints that form a convex "basin" requiring the cost variable y to lie in the epigraph of the function $c(x)$.

Fig. 3 illustrates a convex n-segment piecewise linear cost function

$$c(x) = \begin{cases} m_1(x - x_1) + c_1, & x \le x_1 \\ m_2(x - x_2) + c_2, & x_1 < x \le x_2 \\ \vdots & \vdots \\ m_n(x - x_n) + c_n, & x_{n-1} < x \end{cases} \tag{33}$$
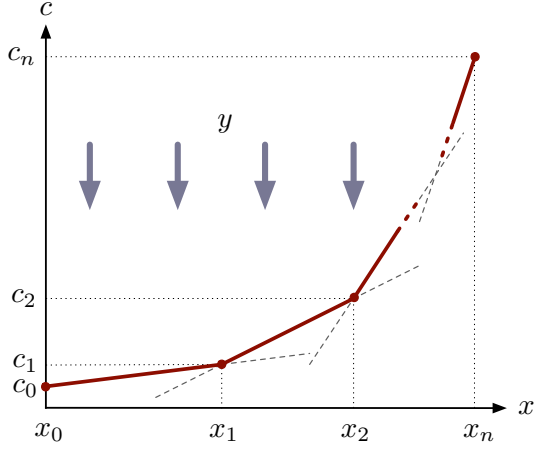
Fig. 3. Constrained Cost Variable

Fig. 4. Marginal Benefit or Bid Function

Fig. 5. Total Cost Function for Negative Injection

defined by a sequence of points (x_j, c_j), $j = 0 \ldots n$, where m_j denotes the slope of the j-th segment,

$$m_j = \frac{c_j - c_{j-1}}{x_j - x_{j-1}}, \quad j = 1 \ldots n \quad (34)$$

and $x_0 < x_1 < \cdots < x_n$ and $m_1 \leq m_2 \leq \cdots \leq m_n$.

The "basin" corresponding to this cost function is formed by the following n constraints on the helper cost variable y.

$$y \geq m_j(x - x_j) + c_j, \quad j = 1 \ldots n \quad (35)$$

The cost term added to the objective function in place of $c(x)$ is simply the variable y.

MATPOWER uses this CCV approach internally to automatically convert any piecewise linear costs on real or reactive generation into the appropriate helper variable and corresponding set of constraints. All of MATPOWER's OPF solvers use the CCV approach with the exception of two that are part of the optional TSPOPF package [5], namely the step-controlled primal/dual interior point method (SCPDIPM) and the trust region based augmented Lagrangian method (TRALM), both of which use a cost smoothing technique instead [6].

B. Dispatchable Loads

A simple approach to dispatchable or price-sensitive loads is to model them as negative real power injections with associated costs. This can be done by specifying a generator with a negative output, ranging from a minimum injection equal to the negative of the largest possible load to a maximum injection of zero.

Consider the example of a price-sensitive load whose marginal benefit function is shown in Fig. 4. The demand p_d of this load will be zero for prices above λ_1, p_1 for prices between λ_1 and λ_2, and $p_1 + p_2$ for prices below λ_2.

This corresponds to a negative generator with the piecewise linear cost curve shown in Fig. 5. Note that this approach assumes that the demand blocks can be partially dispatched or "split". Requiring blocks to be accepted or rejected in their entirety would pose a mixed-integer problem that is beyond the scope of the current MATPOWER implementation.
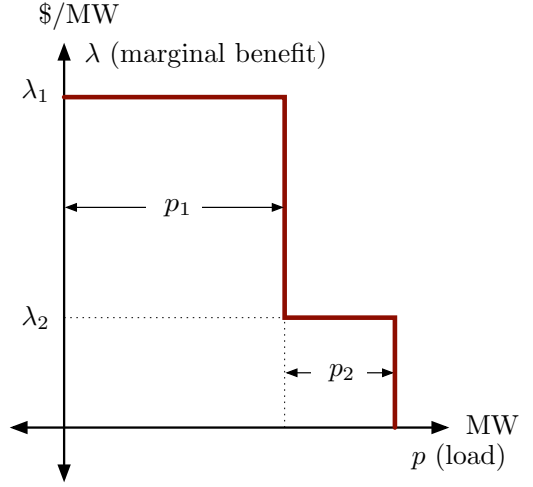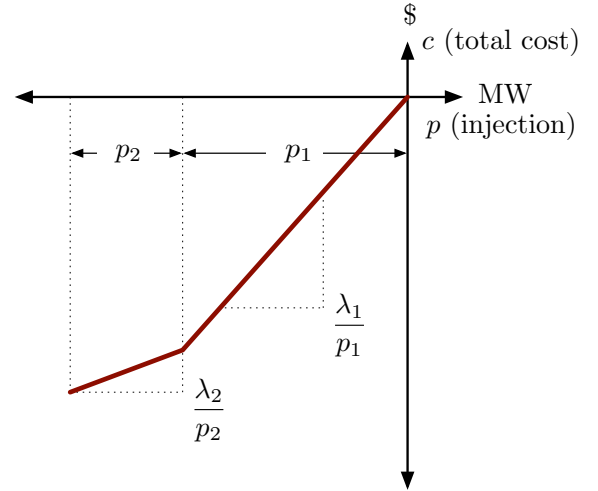
With an AC network model, there is also the question of reactive dispatch for such loads. Typically the reactive injection for a generator is allowed to take on any value within its defined limits. Since this is not normal load behavior, the model used in MATPOWER assumes that dispatchable loads maintain a constant power factor. When formulating the AC OPF problem, MATPOWER will automatically generate an additional equality constraint to enforce a constant power factor for any "negative generator" being used to model a dispatchable load.

C. Generator Capability Curves

The typical AC OPF formulation includes box constraints on a generator's real and reactive injections, specified as simple lower and upper bounds on p ($p_{\min}$ and $p_{\max}$) and q ($q_{\min}$ and $q_{\max}$). On the other hand, the true P-Q capability curves of physical generators usually involve some tradeoff between real and reactive capability, so that it is not possible to produce
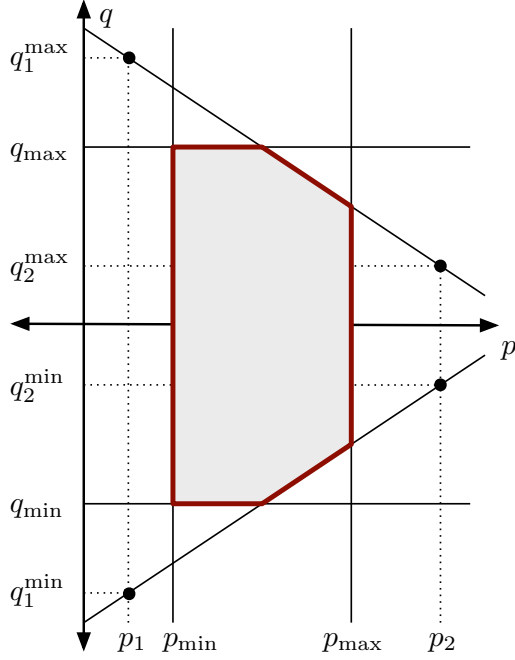
Fig. 6. Generator P-Q Capability Curve

TABLE I
VARIABLE SETS

name	description
Va	bus voltage angles
Vm	bus voltage magnitudes[a]
Pg	generator real power injection
Qg	generator reactive power injection[a]
y	CCV helper variables for piecewise linear cost

[a] AC formulation only

The former is based on MINOS [11] and the latter includes the primal-dual interior point methods and a trust region based augmented Lagrangian method described in [6].

VI. SOFTWARE ARCHITECTURE

Building up the matrices and vectors needed to add variables, costs and constraints to the OPF problem can involve a lot of messy indexing code with offsets that depend on the dimensions of the various portions of the optimization vector, existing constraints, etc. In order to ease this process as much as possible, MATPOWER utilizes an objected-oriented approach to set up and modify the problem formulation. An "OPF-model" (OM) object is constructed to encapsulate the data defining the problem. This object maintains all of the relevant indexing information for named blocks of variables, constraints and costs as they are added.

A. Variables

Adding a block of elements to the optimization vector is done via the OM object's add_vars method, providing a name for the block, the number of elements and optional vectors of initial values and variable bounds. For example, the following code can be used to define a portion of the optimization vector to represent generator real power injections where ng is the number of generators and Pg, Pmin, and Pmax are vectors of initial values, lower and upper bounds, respectively.

```
om = add_vars(om, 'Pg', ng, Pg, Pmin, Pmax);
```

The name Pg can then be used for direct access to this portion of the optimization variable or the corresponding shadow prices on variable bounds without having to keep track of the indexing explicitly.

MATPOWER uses this method internally to set up the various components of the standard optimization vector x as well as the portions of the additional variables z defined by the standard extensions from Section IV. Specifically, this includes the CCV helper variables y needed for any piecewise linear generator costs. These blocks are summarized in TABLE I.

B. Constraints

Similarly, the constraints are constructed by adding named blocks to the OM object with the add_constraints method. The standard non-linear constraints simply supply a name and the number of constraints. Linear constraints additionally require the matrix A and lower and upper bound vectors l and u that define the constraint as in (26), along with an optional ordered list of names of the variable blocks covered by the columns of the A matrix.

the maximum real output and the maximum (or minimum) reactive output simultaneously. To approximate this tradeoff, MATPOWER includes the ability to add an upper and lower sloped portion to the standard box constraints as illustrated in Fig. 6, where the shaded portion represents the feasible operating region for the unit.

The two sloped portions are constructed from the lines passing through the two pairs of points defined by the six parameters p_1, $q_1^{\min}$, $q_1^{\max}$, p_2, $q_2^{\min}$, and $q_2^{\max}$. If these six parameters are specified for a given generator, MATPOWER automatically constructs the corresponding additional linear inequality constraints on p and q for that unit.

D. Branch Angle Difference Limits

The difference between the bus voltage angle θ_f at the *from* end of a branch and the angle θ_t at the *to* end can be bounded above and below to act as a proxy for a transient stability limit, for example. If these limits are provided, MATPOWER creates the corresponding constraints on the voltage angle variables.

V. SOLVERS

The default OPF solvers included with the MATPOWER distribution, based on Matlab's Optimization Toolbox [7], work reasonably well for very small systems. For larger networks, there are a number of additional solvers available that are distributed separately as optional packages due to differences in terms of use.

For DC optimal power flow, there is a MEX build [8] of the high performance BPMPD solver [9] for LP/QP problems. For the AC OPF problem, the MINOPF [10] and TSPOPF [5] packages provide solvers suitable for much larger systems.

TABLE II
CONSTRAINT SETS

name	description
Pmis	nodal real power mismatch
Qmis	nodal reactive power mismatch[a]
Sf	limit on branch injections at *from* end[a]
St	limit on branch injections at *to* end[a]
Pf	limit on branch injections at *from* end[b]
Pt	limit on branch injections at *to* end[b]
PQh	upper portion of generator PQ capability curves[a]
PQl	lower portion of generator PQ capability curves[a]
vl	constant power factor for variable loads[a]
ang	branch angle difference limits
ycon	basin constraints for CCV helper variables

[a] AC formulation only
[b] DC formulation only

Specifying the variables covered by the columns of A eliminates the need to build a constraint matrix with explicit zero blocks corresponding to variables that do not even enter the constraint being defined. In this way, new variables can be added, along with relevant constraints, even after some constraints are already added.

For example, the DC power mismatch equations, are added with the following code.

```
om = add_constraints(om, 'Pmis', ...
     Amis, bmis, bmis, {'Va', 'Pg'});
```

The constraint matrix `Amis` includes only columns corresponding to voltage angles `Va` and generator injections `Pg`, regardless of whether or not more variables may be added later, such as the y variables for piecewise linear costs. In this example, the upper and lower bounds are both set to `bmis`, implying an equality constraint.

The `add_constraints` method is used internally by MATPOWER to register the standard OPF constraints of (2) and (3) as well as the additional linear constraints of (26) associated with the standard extensions from Section IV. TABLE II lists the constraint blocks added automatically by MATPOWER. In the AC formulation, the first four sets of constraints are nonlinear. The DC case omits all constraints involving reactive power and replaces the non-linear real power mismatch and flow limits with linear equivalents.

C. Costs

A similar `add_costs` method is provided for defining additional cost terms, where the arguments are the name of the block, a struct containing the cost parameters and an optional list of the columns covered by the supplied N matrix.

D. User Extensions

There are two methods available to the user for modifying the OPF formulation. The first is to provide directly a set of constraint parameters (A, l, and u) and/or cost parameters (H, C, N, $\hat{r}$, k, m and d). In this case, any new z variables are created implicitly based on the number of columns in A or N relative to the dimension of x.

The second, and recommended, method involves using the `add_vars`, `add_constraints` and `add_costs` methods to modify the problem. After MATPOWER constructs the OM

TABLE III
VARIABLES USED BY IMPLEMENTATION OF OPF WITH RESERVES

name	description
om	OPF model object, already includes standard OPF setup
ng	number of generators
R	name for new reserve variable vector
Rmin	lower bound on R, all zeros
Rmax	upper bound on R, based on ramp rates
Pmax	capacity of generators
I	identity matrix ($n_g \times n_g$)
Az	zone definitions, Az(i,j)=1, iff gen j lies in zone i
Rreq	vector of reserve requirements for each zone
Rcost	cost coefficients for R

object for the basic OPF problem with the standard extensions described in section IV, the user is given access to the OM object via an optional callback function to add any additional variables, constraints and costs.

This structure allows the problem to be constructed in a modular fashion, with some variables, constraints and costs handled up front by MATPOWER and others added explicitly by the user. The full set of optimization variables (x and z) and the final values of the constraint parameters from (26) (A, l, and u) and cost parameters from (28)-(32) (H, C, N, $\hat{r}$, k, m and d) are assembled by the OM object just before calling the OPF solver.

VII. EXAMPLE APPLICATION OF USER EXTENSIONS

As an example of a modified OPF formulation that employs additional variables, constraints and costs, consider the problem of jointly optimizing the allocation of both energy and reserves, where the reserve requirements are defined as a set of fixed zonal MW quantities. Let U be the set of indices of all generators providing reserves, Z_k be the set of generators in zone k and R_k be the MW reserve requirement for zone k.

A new variable r_i is introduced for each $i \in U$, to represent the reserves provided by generator i. This value must be positive and is limited above, based on ramp rate, by $r_i^{\max}$.

$$0 \leq r_i \leq r_i^{\max} \tag{36}$$

If the marginal cost of reserve from unit i is c_i, the user defined cost term from (22) is simply

$$f_u(x,z) = \sum_{i \in U} c_i r_i. \tag{37}$$

There are two additional sets of constraints needed. The first ensures that, for each generator, the total amount of energy plus reserve provided does not exceed the capacity of the unit.

$$p_i + r_i \leq p_i^{\max}, \quad \forall i \in U \tag{38}$$

The second requires that the sum of the reserve allocated within each zone k meets the stated requirements.

$$\sum_{i \in Z_k} r_i \geq R_k, \quad \forall k \tag{39}$$

This problem clearly fits the form of the extensible OPF formulation in (22)-(27). Consider the slightly simplified case where all n_g generators are available to provide reserves. Using the variables described in TABLE III, the standard OPF

problem formulation can be modified to include fixed reserve requirements with a few simple lines of code in a callback function.

```
Ar = [I I];
om = add_vars(om, 'R', ng, [], Rmin, Rmax);
om = add_constraints(om, 'Pg_plus_R', ...
        Ar, [], Pmax, {'Pg', 'R'});
om = add_constraints(om, 'Rreq', ...
        Az, Rreq, [], {'R'});
om = add_costs(om, 'Rcost', ...
        struct('N',I,'Cw',Rcost), {'R'});
```

As an example of how to take advantage of its extensible OPF capabilities, MATPOWER includes a more complete implementation of this problem that takes into account off-line and non-reserve generators. Several more complex applications utilizing this extensibility can also be found in [12]–[14].

VIII. CONCLUSION

In power systems research and education, a flexible simulation tool for power flow and optimal power flow is a valuable, and often essential, tool. Especially in research, it is frequently necessary to explore modified versions of the standard problem formulations. Unfortunately, such modifications often require significant effort in custom coding.

MATPOWER is an open-source package of Matlab tools for electric power system simulation that attempts to address this need by providing an extensible architecture for optimal power flow problems, allowing the user to easily modify and augment the problem formulation with additional variables, constraints and costs. Since this customization is done via optional input parameters, it is still possible to take advantage of the performance benefits of pre-compiled solvers. The object-oriented software architecture facilitates a modular definition of the problem, allowing additions to be made at various stages, with the final constraint and cost parameters constructed automatically just before calling the solver.

An enhanced optimal power flow formulation that jointly co-optimizes energy and reserves is used to illustrate the capabilities of MATPOWER's extensible architecture. It uses the standard OPF formulation with an additional fixed zonal reserve requirement.

REFERENCES

[1] R. D. Zimmerman and C. Murillo-Sánchez. MATPOWER User's Manual, [Online]. Available: http://www.pserc.cornell.edu/matpower/
[2] R. D. Zimmerman, R. J. Thomas, D. Gan, and C. Murillo-Sánchez, "A web-based platform for experimental investigation of electric power auctions," *Decision Support Systems*, vol. 24, no. 3-4, pp. 193–205, 1999.
[3] R. D. Zimmerman and R. J. Thomas, "PowerWeb: a tool for evaluating economic and reliability impacts of electric power market designs," *Power Systems Conference and Exposition, 2004. IEEE PES*, pp. 1562–1567 vol.3, October 10-13 2004.
[4] F. Milano, "An open source power system analysis toolbox," *Power Systems, IEEE Transactions on*, vol. 20, no. 3, pp. 1199–1206, Aug. 2005.
[5] TSPOPF: [Online]. Available: http://www.pserc.cornell.edu/tspopf/
[6] H. Wang, C. E. Murillo-Sánchez, R. D. Zimmerman, and R. J. Thomas, "On computational issues of market-based optimal power flow," *Power Systems, IEEE Transactions on*, vol. 22, no. 3, pp. 1185–1193, August 2007.
[7] *Optimization Toolbox 4 Users's Guide*. The MathWorks, Inc., 2008. [Online]. Available: http://www.mathworks.com/access/helpdesk/help/pdf_doc/optim/optim_tb.pdf
[8] BPMPD_MEX: [Online]. Available: http://www.pserc.cornell.edu/bpmpd/
[9] C. Mészáros, "The efficient implementation of interior point methods for linear programming and their applications," Ph.D. dissertation, Eötvös Loránd University of Sciences, 1996.
[10] MINOPF: [Online]. Available: http://www.pserc.cornell.edu/minopf/
[11] B. A. Murtagh and M. A. Saunders, *MINOS 5.5 User's Guide*, Stanford University Systems Optimization Laboratory Technical Report SOL83-20R.
[12] J. Chen, J. Thorp, and T. Mount, "Coordinated interchange scheduling and opportunity cost payment: a market proposal to seams issues," *System Sciences, 2004. Proceedings of the 37th Annual Hawaii International Conference on*, Jan. 2004.
[13] J. Chen, T. D. Mount, J. S. Thorp, and R. J. Thomas, "Location-based scheduling and pricing for energy and reserves: a responsive reserve market proposal," *Decis. Support Syst.*, vol. 40, no. 3-4, pp. 563–577, 2005.
[14] R. Thomas, C. Murillo-Sánchez, and R. Zimmerman, "An advanced security constrained opf that produces correct market-based pricing," *Power and Energy Society General Meeting - Conversion and Delivery of Electrical Energy in the 21st Century, 2008 IEEE*, pp. 1–6, July 2008.

Ray Daniel Zimmerman is a senior research associate in electrical engineering and applied economics and management at Cornell University. He is the lead developer of the PowerWeb electricity market simulation platform and the MATPOWER power system simulation software. His current research interests center on the interactions between the economic and engineering aspects of electric power system operations and planning. Other interests include software tools for education and research.

Carlos Edmundo Murillo-Sánchez (Ph.D. electrical engr. Cornell '99, M.Sc. electrical engr. UW-Madison '91, electronics engineering ITESM, México, '87) is Professor and Dean of Engineering at Universidad Autónoma de Manizales, and Professor at Universidad Nacional de Colombia in Manizales, Colombia. He is a founding member of the Colombian Automation Society (Asociación Colombiana de Automática). His interests include power systems operation and control, control systems applications, optimization, simulation and mechatronics.

Robert John Thomas currently holds the position of Professor of Electrical Engineering at Cornell University. He has held sabbatical positions with the U.S. Department of Energy Office of Electric Energy Systems (EES) in Washington, D.C. and at the National Science Foundation as the first Program Director for the Power Systems Program in the Engineering Directorate's Division of Electrical Systems Engineering. He has been a consultant to several entities on various electric utility system related issues. He has authored over 90 technical papers. He has been a member of the IEEE United States Activity Board's Energy Policy Committee since 1991. He is a member of the IEEE Technology Policy Council and has been a member of several university, government and industry advisory Boards or Panels. His current technical research interests are broadly in the areas of analysis and control of nonlinear continuous and discrete time systems with applications to large-scale electric power systems. He is a member of Tau Beta Pi, Eta Kappa Nu, Sigma Xi, ASEE and a Fellow of the IEEE. He has been the Director of the National Science Foundation Industry/University Cooperative Research Center, PSERC, a Center focused on problems of restructuring of the electric power industry.