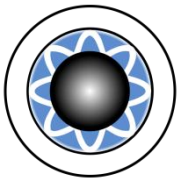
International Energy Workshop 2010
21-23 June 2010, Stockholm, Sweden



The TEMOA Project:
Tools for Energy Model Optimization and Analysis

Joseph DeCarolis, Kevin Hunter, Sarat Sreepathi
Department of Civil, Construction, and Environmental Engineering
North Carolina State University
Raleigh, NC

Abstract
The trend toward increasing complexity of energy economy optimization models gives rise to two critical issues: (1) in the absence of publicly available source code and data, model results cannot be externally verified by third parties, and (2) larger, more complicated models are harder to iterate, making it difficult to perform sensitivity and uncertainty analysis. To help address these issues, we introduce a new modeling framework called Tools for Energy Model Optimization and Analysis (TEMOA). The TEMOA project is utilizing expertise in energy systems, operations research, and computer science to create an open source energy system model with the capability to perform rigorous uncertainty analysis. This paper describes the design philosophy behind the TEMOA project. Our approach to model development is motivated by a desire to redress the issues outlined above and is informed by successful open source scientific computing projects. Once initial development is complete, TEMOA will be a technology explicit, partial equilibrium energy system model. It is being developed using Python Optimization Modeling Objects and will utilize a relational database to store and query both input and output data. Both model source code and data will be available online via a publicly accessible electronic repository. In addition, the repository will store specific snapshots of code and data, allowing third parties to verify published results in the literature. Capabilities will include parametric sensitivity analysis, Monte Carlo simulation, multi-stage stochastic optimization, and modeling-to-generate-alternatives. Once fully developed, the TEMOA project will provide valuable insight to energy and environment planning that is more robust to large future uncertainties.

1. Introduction

Energy economy optimization (EEO) models ― which optimize energy system development over multiple decades ― have emerged as key tools for the analysis of energy and climate policy at the regional, national, and international scale. Over the past two decades, Moore's Law coupled with the increasing availability of energy and environmental data has catalyzed the development of increasingly complex EEO models. Adding detail and sophistication to such models ostensibly improves the modeler's ability to create more accurate projections of energy and emissions. However, increasing model complexity highlights two critical issues associated with EEO model development and application. First, many models cannot be externally verified by third parties because the source code and input data are not publicly available. Second, multi-decadal model time horizons make it impossible to meaningfully benchmark model results against realized outcomes. Without a strong form of model validation, it is very difficult to assess whether model enhancements lead to more accurate projections. The inability to validate results reinforces the trend toward increasing model complexity and produces a concomitant reduction in the ability to perform uncertainty analysis with models that require significant tuning and computational resources. Insights generated with EEO models should—to the degree possible—be robust to large future uncertainties. If not, they are of questionable value to policy planners and decision makers.

With a couple exceptions, EEO models cannot be externally verified because the source code and data are not open source[1]. While there have been rigorous efforts to compare model results (e.g., Stanford Energy Modeling Forum, Innovative Modelling Comparison Project), the lack of access to source code prevents a deeper level of external verification. Because EEO models necessarily have long timeframes, expansive system boundaries, and encompass both physical and social phenomena, the level of descriptive detail that can be provided in model documentation and in peer-reviewed journals is insufficient to reproduce a specific set of published results. Making executable models available without source code is a step in the right direction, but it still does not allow the user to investigate how the underlying system of

[1] Notable exceptions include the DICE model, http://nordhaus.econ.yale.edu/DICE2007.htm and more recently, the OSEMOSYS, http://osmosys.yolasite.com/. Though these models are quite different in scope, approach, and focus, we consider both to be forms of EEO models.

equations and data lead to specific outcomes. As noted by Dubois (2002), replication and verification of large scientific models can only be achieved if the source code is available for scrutiny. We posit several reasons why most EEO models and datasets remain closed source: protection of intellectual property, fear of misuse by uninformed end users, inability to control or limit model analyses, implicit commitment to provide support to users, overhead associated with maintenance, and unease about subjecting code and data to public scrutiny. Despite these concerns, the problem of EEO model verification can be solved by making source code and data publicly available.

Treatment of uncertainty in EEO models is often absent or cursory. Much effort is expended trying to build larger, more complex models that endogenize observed phenomena. The resultant size and complexity of many EEO models makes it difficult to perform uncertainty analysis, which requires additional computational resources in order to enable model iteration. Given the multi-decadal timescales associated with EEO models, there is no practical way to validate model results through comparison to observed phenomena. The inability to validate results implies that modelers have little ability to assess whether particular model features improve model performance. As a result, increasing complex models are often used to produce a few highly detailed, illustrative scenarios that contribute relatively little insight about alternative ways to structure and analyze the system under consideration (Morgan and Henrion, 1990). The poor performance associated with past efforts to predict future energy outcomes supports this assertion (Craig et al., 2002). Model time horizons on the order of 50 to 100 years necessitate a rigorous exploration of the decision space to ensure that model insights are robust to large future uncertainties. While there is no practical solution to the validation problem, modelers should be cognizant of creep in EEO model complexity. While judgments about the appropriate level of model detail and sophistication are subjective and should be tailored to specific research objectives, the ability to conduct uncertainty analysis should not be compromised.

To help address these issues, we introduce a new modeling framework called Tools for Energy Model Optimization and Analysis (TEMOA).[2] The TEMOA project is utilizing expertise in energy systems, operations research, and computer science to create an open source energy system model with the capability to perform rigorous uncertainty analysis. The core activity will

[2] Temoa also means "to seek" in the Nehautl (Aztec) language (Karttunen, 1992).

be the development of the energy model itself, and existing open source tools will be utilized to provide needed functionality. Initial efforts are currently focused on the development of a technology explicit, partial equilibrium energy system model. While some aspects of the TEMOA project are shared with other modeling efforts, we believe that the following set of project characteristics is unique within the energy modeling community:

- Open source code
- Open source data
- No commercial software dependencies
- Input and output data managed directly with a relational database
- Data and code stored in a web accessible electronic repository
- Programming environment with links to linear, mixed integer, and non-linear solvers
- Built-in capability for sensitivity and uncertainty analysis

The purpose of this paper is to describe the design philosophy behind the TEMOA project. Our approach to model development is motivated by a desire to redress the issues of verification and uncertainty analysis discussed above and is informed by successful open source scientific computing projects. Section 2 provides an overview of the model and describes the objective function formulation. Sections 3 and 4 describe the modeling environment and data management system for TEMOA, respectively. The optimization tools that TEMOA can utilize are outlined in Section 5. Section 6 describes existing open source development and project management tools being utilized by the TEMOA project. While many of these tools are employed frequently in open source software projects, they may be unfamiliar within the energy modeling community. Section 7 provides a summary and outlines future work with TEMOA.

2. **Model Formulation**

Once the initial formulation is completed, the TEMOA model will serve as a technology explicit, partial equilibrium energy system model. The formulation below is strongly influenced by the well-documented MARKAL / TIMES model generators (Loulou et al, 2004; Loulou et al, 2005). The model minimizes the present system-wide cost of energy supply by optimizing technology capacity and commodity flows to meet end use demands. Technologies are divided into three

basic categories: resource supply, intermediate transformations, and demand technologies. See Figure 1.

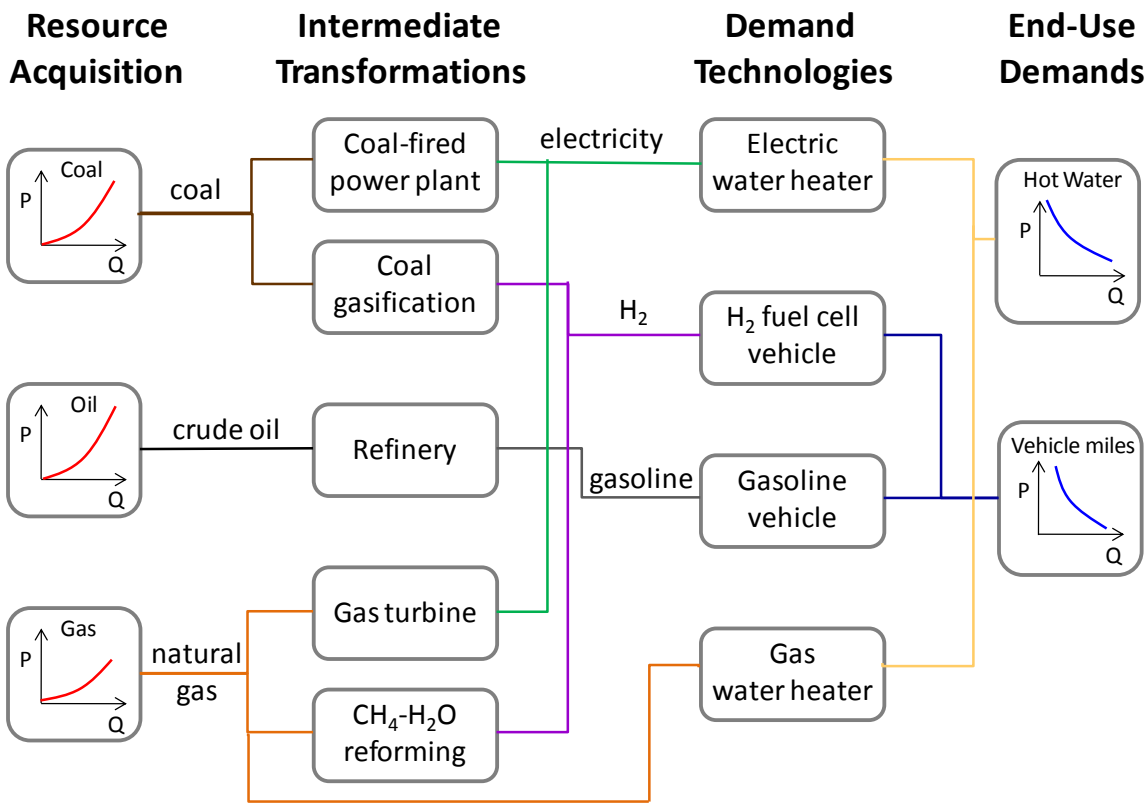


Figure 1. Schematic illustration of a simple energy system. Technologies are linked to one another via commodity flows. Moving left to right, commodities are transformed from raw materials to useful energy carriers that serve end use demands.

Energy prices and flows are determined endogenously such that end use demands are matched exactly by energy supply. The intersection of the inverse demand curve and supply curve represents equilibrium, and this equilibrium is calculated for every commodity in the energy system. Exogenously specified fixed demands will be used initially in TEMOA and represent a vertical demand curve. In this case, the equilibria are not responsive to changes in demand based on commodity prices. Elasticities can be specified in order to make demand responsive to price, but in either case, the model still represents a partial equilibrium on energy markets because it does not take into account broader economic effects and the associated feedbacks outside of the energy system. For a detailed explanation of the economic rationale behind partial equilibrium models, see Loulou et al. (2004).

The TEMOA objective function is provided below and represents the present cost of supplying energy over the model time horizon (C_{tot}):

$$C_{tot} = \sum_{tec}\sum_{per}\sum_{iper}\sum_{y=0}^{t(per+1)-t(per)} \left(\begin{array}{c} \left\{ C_i(tec,iper) \cdot \left[\frac{r_i(tec)}{1-(1+r_i(tec))^{-r_l(tec)}} \right] \cdot imat(tec,iper,per) + C_f(tec,iper,per) \right\} \\ \cdot \mathbf{x\_cap}(tec,iper) \\ + \\ C_m(tec,iper,per) \cdot vmat(tec,iper,per) \cdot \mathbf{x\_util}(tec,iper,per) \end{array} \right) \cdot \frac{1}{(1+r_g)^{t(per)+y-t(per0)}}$$

The three key sets used in the model formulation are the model time period (*per*), technology investment period (*iper*), and technologies (*tec*). All investment costs (C_i) are first annualized to avoid end effects in the model and the need to specify salvage values. Other parameters include the technology-specific marginal cost (C_m), the technology-specific interest rate on investment (r_i), the technology-specific loan period (r_l), and the global discount rate (r_g). The decision variables ***x_cap*** and ***x_util*** represent technology capacity and utilization, respectively. The two three-dimensional parameters *imat* and *vmat* are binary matrices to track the investment period and lifetime by technology vintage. Useful features of this objective function formulation include:

- Separate specifications for the loan period and lifetime of each technology
- Specification of a technology-specific interest rate on investment and a global discount rate to bring future costs back to the present
- Automatic tracking of fixed and marginal costs by technology vintage
- Model time periods can be different lengths
-

While many constraints are required to formulate the TEMOA model, three key constraints sets are listed below:

- A technology-level constraint to ensure that the sum of all commodity inputs are greater than or equal to all commodity outputs.
- A technology-level constraint to ensure that actual production cannot exceed the installed capacity. These constraints ensure that commodity production is physically tied to technology capacity.

- A set of constraints to enforce that all end use demands are met by production from appropriate demand technologies.

For more details on model formulation, see the documentation available at: http://temoaproject.org/

3. Modeling Environment

One of the goals of the TEMOA project is to attract users and developers, so the choice of programming language is critical. We have chosen to use the Python Optimization Modeling Objects (Pyomo) package, which is built in the Python language and strongly influenced by the design of AMPL (Hart, 2009). Pyomo provides a capability that is commonly associated with algebraic modeling languages (AMLs) such as AMPL, AIMMS, and GAMS; however, the Pyomo modeling objects are also embedded within a full-featured high-level programming language with a rich set of supporting libraries (Sandia, 2010). Pyomo leverages the Python component architecture to support extensibility in a modular manner. Modeling in a high level language allows modelers to utilize modern programming constructs, ensure cross-platform portability, and access the broad range of functionality found in standard software libraries. The benefits of using Python are twofold: (1) it serves as a robust, well-tested foundation for model development and (2) extensions generally require new classes or routines rather than changes to the language itself (Hart, 2009). Given the popularity of Python, there is a rich set of libraries (e.g., numpy scipy, matplotlib) that cover nearly every task.

While Pyomo provides a powerful programming environment, it lacks the compact syntax of pure AMLs, which were specifically designed to formulate of optimization problems. Despite its verbosity, Pyomo is intuitive and uses the same model structure (i.e., sets, parameters, variables, equations) as AMLs. The simple example below illustrates the algebraic expression of a model constraint set, followed by the relevant AMPL and Pyomo syntax:

Constraint:

$$\sum_{t \in tech} production_{t,seg} = dmd_{seg}, \forall seg \in segments$$

AMPL:

```
1. s.t. elc_demand{seg in segments} : sum{t in tech[seg]} production[t] =
   dmd[seg];
```

Python:

```
1. def Elc_Demand ( seg, model ):
2.    "Constraint: Electricity production must equal demand, per segment"
3.    constraint_val = sum(
4.       model.production[t]
5.       for t in model.tech[ seg ]
6.    )
7.    return ( constraint_val == model.dmd[ seg ] )
8. model.Elc_Demand = Constraint( model.segments, rule=Elc_Demand )
```

The example above represents a constraint to ensure that electricity production is sufficient to meet demand in three segments: base, shoulder, and peak. While both the AMPL and Pyomo syntax generate the intended constraint, the AMPL version accomplishes in one line what takes Pyomo eight lines. The Python version adds a function definition (L1), a comment (L2), the constraint calculation (L3-L6), an explicit test that returns true or false from the function (L7), and finally attaches a constraint to the model object (L8). Note that L3-L7 could be combined into a single line, but we chose to use more lines in order to improve legibility.

Python encourages model commenting (e.g., line 2, omitted from AMPL version). The comment line in the Pyomo version provides useful, descriptive information to the user. While the same commenting could be done in the AMPL version, Python includes tools to parse source code and create reports involving comments, representing a powerful way to document model source code and encourage document-as-you-go procedures. The block comments following object creation can be used to dynamically generate model documentation for end users. This streamlines the documentation effort by allowing developers to focus on embedding descriptive comments in the source code, which also eliminates discrepancies that arise from maintaining

multiple forms of documentation. Such source code documentation will be supplemented by higher level model descriptions for end users who are not interested in the source code.

We made the decision to sacrifice the conciseness of an AML for the power and flexibility of a general purpose programming language. The only free and open source AML currently available is GNU MathProg, an open source subset of AMPL and part of the GNU Linear Programming Kit (Makhorin, 2008). However, Mathprog is limited to linear and mixed integer formulations, whereas Pyomo is not. And although Python is more verbose than an AML, it is widely used in the scientific community. Such broad familiarity with Python is an important consideration for a new modeling effort trying to attract developers and users from multiple disciplines.

4. **Data Management**

Model transparency depends not only on well written source code, but also on data organization and management. The prevailing method for organizing input and output data for EEO models is often a combination of spreadsheets and text files[3]. TEMOA will use a relational database, which has two distinct advantages: integrity management and declarative access. Both features rely on a user-defined description — a database schema — to determine how the data should be organized and managed. A schema is a formal definition of the multiple sets, types, and relationships of data contained in a database. A relational database enforces data integrity by verifying that all input is congruent with the description of the schema. For example, where Excel would accept any value for the capacity factor, a well-designed database would automatically refuse any input that does not fall between 0 and 1. When input data resides in text file or spreadsheets, such input errors are often only discovered through the course of time-consuming calibration exercises. In addition to enforcing data integrity, a database increases workflow efficiency by immediately identifying an error.

The database schema also enables declarative access to both model input and output data, which automates the process of data selection and transformation. Database requests are made using Structured Query Language (SQL), which employs simple and intuitive syntax. For

[3] Front/back end software for the MARKAL / TIMES models (i.e., ANSWER and VEDA) build Microsoft Access databases to store data, though the average user does not directly interact with the database.

example, to request the average system-wide CO_2 emissions from multiple runs in which the installed nuclear capacity exceeded 200 GW, a researcher might ask:

```
SELECT AVERAGE(co2_output) FROM impact_table, nuclear_table WHERE
impact_table.run = nuclear_table.run AND nuclear_table.cap > 200;
```

Such a declarative statement greatly reduces the potential for user error. The database queries are also scalable: the above request can be performed on a set of 10 or 10^6 model runs, simplifying the workflow associated with sensitivity and uncertainty analysis. In addition, complex queries that take a multi-dimensional slice through the data are easy to perform. Because declarative access eliminates the need for manual data manipulation, the researcher can spend more time performing analysis.

The power of declarative access in a database can also be used to consolidate datasets across different energy economy models. Significant effort is required to build an input dataset for a specific model, even though models of the same class (e.g., technology explicit partial equilibrium) share much of the same data. Via a thin declarative layer within the database, each project can request only the data it needs, and have it dynamically generated on demand. There are two key benefits to a consolidated database. First, it facilitates the use of multiple models, enabling inter-model validation by performing runs based on the same underlying data set. Second, duplicative effort associated with building datasets to represent the same region (as part of different projects) can be reduced. While it is overly ambitious to build a database schema that can serve all models of the same class, the TEMOA database schema is being designed in a generic way to promote its extension to other models. For example, efforts are already underway to ensure that the database schema is compatible with the OSeMOSYS model.

5. Optimization Tools

Pyomo, the modeling environment used to build the TEMOA model, is part of a larger package called the COmmon Optimization Python Repository (COOPR), which was developed at Sandia National Laboratories (Sandia, 2010). A key strength of COOPR is that models developed in Pyomo are linked to optimizers written in low-level languages (e.g., C and Fortran). This two-language approach leverages the flexibility of a high-level language for formulating optimization problems and the efficiency of low-level languages for numerical computations (Hart, 2009).

Another strategic reason for using COOPR is the ability to link TEMOA with solvers that can handle a variety of mixed integer and non-linear formulations. COOPR, in turn, is part of a larger collection of packages called A Common Repository of Optimizers (ACRO). ACRO integrates a variety of optimization software packages, including both libraries developed at Sandia National Laboratory as well as publicly available third-party libraries (Sandia, 2010). ACRO supports a variety of optimization capabilities, including:

- linear programming
- mixed-integer linear programming
- nonlinear global optimization
- derivative-free local searches
- stochastic global optimization methods
- parallel branch-and-bound
- bound-constrained derivative-based local optimization

As such, ACRO provides the capability to formulate mixed integer or non-linear models. Future TEMOA development may include a mixed integer or non-linear formulation to model endogenous technological change (Barreto, 2001; Manne and Barreto, 2004), non-linear macroeconomic production functions (e.g., Bauer et al., 2008), non-linear demand response, and use of the logit function as a market sharing algorithm (Leifman, 2006).

In addition, the Common Optimization Library INterface (COLIN) package within ACRO is a general purpose C++ optimization interface that provides a middleware layer for optimizers that facilitates the connection of solvers to black-box target applications (Sandia, 2010). Separating the applications and solvers using a well-defined and flexible interface introduces many possibilities, including the creation and use of hybrid optimizers (i.e., use of multiple optimization algorithms within a single solve). In addition, COLIN includes a sophisticated evaluation management system for scheduling and managing synchronous and asynchronous application evaluations. COLIN provides a flexible analysis driver through an eXtensible Markup Language (XML) specification and interpreter that can coordinate the construction, configuration, and execution of a wide variety of analysis tasks, from simple single-algorithm optimization to complex parallel distributed multi-algorithm hybrids (Sandia, 2010). Such capabilities facilitate the iteration of EEO models such as TEMOA in a parallel

environment. An 11-node, 88-core compute cluster assembled at NC State University specifically for energy modeling will be utilized to enable rapid model iteration.

Lastly, the COOPR package contains Python-based Stochastic Programming (PySP), a modeling and solver library for generic stochastic programming. It can currently be used to solve multi-stage stochastic linear programming problems. Modeling in PySP is accomplished by developing a deterministic, single-scenario model of the problem in Pyomo. PySP assumes discretization of the probability space such that a scenario tree can be easily specified. included The specification includes the assignment of problem variables to model time stages. Given a deterministic model, scenario tree data, and the problem instance data, PySP can write and/or solve the extensive form of the stochastic program. However, PySP can also address instances where the extensive form is too difficult, due either to the presence of integers (in any stage) or a large number of scenarios. In this case, PySP provides a generic and highly customizable scenario-based decomposition solver based on Rockafellar and Wets' Progressive Hedging algorithm, which has proved effective as a heuristic for difficult, multi-stage stochastic mixed-integer programs (Rockafellar and Wets, 1991). Current PySP development efforts are focused on large-scale parallelization (cluster-oriented parallelization is already available), generic Benders-based decomposition solvers, and effective heuristics for risk-oriented and chance-constrained stochastic programming formulations (Sandia, 2010).

Utilization of PySP by the TEMOA model is planned in the near future. Multi-stage stochastic optimization represents a more sophisticated analytic approach by embedding the probability of different outcomes within the model formulation, which yields a hedging strategy that accounts for future uncertainties. While stochastic optimization has been applied to energy system models, the computational requirements have limited analysis to relatively simple probability trees (e.g., Kanudia and Loulou, 1998). In addition, the ability of PySP to handle mixed integer formulations allows us to explore complex probability trees with an integer formulation to represent endogenous technological learning.

6. Open Source Development Tools

The TEMOA project utilizes a variety of existing open source development tools and practices. We investigated the software engineering practices employed by several large scale scientific software projects with the goal of incorporating their best practices in our workflow. Specifically, we have studied the software management practices associated with the following projects:

- MPICH2, a high performance parallel programming library (Argonne, 2010a)
- Portable, Extensible Toolkit for Scientific Computation (PETSc), a parallel differential equations solver library (Argonne, 2010b)
- Community Climate System Model (CCSM), a climate system model (UCAR, 2010)

Software configuration management (SCM) is a software engineering practice to identify the configuration of a software system at distinct points in time for the purpose of systematically tracking and controlling changes to software (Bersoff, 1997). The goal is to ensure the integrity and traceability of changes during the entire software lifecycle.

Revision control, a major component of SCM, is focused on tracking the changes made to software, documents, or other digital information. It enables multiple developers to simultaneously work on a common software component and enhance productivity by automatically integrating changes made by different people. If conflicting changes exist, the revision control tools provide a streamlined mechanism to resolve such conflicts. In scientific software projects, it is common to have different groups of people working on different prototypes that, when completed, will be integrated into the core software (usually referred to as the ‘trunk’ or ‘head’). To serve that purpose, revision control systems support branching, a technique that enables development on different branches emanating from the main repository trunk. Once the prototype development is finished, the modifications from the branch can be integrated back into the main repository (trunk). Revision control systems also allow the creation of software snapshots, called ‘releases’, which represent a well tested and clearly defined milestone in the software lifecycle.

The MPICH2 project uses Subversion (Collins-Sussman et al., 2004) for revision control whereas the PETSc project uses Mercurial (O’Sullivan, 2009). We have performed a

comparative analysis on state-of-the-art revision control systems and concluded that Subversion satisfies our requirements. Hence we deployed Subversion to track our model and data development efforts. Subversion will also allow us to build and maintain a public archive of data and source code used to produce research publications. As a result, interested parties can download specific snapshots of source code and data in order to verify the results presented in the literature.

Both MPICH2 and PETSc projects provide an extensive suite of well-documented examples to introduce the diverse capabilities of their software to the end-user. Deriving inspiration from those efforts, we are planning to develop a suite of simple test datasets that work with TEMOA to illustrate the model characteristics, demonstrate model performance, and provide illustrative results.

Many large scale software projects, including MPICH2 and PETSc, utilize automated build and testing tools for nightly testing to maintain robustness and integrity. We currently perform periodic tests to ensure that our model builds successfully. Current efforts include leveraging existing infrastructure like NSF's National Middleware Intitiative to automate model building and testing on diverse computational platforms (NMI, 2010).

Community outreach is an important component of open source development model. The developers of PETSc and MPICH2 are highly responsive to user's queries on their traffic-intensive mailing lists. We will be using mailing lists for communication among developers as well as interaction with end users. In addition to those, we are also utilizing online forums, as they are more accessible and amenable for searching.

The CCSM project provides an online source code browsing capability that makes the model even more accessible to users. In addition to user guides, the aforementioned projects also provide detailed design documentation to aid developers and advanced users to understand the rationale behind the design decisions. This kind of insight cannot be obtained just by perusing code. Hence streamlined design documentation will be presented in the TEMOA wiki.

In summary, to encourage participation by interested developers and users, we have created a website: http://temoaproject.org/ that includes the following features:

- Open access to the Subversion repository
- Wiki to provide relevant documentation to the user and developer communities

- Online code browser
- Forum for discussions among both users and developers
- Mailing list

Interested users or developers are encouraged to participate in the TEMOA project, regardless of their experience with any of the aforementioned software tools. We also welcome feedback from the community and would appreciate any bug reports or feature requests, which can be directly submitted on the TEMOA website.

7. Discussion and Future Work

This paper describes an ambitious effort to create an open source energy economy optimization model. The tools described in Section 6 have been deployed at: http://temoaproject.org/. A working version of TEMOA that optimizes the U.S. electric sector currently resides in the Subversion repository, with input and output data stored in text files. Within six months, we plan to have a working version of TEMOA that can optimize an energy system with data drawn from a relational database. In the same timeframe, we plan to utilize the Pyomo-PySP linkage to perform multi-stage stochastic optimization of the U.S. electric sector.

Returning to the issues posed in the introduction, the TEMOA project solves the verification issue by making both the source code and data publicly available in a web-based repository. In addition, versions of the source code and data used to produce publications will be archived and available for download. Other users who utilize the TEMOA model to produce published analysis will be strongly encouraged to submit their source code and data for online archiving as well. While it is not possible to validate an EEO model in the same way as a model of an observable physical process, creating a generic database schema coupled with a declarative layer that can dynamically generate input data for multiple models provides a weak form of validation by facilitating inter-model comparisons.

Models are of greatest value when designed to answer specific questions. TEMOA ─ an open source, community-driven EEO model ─ can be used to address important issues related to energy and climate policy. Key questions include:

- How do different climate policy architectures affect future energy and environmental outcomes?

- Are regional variations in energy system development significant?
- Which energy technologies should be targeted for R&D funding?
- Given future uncertainty in climate change and policy, what are robust hedging strategies for energy system development?

Because users can specify technology-specific discount rates and loan periods, TEMOA can be used to represent the perspective of individual firms or that of system planners at the regional or national level.

The general design philosophy of TEMOA is to make the model just complex enough to answer specific questions, but no more. The goal is to keep the model lightweight in order to facilitate rigorous uncertainty analysis. Once the database and model are constructed and calibrated, techniques such as sensitivity analysis, Monte Carlo simulation, multi-stage stochastic optimization, and modeling-to-generate alternatives (MGA) will be utilized to perform uncertainty analysis. These methods can be linked in series to develop robust strategies for future energy system development that account for large future uncertainties. First, sensitivity analysis can be used to estimate partial rank correlation coefficients and identify the input parameters that have the greatest effect on the key model outputs (Tschang and Dowlatabadi, 1995; Kann and Weyant, 2000). Second, a joint distribution of the key uncertain parameters can be created for use in a multistage stochastic optimization in order to develop a hedging strategy. Finally, MGA can be applied to the stochastic optimization solution to test the robustness of the optimal hedging strategy. Once this capability is fully developed, TEMOA will provide valuable insight into the planning process while including a detailed characterization of uncertainty in model outputs. In addition, the open source code and data ensure that the model can be scrutinized and specific results verified.

References

Argonne. (2010a). *MPICH2*. Argonne National Laboratory. Website: http://www.mcs.anl.gov/research/projects/mpich2/

Argonne. (2010b). Portable, *Extensible Toolkit for Scientific Computation (PETSC)*. Argonne National Laboratory. Website: http://www.mcs.anl.gov/petsc/petsc-as/

Barreto L (2001) Technological learning in energy optimisation models and the deployment of emerging technologies. PhD Thesis no 14151, Swiss Federal Institute of Technology Zurich (ETHZ). Zurich, Switzerland

Bauer N, Edenhofer O, Kypreos S (2008). "Linking energy system and macroeconomic growth models" *Computational Management Science*, 5: 95-117.

Bersoff EH. (1997). "Elements of Software Configuration Management" in Software Engineering, M. Dorfman and R.H. Thayer, eds., *IEEE Computer Society Press*.

Collins-Sussman B, Fitzpatrick BW, Pilato CM. (2004). *Version Control with Subversion, Next Generation Open Source Version Control*. O'Reilly Media, Inc.

Craig PP, Gadgil A, Koomey JG. (2002). "What Can History Teach Us? A Retrospective Examination of Long-Term Energy Forecasts for the United States" *Annual Review of Energy and the Environment*, 27: 83-118.

Dubois P. (2002). "Key techniques for open-source science" Coalition for Academic Scientific Computation (CASC) Seminar, March 4, 2002.

Hart W. (2009). "Python Optimization Modeling Objects (Pyomo)", *Operations Research and Cyber-Infrastructure*, Eds. J.W. Chinneck, B. Kristjansson, M. Saltzman, Springer, New York, 2009, pp. 3-20.

Kann A, Weyant JP. (2000). "Approaches for performing uncertainty analysis in large-scale energy/economic policy models" *Environmental Modeling and Assessment*, 5: 29-46.

Kanudia A and Loulou R. (1998). "Robust responses to climate change via stochastic MARKAL: The case of Quebec" *European Journal of Operational Research*, 106(1): 15-30.

Karttunen, F. (1992). *An analytical dictionary of Nahuatl*. University of Oklahoma Press.

Leifman M, Short W, Ferguson T. (2006). "The Stochastic Energy Deployment Systems (SEDS) Model" Presented at*: Energy and Economic Policy Models: A Reexamination of Some Fundamental Issues*, 16-17 Nov 2006.

Loulou R, Goldstein G, Noble K. (2004). *Documentation for the MARKAL Family of Models*. Energy Technology Systems Analysis Programme. Available at: http://www.etsap.org/documentation.asp

Loulou R, Remne U, Kanudia A, Lehtila A, Goldstein G. (2005). *Documentation for the TIMES Model: PART II*. Energy Technology Systems Analysis Programme. Available at: http://www.etsap.org/documentation.asp

Makhorin A. (2008). GLPK (GNU Linear Programming Kit). Webpage: http://www.gnu.org/software/glpk/

Manne AS, Barreto L. (2004). "Learn-by-doing and carbon dioxide abatement". Energy Economics, 26: 621-633.

Morgan G, Henrion M. (1990). *Uncertainty*. Cambridge University Press: New York.

NMI. (2010). *National Middleware Initiative*. National Science Foundation; University of Wisconsin-Madison Computer Sciences Dept. Website: https://nmi.cs.wisc.edu/

O'Sullivan B. (2009). Mercurial*: The Definitive Guide*. O'Reilly Media, Inc.

Rockafellar RT, Wets RJ (1991). "Scenarios and Policy Aggregation in Optimization Under Uncertainty" *Mathematics of Operations Research*, 16(1): 199-147.

Sandia. (2010). *Acro Packages*. Sandia National Laboratory. Webpage: https://software.sandia.gov/trac/acro/wiki/Packages

Tschang FT, Dowlatabadi H. (1995). "A Bayesian technique for refining uncertainty in global energy model forecasts", *International Journal of Forecasting*,11: 43-61.

UCAR. (2010).Community Climate System Model (CCSM). University Corporation for Atmospheric Research. Website: http://www.ccsm.ucar.edu/