500 MW DEMONSTRATION OF ADVANCED WALL-FIRED COMBUSTION TECHNIQUES FOR THE REDUCTION OF NITROGEN OXIDE (NOx) EMISSIONS FROM COAL-FIRED BOILERS

Phase 4C – Unit Optimization

Topical Report

DOE Cooperative Agreement
DE-FC22-90PC89651

Project Period
July 31, 1989 through April 30, 2003

Prepared by:

Charles Boohaker
John Sorge
Southern Company Services, Inc.
600 North 18th Street
P. O. Box 2625
Birmingham, AL 35291

November 2003

LEGAL NOTICE

ABSTRACT

As of September 1998, power plant optimization software had been used in more than 100 units in the United States. In most cases, the optimization has focused on boiler optimization and in particular, NOx emissions, and boiler efficiency. Although applied successfully for boiler optimization, applications of on-line optimization of other processes in a power plant were uncommon. The success of on-line boiler optimization, coupled with industry trends for competition and emissions abatement, suggested that the on-line optimization envelope should be expanded to include other power plant processes and the unit or possibly plant. As of 1998, some efforts had been made in this area but broad integrated optimization had not yet been demonstrated in the utility industry. In 1999 EPRI, Powergen, Southern Company, URS Corporation, DTI, and DOE agreed to sponsor a demonstration of on-line optimization of other processes in the power plant and coordination of these optimizers. The goal included identification of appropriate software and, if not available, development of the software, installation of the software, and testing at Hammond 4. Other participants in this project extension included Synengco Engineering and the Center for Electric Power at Tennessee Technological University. The project goals were achieved with varying degrees of success. Specifically, due to delays resulting in several project extensions, there was insufficient plant testing to quantify the benefit of the technologies. This report documents the design decisions and technologies developed through this project. The report also provides testing results as available including that from simulated testing and actual testing on the unit. Given the possible great returns by the application of these technologies, additional work is planned including improvements to the software and further testing.

ACKNOWLEDGEMENTS

The authors wish to thank DOE, DTI, EPRI, Powergen, and Southern Company for their financial support for this project and, more importantly, the thoughtful advice provided by Jim Longanbach (DOE) and Jeff Stallings (EPRI), the project managers for their respective organizations. Outstanding support from the Georgia Power Plant Hammond staff is also greatly appreciated, especially that provided by Ernie Padgett, who has been integral to the project at Hammond from the beginning, and Don Jacobs. The following individuals and their organizations invested considerable time and energy into this phase of the project and their efforts are very appreciated:

Powergen – Dave Turner and Ian Mayes

URS Corporation – Jim Noblett and George Warriner

Energy Technologies Enterprises Corporation - Stratos Tavoulareas

Tennessee Technological University – Sastry Munukutla

Southern Company Services – Mark Berry, Teresa Cunningham, Mark Faurot, Kerry Kline, Eddie Kominek, and Steve Logan

Synengco Engineering – Don Sands

Many thanks to Julia York and Rebecca Williams, for guiding us through the regulatory and contractual labyrinth, Brian Mead for data reduction and reviewing this report, and lastly and particularly to Toby Whatley for her continuous, unsurpassed support and patience.

POINTS OF CONTACT

Southern Company Services

John Sorge
Southern Company Services, Inc.
600 North 18th Street
Birmingham, AL 35202
Tel: 205.257.7426
E-mail: jnsorge@southernco.com

U.S. Department of Energy

Jim Longanbach
U.S. Department of Energy
Federal Energy Technology Center
3610 Collins Ferry Road
Morgantown, WV 26505
Tel: 304.285.4659
E-mail: jlonga@fetc.doe.gov

Powergen

David Turner
Powergen, plc.
Power Technology Centre
Ratcliffe-on-Soar
Nottingham NG11 0EE UK
Tel: +44 (0) 115 936 2371
E-mail: Dave.Turner@powertech.co.uk

Electric Power Research Institute

Jeff Stallings
Electric Power Research Institute
3412 Hillview Ave.
Palo Alto, CA 94304-1395
Tel: 650.855.2427
E-mail: jstallin@epri.com

EXECUTIVE SUMMARY

Background

Initially deployed in the mid-1990s, as of 2002, power plant optimization software has been used in more than 100 units in the United States. In almost all cases, the optimization has focused on boiler optimization and in particular, NOx emissions and boiler efficiency. Although results are site and vendor specific, NOx emission reductions of approximately 10% and efficiency improvements of more than 0.5% are not uncommon. Given the (1) relatively low cost of optimization compared to other control technologies and (2) increasing number of documented results, optimization has proven itself to be a viable alternative for NOx reductions and boiler efficiency improvements. Commercial software tools that have been successfully applied to on-line boiler optimization include *GNOCIS*, *NeuSIGHT*, *ProcessLink*, and *SmartProcess*.[1] Although there are differentiators between these products, fundamentally they are similar. In general, all utilize a neural-network or other software model of the combustion characteristics of the boiler reflecting both short-term and long-term trends in boiler operating characteristics. A constrained non-linear optimizing procedure is applied to identify the best set points for the plant. These recommended set points might be implemented automatically without operator intervention (closed-loop), or, at the plant's discretion, conveyed to the plant operators for implementation (open-loop).

Although significant cost benefits were obtained by boiler optimization alone, there was and continues to be an impetus for expanding the optimization envelope at the power plant. The impetus is the result of several factors, in particular:

- Utility deregulation and competition
- Environmental regulations and emissions trading

In the late 1990s, deregulation of the utility industry was rapidly progressing and in several

[1] *GNOCIS* was developed by EPRI, Powergen, Southern Company, UK Department of Trade and Industry, and US Department of Energy. *NeuSIGHT* is a product of Pegasus Technologies. *ProcessLink* is a product of NeuCo, Inc. *SmartProcess* is a product of Emerson Electric.

countries and US states active electricity markets were in practice. Although there have been distortions in some markets which resulted in increased cost to the customers, it is still generally held that competition has resulted in lower prices, particularly for large industrial and commercial customers. This deregulation has made many utilities refocus their attention on efficiency issues in order to maintain the historical return on equity or profitability for that unit. Also, deregulation has lead to high volatility in electricity pricing with prices varying between $20/MWh to above $7000/MWh. As a result, utilities are looking at methods to increase the flexibility of their units to respond to this volatility.

Emissions is another factor that is leading utilities to improve the performance (both environmental and thermal efficiency) of their units. Power plants continue to come under increasingly tighter environmental standards. These include the Clean Air Act Amendments of 1990, Ozone Transport, Global Warming Initiative, and CAM Regulations. NOx emission trading has become a reality for most utilities with current projected NOx emission credits ranging in cost from $4000 to $5000/ton NOx. As this price level, NOx emissions are comparable as a factor with fuel in the cost of operation of a unit.

Since the boiler is in many respects the most flexible process in a power plant and has the potential for greatest returns, on-line boiler optimization is often considered the first focus when improving the overall performance of a unit. Although applied successfully for boiler optimization, applications of on-line optimization of other processes in a power plant are uncommon. As of 1999, some efforts had been made in this area but broad integrated optimization had not yet been demonstrated in the utility industry.

Expanding the optimization envelope beyond the boiler provides additional opportunities but with the cost of additional complexity. These complications are the result of several factors including:

- Power plant processes are very highly coupled with substantial interaction between the components. An example of this interaction is shown in Figure ES-1.
- Power plants, particularly coal-fired plants, are very complex, non-linear, non-stationary processes. This is in part the result of coal supply variations but also weather and plant equipment affect the performance and emission characteristics of a unit.
- Many important process performance indicators are difficult to measure in a timely manner. An example of this is the measurement of fly ash unburned carbon content (UBC or LOI).

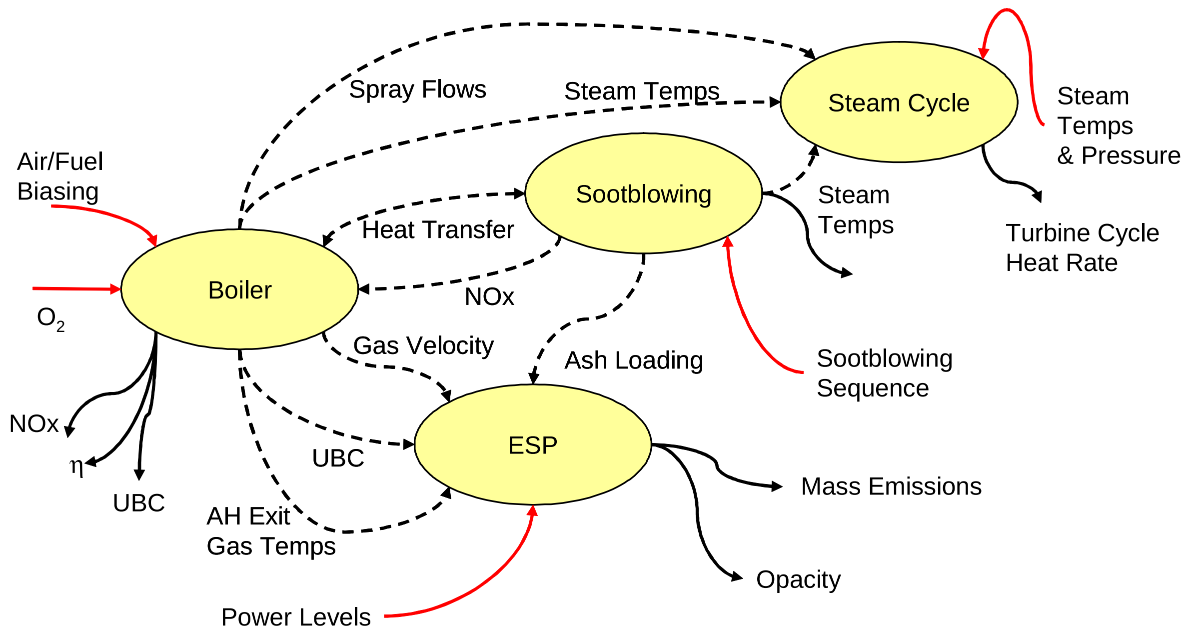


Figure ES-1 Process Interaction

Project Description and Objectives

Georgia Power Company's Plant Hammond Unit 4 has served as a host site of a DOE sponsored project demonstrating several NOx control technologies and technologies to improve the thermal performance of the unit. The technologies demonstrated have included advanced over-fire air, low NOx burners, LOI monitors, and GNOCIS. These efforts have been documented in previously published reports.

Carrying forth with prior efforts at this site, in 1999 EPRI, Powergen, Southern Company, URS Corporation, UK Department of Trade and Industry, and DOE agreed to pursue further expansion of the project to include demonstration of on-line optimization of other processes in the power plant and coordination of these optimizers. The following packages were anticipated (Figure ES-2):

- Real-Time Heat Rate Monitoring
- Boiler Optimization Package
- Steam Turbine Optimization Package
- Intelligent Sootblowing Package

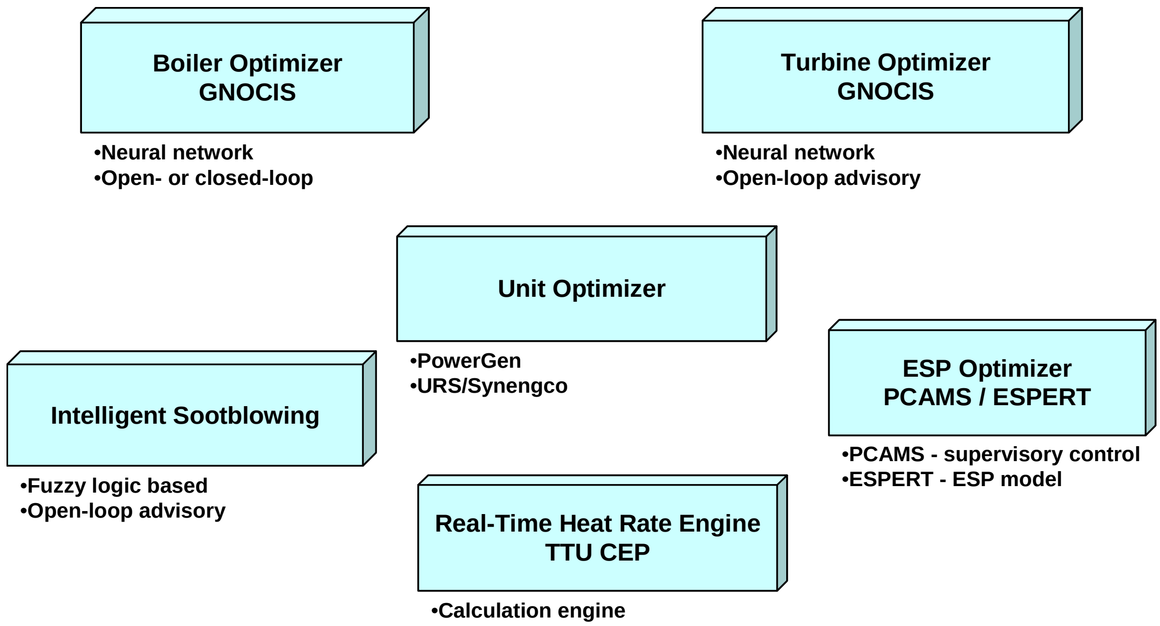


Figure ES-2 Major Components of the Project

- ESP Package
- Unit Optimizer Package

The goal included identification of appropriate software and, if not available, development of the software, installation of the software, and testing. Other participants in this project extension included Synengco Engineering and the Center for Electric Power at Tennessee Technological University.

Status and Results

The project goals were achieved with varying degrees of success. Specifically, due to delays resulting in several project extensions, there was insufficient plant testing to fully quantify the benefit of the technologies. The following is a brief description of and status of the various major components as of March 2003. A diagram showing the interrelation of the software developed is shown in Figure ES-3.

ESP Package – EPRI's ESPert was installed at the site as part of this project. The ESPert package, originally developed in the 1990s, is a diagnostic and predictive model for ESPs designed to evaluate and predict ESP performance and diagnose problems. ESPert interfaces with the PCAM system, a supervisory control system for the ESP. Initial expectations were to

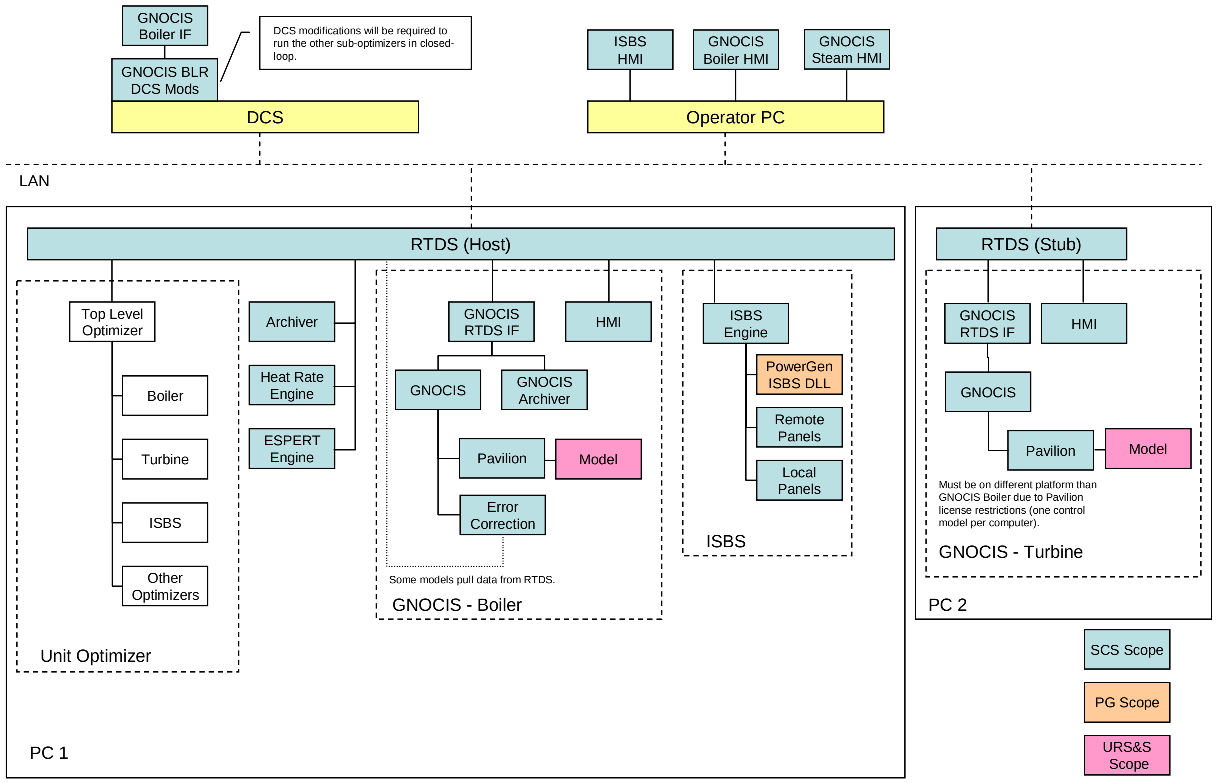


Figure ES-3 Software Architecture

use the ESPert/PCAM software as an optimization platform; however to date, it has been used only as a predictive model.

<u>GNOCIS/Boiler</u> – GNOCIS is a real-time, closed-loop system for performing boiler optimization. GNOCIS was first installed at Hammond 4 in 1996 and was upgraded as part of this current project. A major improvement was the development and incorporation of on-line model error correction. This error correction greatly improves the accuracy and robustness of the neural-network combustion models. An operator interface exists on the DCS for this component and this system is capable of both open- or closed-loop operation. The current configuration makes recommendations on excess O_2, feeder coal flows, and overfire air for optimizing NOx emissions, boiler efficiency, and fly ash unburned carbon. Previous testing of GNOCIS at this site attested substantial benefits could be obtained by its application.

<u>GNOCIS/Turbine</u> – GNOCIS was adapted to be applied to steam cycle optimization. This package uses the same code base as that used by the GNOCIS/Boiler; however, a different model (for the turbine) is used. At present, this is an advisory system only, lacking the DCS configuration modifications required to be closed-loop. Also, the operator interface runs on a local or remote PC and not on the DCS. This system is configured to make recommendations on

main steam and hot reheat temperatures and main steam pressure to optimize turbine cycle heat rate.

Intelligent Sootblowing System (ISBS) Package – The ISBS is an advisory system providing guidance on sootblower operation. Powergen developed the dynamic link library (DLL) implementing the fuzzy rule-base and SCS developed the interface and other supporting code. This package is a rule-based advisory system and not an optimizer as are the two GNOCIS packages. The user interface for the package runs on a PC, either local or remote, and not on the DCS. Brief testing of the technology indicated that the application would provide substantial benefits primarily in reducing sootblowing activity. The ISBS package is installed and available for operation at Hammond.

Real-Time Heat Rate Package - The Center for Electric Power at Tennessee Technological University developed a set of on-line unit heat rate and boiler performance calculations for the unit. SCS interfaced this package to the balance of the software system. The software was installed to provide more information concerning the real-time unit performance than previously available. Although not a primary goal of the project, plans are being made to compare the outputs of the program (heat rate, boiler efficiency, coal flow, coal higher heating value, and coal nitrogen content) to that generated by other methods.

Unit Optimization Package – The focus of this package was to develop a framework and software to coordinate multiple process optimizers. This package consists of several components including global optimizers and adaptations of the “package” optimizers (and sub-optimizers) to allow communication to the global optimizer. Although the framework and software will support other global optimizers, two were included in this scope. SCS adapted a Powergen developed proof-of-concept global optimization algorithm to fit within the framework. The other global optimizer incorporated was one developed by Synengco and marketed in the US by URS. Although functional, this software requires further testing to ensure that it is operating robustly and reliably.

Further Work

As of March 2003, other than the current project, the authors know of no other active attempts to apply coordinated optimization to power plants. As part of DOE’s recent Clean Coal Power Initiative, DOE has proposed to co-sponsor a project of similar scope with project completion in 2006. Given the possible great returns by the application of these technologies, additional work is planned including improvements to the software and further testing. Areas of work and improvement that may prove beneficial include:

- Further testing on both simulator and plant to fully quantify performance and emission benefits of applying the software.
- Refinement of software components to improve robustness and flexibility.
- Development of training and operating manuals for plant staff.
- Improvement of user interfaces for operations personnel.
- Migration of software to use the recently installed plant operations information system (ASPEN Technologies InfoPlus 21).
- Add capabilities for closed-loop operation on steam cycle optimization package.
- Install most recent version of ESPert and modify software to take advantage of limited optimization capabilities of ESPert and PCAMS.
- Complete enhancement of ISBS package so that it may be operated as an optimizer and investigate potential closed-loop operation.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

TABLE OF ABBREVIATIONS

(M)Btu	(million) British thermal unit
°C	Degrees Celsius
°F	Degrees Fahrenheit
AGC	Automatic generation control
AMIS	All mills in service
ANN	Artificial neural network
AOFA	Advanced Overfire Air
B&W	Babcock & Wilcox
C	carbon
CAA(A)	Clean Air Act (Amendments)
CAM	EPA Compliance Assurance Monitoring
CCEM	Compliance continuous emissions monitor
CCT	(US DOE) Clean Coal Technology
CEM	Continuous emissions monitor
CF	cleanliness factor
CIA	Carbon-in-Ash
Cl	chlorine
CO	carbon monoxide
CO_2	carbon dioxide
DAS	data acquisition system
DCS	distributed control system
DOE	US Department of Energy
DP	differential pressure
DPU	data processing unit
DTI	UK Department of Trade and Industry
ENTEC	Energy Technologies Enterprises Corporation
EPA	Environmental Protection Agency
EPRI	Electric Power Research Institute
ESP	electrostatic precipitator
ESPERT	EPRI precipitator diagnostic software
FC	fixed carbon
FD	forced draft
ft	feet
FTP	file transfer protocol
GNOCIS	Generic NOx Control Intelligent System
GPM	gallons per minute
H	hydrogen
HHV	higher heating value
HP	high pressure
I/O	input/output
ID	induced draft
IP	intermediate pressure
ISBS	Intelligent Sootblowing System
lb(s)	pound(s)
LH	left hand
LNB	low NO_x burner
LOI	loss-on-ignition (an estimate of CIA)
LP	low pressure
MOOS	Mills-out-of-service
MWe	megawatt (electrical)
N	nitrogen
NETL	(DOE) National Energy Technology Laboratory
NO_x	nitrogen oxides
NSPS	New Source Performance Standards
O, O_2	oxygen
OFA	overfire air
PA	primary air
PG	Powergen
PID	Proportional-Integral-Derivative
PLC	programmable logic controller
ppm	parts per million
psi	pounds per square
psig	pounds per square inch gauge
psia	pounds per square inch absolute
PTC	Performance Test Codes
RH	right hand
RPM	revolutions per minute
RSD	relative standard deviation
RTD	resistance temperature detectors
S	sulfur
SCS	Southern Company Services
SO_2	sulfur dioxide
THC	total hydrocarbons
TPCO	Total Plant Cost Optimization
TTU	Tennessee Technological University
UMS	unit master station
VAC	volts-alternating current
VDC	volts-direct current
VM	volatile matter
VWO	valves wide open
WW	water wall

1 INTRODUCTION

Purpose of this Report

This report documents the results of a project with an overall goal to demonstrate on-line optimization techniques to power plant processes and to the unit as a whole. The project was conducted at Georgia Power Company's Plant Hammond 4. Sponsors for this project include EPRI, Powergen, Southern Company, UK Department of Trade and Industry, and US Department of Energy.

Background

Initially deployed in the mid-1990s, as of 2002, power plant optimization software has been used in more than 100 units in the United States [EPR02]. In almost all cases, the optimization has focused on boiler optimization and in particular, NOx emissions and boiler efficiency. Although results are site and vendor specific, NOx emission reductions of approximately 10% and efficiency improvements of more than 0.5% are not uncommon (Figure 1-1) [EPR02].[1]

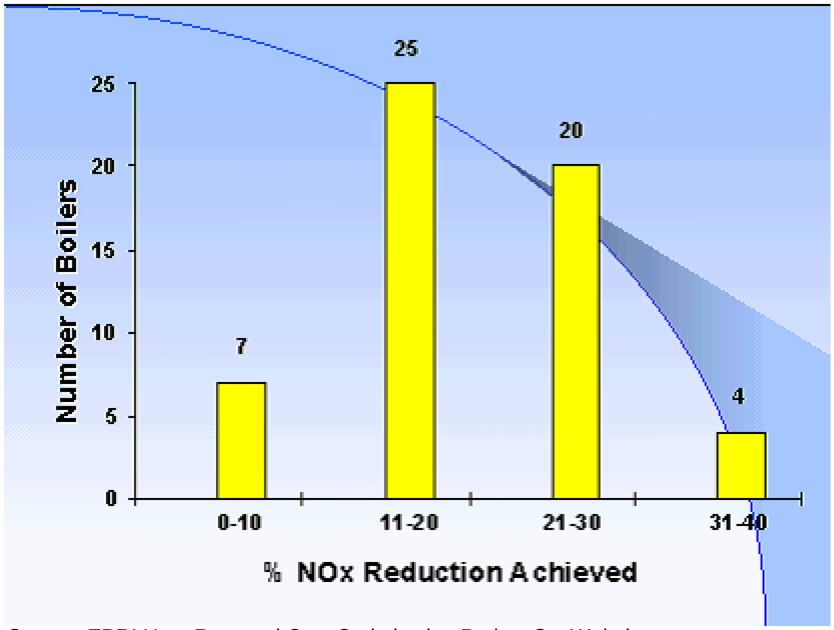
Although significant cost benefits were obtained by boiler optimization alone, there was and continues to be an impetus for expanding the optimization envelope at the power plant. The impetus is the result of several factors, in particular:

- Utility deregulation and competition
- Environmental regulations and emissions trading

In the late 1990s, deregulation of the utility industry was rapidly progressing and in several countries and US states active electricity markets were in practice. Although there have been distortions in some markets which resulted in increased cost to the customers, it is still generally held that competition has resulted in lower prices, particularly for large industrial and commercial customers. This deregulation has made many utilities refocus their attention on efficiency issues in order to maintain the historical return on equity or profitability for that unit.

[1] Some NOx reduction values used to compile this chart were provided by the technology vendors without independent verification and include values that may be aberrations from normal installations or special cases. When the extrema are excluded, the chart reasonably represents NOx reductions that may be achieved on most utility coal-fired units.

Also, deregulation has lead to high volatility in electricity pricing with prices varying between $20/MWh to above $7000/MWh. As a result, utilities are looking at methods to increase the flexibility of their units to respond to this volatility.



Source: EPRI Heat Rate and Cost Optimization Project Set Website

Figure 1-1 NOx Reduction Experience with Combustion Tuning / Optimization

Emissions are another factor that is leading utilities to improve the performance (both environmental and thermal efficiency) of their units. Power plants continue to come under increasingly tighter environmental standards. These include the Clean Air Act Amendments of 1990, Ozone Transport, Global Warming Initiative, and CAM Regulations. NOx emission trading has become a reality for most utilities with current projected NOx emission credits ranging in cost from $4000 to $5000/ton NOx. As this price level, NOx emissions are comparable as a factor with fuel in the cost of operation of a unit.

On-line optimization has the potential to provide the following:

- Displacement of higher cost NOx control technologies - Optimization techniques are very cost effective on a $/ton NOx removed basis as compared to other cost control technologies [SCS98b][EPR02]. For units with low to moderate NOx reduction requirements, optimization may eliminate the need to install a more costly technology (LNB, SCR, SNCR, etc.) [CE98][EPR97].
- Reducing the cost of SNCR and SCR - Utilizing GNOCIS or other boiler optimization packages may lead to reduced capital and operating costs for a given target stack NOx

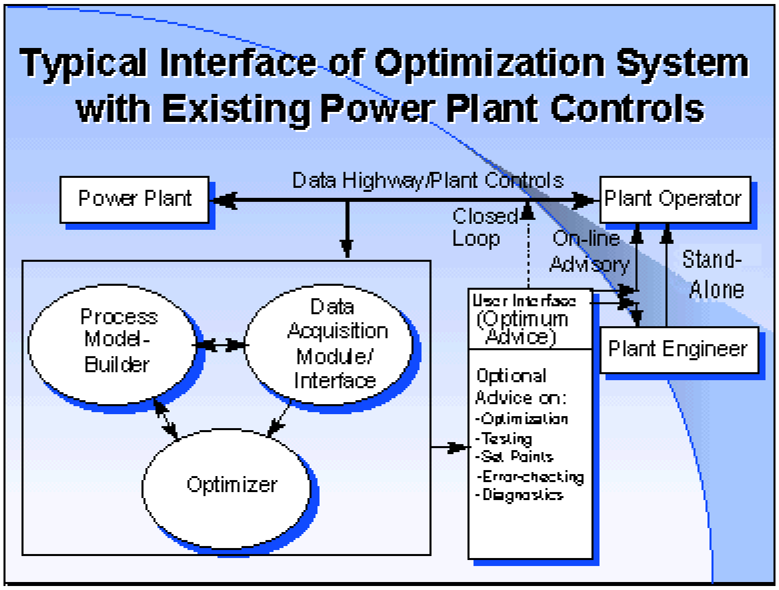
emission level. This cost reduction is the result of lower inlet NOx levels to the post combustion NOx control technology which allows the use of less reagent and/or lower capacity SNCR or SCR equipment.

- Reducing the overall cost of a NOx averaging plan - Installation of boiler optimization packages on a number of units in an averaging plan can lead to substantial aggregate NOx emission reductions. Also, when averaging, the use of boiler optimization packages may eliminate the need for more costly NOx control technologies on one or more units.
- Be a valuable tool in emissions trading - The US Clean Air Act Amendments of 1990 created a market for SO_2 credits. Similarly nascent markets are developing for NOx and CO_2 emissions. On-line optimization systems allows the utility to flexibly and quickly configure the unit to best match the emissions market while also considering unit efficiency.
- Optimize ESP Performance - ESP optimization tools (such as EPRI's ESPert) have the potential, particularly if it can be linked to boiler optimization packages (such as GNOCIS), to help plant operators optimize ESP performance to stay in compliance and hence satisfy CAM requirements while simultaneously minimizing operating cost.

During the past several years, there have been a number of on-line, continuous boiler optimization systems successfully applied. Successfully applied products include GNOCIS (Powergen/SCS/URS), NeuSight (Pegasus Technologies), ProcessLink (NeuCo), Process Insights (Pavilion Technologies), and SmartProcess (Emerson Electric/Westinghouse). Although there are differentiators between these products, fundamentally they are similar (Figure 1-2). Each of these products utilizes a neural-network model of the combustion characteristics of the boiler. A constrained-nonlinear optimizing procedure is applied to identify the best set points for the plant. The recommended set points may be implemented automatically without operator intervention using the plants distributed control system (closed-loop), or, at the plant's discretion, conveyed to the plant operators for implementation (open-loop) [EPR02].

GNOCIS was demonstrated at Hammond as part of the CCT Wall-Fired Project. The Wall-Fired Project has been the mechanism to investigate numerous technologies as to their usefulness in reducing emissions, particularly NOx, and improving thermal and combustion efficiency [SCS93][SCS97a][SCS98a][SCS98b]. Testing of GNOCIS at Hammond is consistent with results from other sites having shown that NOx reductions of approximately 15% are achievable when set as a goal. When efficiency is targeted, boiler efficiency improvements of near 0.5% and carbon-in-ash reductions of 30% are obtainable. Performance at other GNOCIS sites have been similar to that observed at Hammond [SCS97b][EPR02].

GNOCIS and other optimization packages have been in general successfully applied to the boiler [EPR02]. This success, coupled with industry trends for competition and emissions abatement, suggests that the on-line optimization envelope should be expanded to include other power plant processes and the unit or possibly the plant. Some prior efforts have been made in this area, however, this broad integrated optimization has not been demonstrated in the utility industry to date.



Source: EPRI Heat Rate and Cost Optimization Project Set Website

Figure 1-2 Optimization System Typical Arrangement

Although applied successfully for boiler optimization, applications of on-line optimization of other processes in a power plant were uncommon. The success of on-line boiler optimization, coupled with industry then trends for competition and emissions abatement, suggested that the on-line optimization envelope should be expanded to include other power plant processes and the unit or possibly plant. Carrying forth with earlier efforts at this site, in 1999 EPRI, Powergen, Southern Company, DTI, and DOE agreed to pursue further expansion of the project to include demonstration of on-line optimization of other processes in the power plant and coordination of these optimizers.

Project Description

The overall goal of the project expansion was to demonstrate on-line, optimization techniques to power plant process and to the unit as a whole. As proposed, the project consisted of the following major tasks:

- Project Management
- Optimization Design
  - Optimization Scoping Study
  - Real Time Heat Rate Package
  - Unit Optimization Package
  - Boiler Optimization Package
  - ESP Optimization / Modeling Package
  - Sootblowing Optimization Package
  - Steam Turbine Optimization Package
- Testing
- Reporting

As mentioned previously, testing to date has been limited.

Project Team

The project team consisted of the following organizations:

Energy Technologies Enterprises Corp. – EnTec was responsible for the application of EPRI's Total Plant Cost Optimization Software at Hammond.

EPRI – Funder to the project.

Georgia Power – Georgia Power Company's Plant Hammond Unit 4 served as the host site for the project.

Powergen – Powergen provided design consulting on the unit optimization and prototype optimization packages. In this effort, they provided a prototype implementation of a global optimization algorithm. They also provide the fuzzy rule base for the Intelligent Sootblowing Package.

Southern Company – Funder to the project.

Southern Company Services – SCS was responsible for overall project. SCS also provided software design for various packages including the unit optimization, GNOCIS, Intelligent Sootblowing, and RTHR packages. SCS was also responsible for DCS modifications to support the installation.

Synengco – Synengco was the provider of their Sentinent Systems optimization software.

Tennessee Technological University – The Center for Electric Power at Tennessee Technological University provided the on-line real-time heat rate software library.

UK Department of Trade and Industry – Funder to the project.

URS Corporation – URS performed process modeling for the boiler and turbine cycle optimization packages. Additionally, URS provided ad hoc consulting and support in other areas such as testing and reporting tasks.

US Department of Energy – Funder to the project.

Project Cost

The total expected cost for the project was $762,000. The participants funding contributions are shown in Figure 1-3 and Table 1-1. The DOE Clean Coal Project funding is detailed in Table 1-2. As shown, this particular extension comprised less than 5% of the total funding for the project.

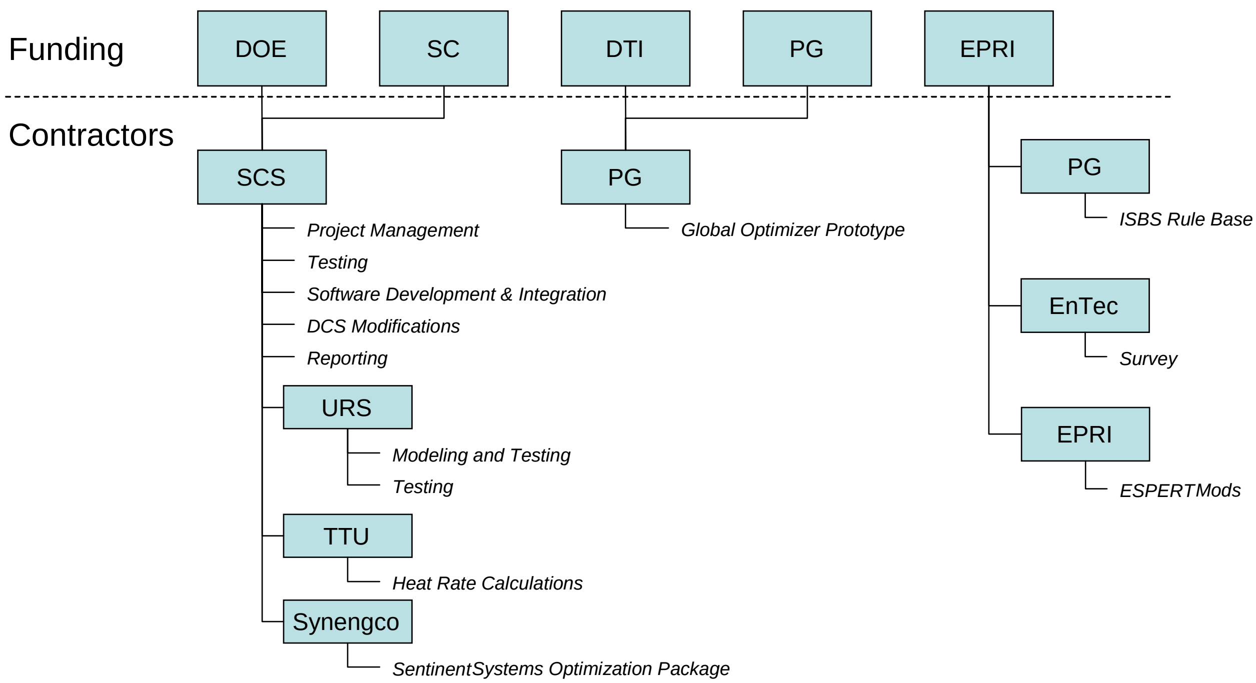


Figure 1-3 Project Funding

Table 1-1 Project Funding by Participant

Participant	Contribution ($1000)	Percent
Wall-Fired Clean Coal Project (SCS – 56%, DOE - 44%)	539	71
EPRI	123	16
DTI / Powergen	100	13
Total	762	100

Table 1-2 Clean Coal Project Cost by Phase

Participant	Total Cost ($1000)	Participant Share Percent	DOE Share Percent
Phase 0 – Pre-Award	298	59	41
Phase 1 - Baseline	1,677	55	45
Phase 2 - AOFA	3,828	55	45
Phase 3 – LNB, LNB+AOFA	4,947	55	45
Phase 4 – DCS, GNOCIS, Unit Optimization	4,017	70	30
Phase 5 - Closeout	394	55	45
Total	15,161	56	44

Other Power Plant Optimization Projects

The following paragraphs provide background information on the application of on-line optimization to power plants. As of 2002, other than the effort at Hammond, the authors are not aware of other applications of unit wide or plant wide on-line optimization applied in a utility power facility.

<u>Dynergy's Baldwin Energy Complex</u> - In January 2003, DOE entered into a cooperative agreement with NeuCo to develop and demonstrate integrated optimization software at Dynergy's Baldwin Energy Complex which consists of three 600 MW coal-fired units [DOE 2003A]. The project, scheduled for completion in 2006 and valued at $18.6M, will address sootblowing, SCR operations, overall unit thermal performance, and plant wide profit optimization.

2
UNIT DESCRIPTION

Georgia Power Company's Plant Hammond Unit 4 is a Foster Wheeler Energy Corporation (FWEC) opposed wall-fired boiler, rated at 500 MW gross, with design steam conditions of 2400 psig and 1000/1000°F superheat/reheat temperatures, respectively. Hammond 4 was placed into commercial operation on December 14, 1970 (Figure 2-1). Six Babcock and Wilcox MPS 75 mills supply eastern bituminous coal (12,600 Btu/lb, 30% VM, 54% FC, 1.0 % S, 1.4% N) to 24 FWEC Control Flow/Split Flame burners. The burners are arranged in a matrix of twelve burners (4 wide x 3 high) on opposing walls with each mill supplying coal to four burners per elevation. The unit is also equipped with a FWEC designed Advanced Overfire Air System (AOFA). The unit is also equipped with an SCR, coldside ESP, and utilizes two regenerative secondary air heaters and two regenerative primary air heaters. A summary of the Hammond 4 design characteristics is provided in Table 2-1 and a more detailed description of the unit and process follows.

Figure 2-1 Plant Hammond

Table 2-1 Hammond Unit 4 Design Characteristics

Unit Size	500 MW (nominal)
Commissioning Year	1970
Furnace	
Vendor	FWEC
Firing System	Opposed wall-fired
Configuration	Single Furnace, Overfire Air
Width × Depth	52.5 ft × 40 ft
Burner Zone Liberation Rate	425,000 Btu/hr-ft^2
Burners	FWEC Controlled Flow Split Flame
Coal Elevation Spacing	8.5 ft
Top coal elev.-to-furnace outlet (nose)	55 ft
Pulverizers	
Number of Mills/Mill Type	6 B&W MPS75 Mills
Air/Fuel Ratio	2.1
Coal	
Coal Type	Eastern bituminous
Higher heating value	12,600 Btu/lb (as received)
Sulfur	0.87%
Moisture	6.5%
Higher heating value	10.5%
FC/VM	1.6
ESP (cold-side)	
Specific collection area	379 ft^2 (based on 9 inch spacing)
Aspect Ratio	0.91
ESP Manufacturer	Research-Cottrell
Electrode Design	Rigid Discharge Electrodes
Number of Transformers	24
Electrical / Mechanical Fields	6 / 6
Rapper Type	Magnetic Impulse Gravity Impact
Conditioning Agent	SO_3
Main Turbine	
Vendor	Westinghouse
Type	Tandem-compound
Speed	3600 RPM
Rated main steam pressure	2400 psig
Rated main steam temperature	1000°F
Rated hot reheat temperature	1000°F

Boiler

The basic configuration of the boiler is a natural circulation, waterwall furnace with a partial division wall superheater and a heat recovery area (HRA), convection superheater, non-drainable pendant superheater, convection reheater, and extended surface and bare tube economizers (Figure 2-2 through Figure 2-5). Feedwater enters the boiler through the economizer inlet header, flows through two series-connected economizer sections, and leaves through the opposite ends of a split outlet header. The water mixes with re-circulated water from the furnace waterwalls, and leaves the steam drum through downcomer pipes. The downcomer pipes are arranged so that the water flows to all the furnace waterwalls through a series of feeder tubes connecting the lower end of the downcomers to lower waterwall headers. The flow through the downcomers and waterwalls is much greater than the feedwater flow by approximately an order of magnitude. As the circulated saturated water rises in the water-wall tubes, heat is continually transferred to it from the gas path, raising the quality of the liquid/gas mixture, and then returns to the drum. The saturated water and steam is then separated with the former returning to the drum and the saturated steam flowing to the convection, division wall, pendant superheat sections, and finally to the high pressure (HP) steam turbine (design point is 2400 psig, 1000°F). After the steam flows through the HP turbine, it returns to the boiler as cold reheat where it is reheated (design point 1000°F) and then flows to the intermediate pressure (IP) turbine. The pressure at the inlet of the IP turbine varies with load, ranging from approximately 550 psig at full load to 200 psig at lower loads.

<u>Economizer</u>

The economizer is a feedwater heater that extracts heat from the furnace exiting flue gas prior to the gas entering the air heaters. The typical gas temperature at the inlet of the economizer is 700°F while that at the outlet is 300°F. The economizer consists of horizontal extended surface and bare tubes arranged across the width of the boiler. The feedwater enters the economizer through the inlet header, flows through the inlet section and intermediate header to the outlet section, and then through the split outlet header and external feed pipes to the steam drum. The gas flow across the economizer tubes is opposite to the flow within the tubes.

<u>Steam Drum</u>

The steam drum is located at the top and front of the boiler, lying horizontal with its longitudinal axis traversing the boiler width. The steam drum serves as a water reservoir for the boiler circuits and as a mixing chamber for incoming feedwater and the water separated from generated steam. The drum contains the water/steam separating equipment, steam dryers, distribution piping for adding chemicals to the water, external feedwater manifold, and provides for blowdown of the water to control the concentration of solids.

Furnace Waterwalls

The furnace consists of four waterwalls formed by panels of welded fin tubes arranged to form a rectangular section in the boiler's front portion. The bottom of the front and rear waterwalls forms a wedge-shaped ash hopper. The bottom of the hopper is open for the discharge of ash into the ash pit. The furnace is sealed with a water seal which must be maintained to prevent boiler gases from escaping into the building or air being drawn into the furnace which is under a slight negative draft. The water seal also provides protection against over pressurization of the furnace (explosion protection) as the water seal would be blown out relieving the pressure to the atmosphere.

The furnace rear wall contains a nose section that properly distributes the furnace hot gas and air mixture across the pendant superheater section located just above the nose. The rear wall fins terminate above the nose, and tube spacing is increased to form a screen through which the gas leaves the furnace.

Heat Recovery Area (HRA)

The partition wall, rear wall, left and right side walls and roof tubes of the HRA consist of parallel rows of tubes with fins welded between the buses. The fins of the partition wall, which divide the HRA into two vertical gas passes, begin at the bottom of the wall but terminate before the top. The tube spacing increases above the fins to form a screen through which the hot gas flows. This allows the hot gas to flow down both gas passes of the HRA.

Steam from the steam drum enters the roof inlet header and then flows over the furnace in the HRA roof tubes to the upper partition wall header. The steam flow divides at this header and is distributed through parallel paths to the convection superheater.

Convection Superheater

The convection superheater is the primary superheater section and is located in the front gas pass of the HRA. The convection superheater's tube elements are shaped in vertical stacks of horizontal loops, two tubes in each loop, and the loops are arranged across the width of the HRA. Steam enters the superheater and flows upward, opposite the flow of the gas. The steam then flows downward to the superheater outlet header. From here, the steam flows through pipes to the five furnace division walls.

Division Walls

The five partition division walls are equally spaced across the width of the upper portion of the furnace. The walls are formed by parallel rows of tubes tied together at points along the tubes. Each wall of tubes enters the furnace above the top burners at right angles to the front furnace wall and then bends vertically upward. The walls terminate in outlet headers in the unheated furnace section above the roof tubes. Superheated steam enters the division wall inlet header from the convection superheater through transfer pipes. The steam is distributed to the five division walls and flows upward to the division wall outlet headers and then through transfer pipes to the pendant superheater.

Pendant Superheater

The pendant superheater is the finishing superheater section and is located at the furnace exit above the rear furnace waterwall nose. The pendant superheater tube elements are arranged in vertical loops across the width of the boiler. Supply tubes from the partial division walls distribute steam to the pendant superheater inlet. After flowing through the superheater tubes, the steam leaves through the outlet header and flows to the high pressure turbine as superheated dry steam at 1000°F and 2400 psi.

Reheater

The reheater consists of a drainable horizontal loop inlet tube section and a vertical loop outlet tube section. Both sections are located in and arranged across the width of the HRA rear gas pass. Steam from the high pressure turbine outlet (cold reheat) enters the reheater inlet section through the inlet header, flows upward opposite the flow of hot gas, leaves through the outlet header, and flows to the intermediate pressure turbine as hot reheat.

Windbox

The windbox surrounds the furnace area and supplies air to the low NOx burners to support combustion. Heated secondary air from the secondary air preheaters is supplied to the windbox assembly under pressure from the forced draft fans. Balancing dampers create a restriction as well as direct the secondary air toward either side of the windbox. The flow of secondary air into the furnace is regulated at each burner assembly by the inner air register, outer air register, and moveable sleeve damper. The secondary air duct supplies the burner windbox as well as the furnace overfire air system. The overfire air system is operated in conjunction with the low NOx burners to aid in the control of NOx production during the combustion process. The restriction imposed in the secondary air flow path as the secondary air enters the burner windbox creates a positive pressure in the secondary air duct and causes the air to be pushed upward into the overfire air duct assembly.

Advanced Overfire Air System

NOx formation is strongly dependent on the flame zone stoichiometry. The advanced overfire air system removes some of the combustion air (above the stoichiometric quantity) from the burner flame zone and reintroduces it later in the combustion area, away from the high temperature flames, reducing NOx formation, although generally at the expense of less carbon burn out and higher fly ash LOI. At Hammond 4, the overfire air system introduces air into the furnace immediately above the burners. The system utilizes air supplied from the secondary air ducts under pressure from the forced draft fans. When overfire air duct dampers are opened, less air is injected into the furnace through the burners and more air is diverted and injected above the burners through the eight overfire air ports (four each on the front and rear walls). The amount of overfire air delivered to the furnace is controlled by four overfire air control dampers admitting combustion air to the overfire air windbox. Although practice has varied since their installation, presently these control dampers are indexed to load. At full load, approximately 15 to 20% of the combustion air is provided through the overfire air ducts.

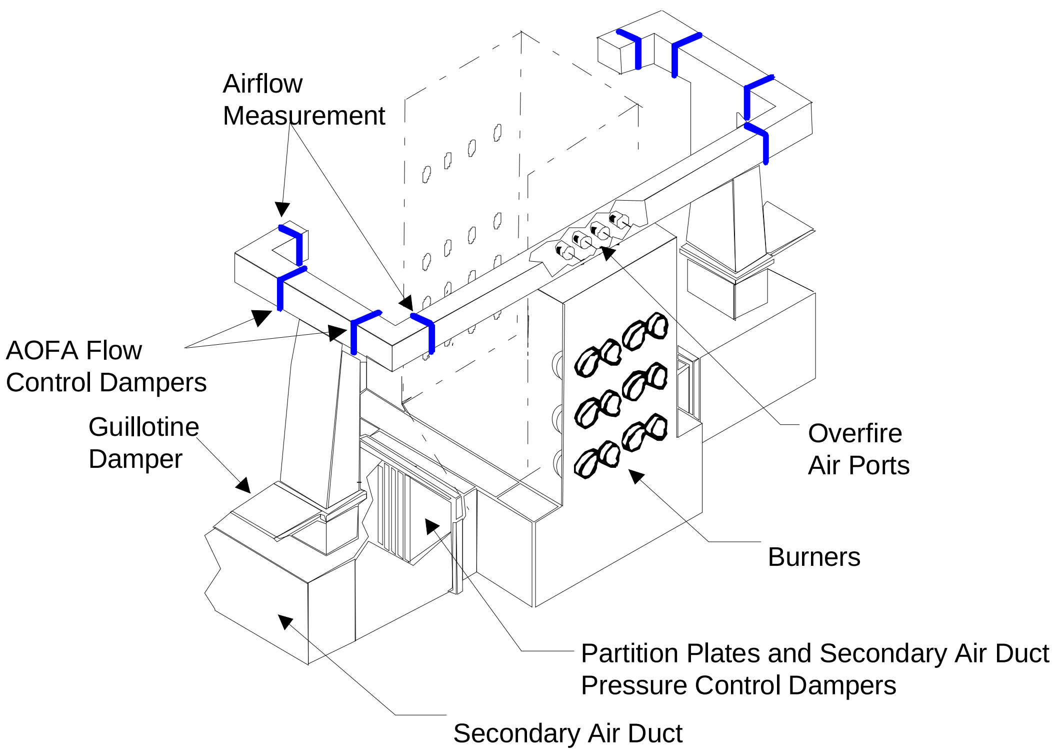


Figure 2-2 Boiler Overview

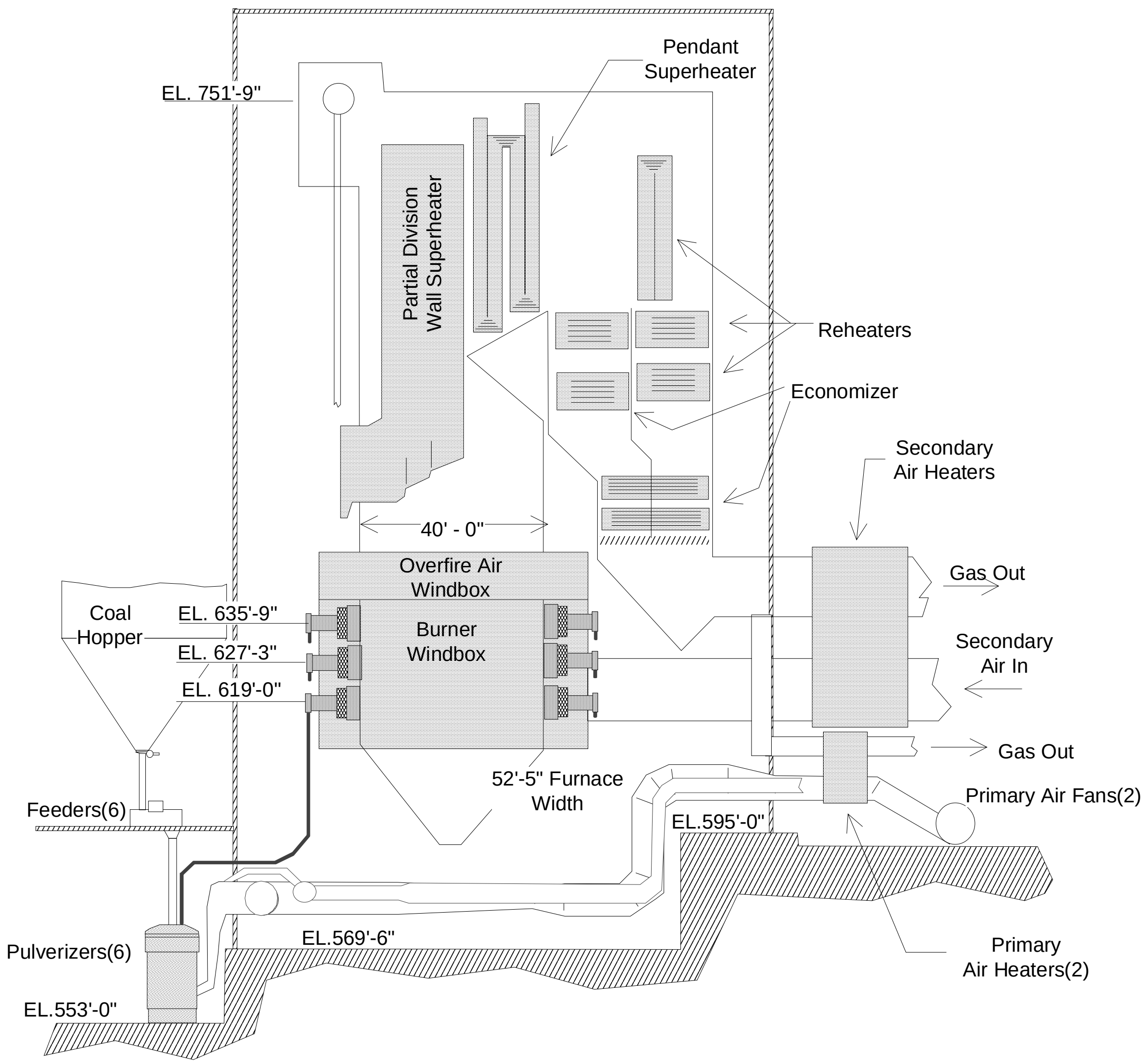


Figure 2-3 Boiler Outline

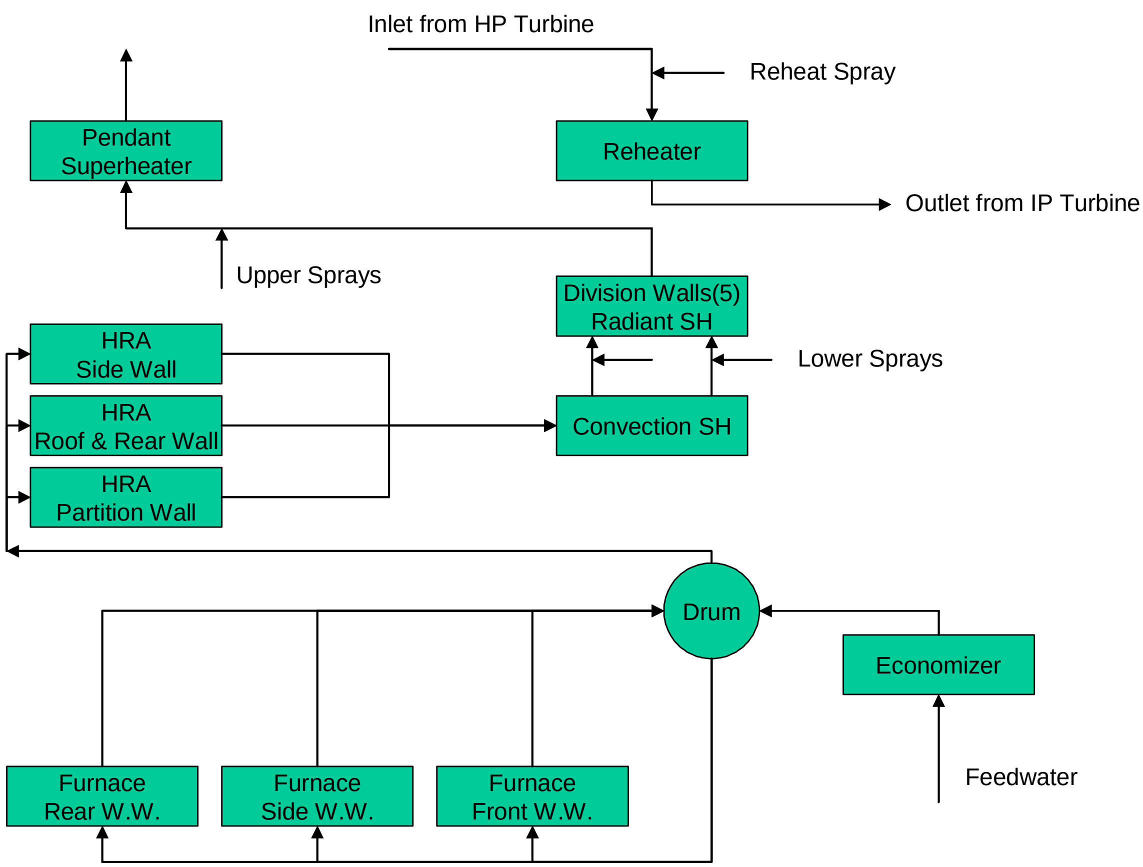


Figure 2-4 Boiler Steam/Water Flow Path

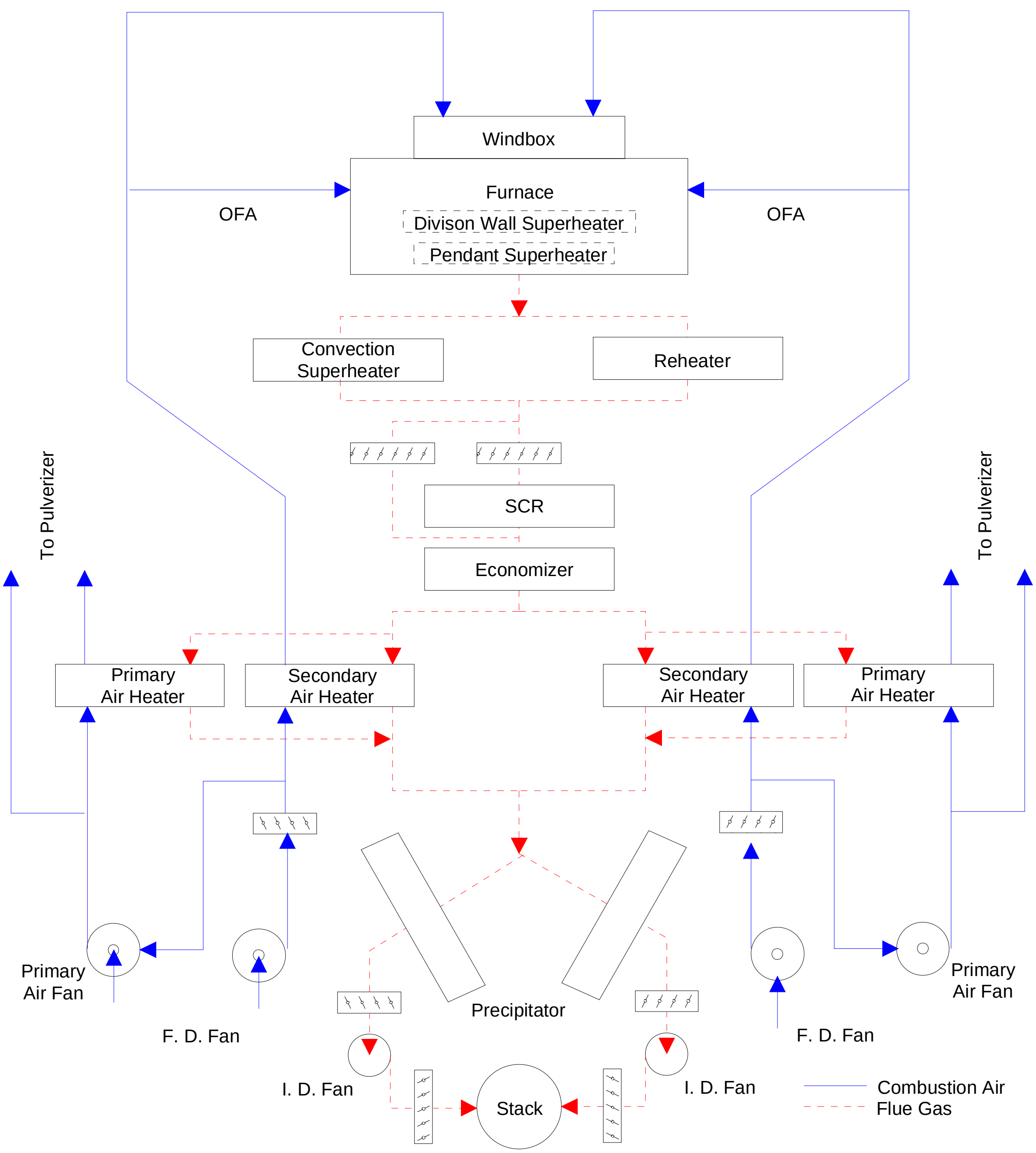


Figure 2-5 Boiler Combustion Air and Flue Gas Paths

Steam and Feedwater System

Main Turbine

The Unit 4 turbine is a Westinghouse three casing tandem-compound quadruple exhaust, condensing reheat turbine. The design ratings and steam conditions are as follows:

Guaranteed KW	500,000 KW
Speed	3600 RPM
Design Steam Conditions:	
Inlet Pressure	2400 psig
Inlet Temperature	1000°F
Reheat Temperature	1000°F
Exhaust Pressure	3.5 in Hg. Abs.

The turbine rating, capability, steam flow, speed regulation and pressure control are based on operation at rated steam conditions. The turbine-generator unit is capable of operation under increased conditions in steam pressure and temperature.

The high and intermediate pressure turbine is of the combination impulse and reaction type. The steam enters the high pressure element through two throttle valve steam chest assemblies, one located at each side of the turbine. The steam chest outlets are connected to the HP-IP casing through eight inlet sleeves, each connected to its nozzle chamber by a slip joint. Four of these inlet sleeve connections are in the base and four are in the cover. The eight throttle valves are controlled in partial mode instead of full-arc mode. The steam passes through the impulse stage and high-pressure blading to the boiler reheater through two exhaust openings in the outer casing base. At VWO (valve wide open), the design efficiency of the HP turbine is near 84%.

The steam returns from the boiler reheater to the intermediate pressure element through two interceptor-reheat stop valve assemblies, one located at each side of the turbine. The steam passes through the intermediate pressure element reaction blading to two exhaust openings in the outer casing cover. Each of these exhaust openings is connected through a separate crossover pipe to an opening in the casing cover of one of the low pressure turbines. The IP efficiency is in the range of 88 to 90%.

Each double flow low pressure turbine is a straight reaction double flow type element, with steam entering at the center of the blade path and flowing toward an exhaust opening at each end. From there the steam flows downward into a combined exhaust into the condenser.

Feedwater System

The feedwater system consists of two turbine-driven boiler feed pumps, one electric motor driven boiler fill pump, four high pressure, horizontal feedwater heaters, and the associated high pressure piping and valves. All feedwater equipment is utilized to move and heat the water from the de-aerator surge tank to the boiler economizer section.

The feedwater flow starts at the deaerator storage tank and is gravity fed to the boiler feed pumps through a common suction line. The boiler feed pumps increase the feedwater pressure and provide feedwater flow through four high-pressure extraction feedwater heaters, to further preheat the feedwater. The feedwater travels through the feedwater pressure valve and then enters the boiler lower economizer section.

Boiler Feed Pump

The two boiler feed pumps are five stage DeLaval pumps driven by Westinghouse steam turbines. The rated capacity of each pump is 4800 gpm at 7450 feet of head at 5100 rpm. The boiler feed pumps are used to maintain boiler drum level, and provide spray water pressure and flow for the superheater desuperheaters. Superheat spray flow is extracted from the feedwater following the feedwater pumps and prior to the next feedwater heater. Though not currently used on this unit, reheat spray flow is extracted from an inner stage of the boiler feed pump.

Boiler Feed Pump Drive Turbine

The boiler feed pump turbines, manufactured by Westinghouse, are straight condensing type designed for variable high-speed operation. The turbine is arranged for direct connection through a flexible coupling to the boiler feed pump. Each of the two turbines has a rated speed of 5100 rpm and a maximum speed of 5300 rpm. Steam is supplied to the turbine from both the main steam and cold reheat lines. The rated steam conditions are 155 psia and 680°F for low pressure (cold reheat) and 2415 psia and 1000°F for high pressure (main steam).

Electrostatic Precipitator (ESP)

The unit is equipped with Research-Cottrell electrostatic precipitators located in the furnace gas outlet ducts. The ESP electrically attracts and collects suspended fly ash particles from exiting furnace outlet gas, which reduces the amount of particulate matter released into the atmosphere via the stack. There are 24 ash collection hoppers provided for the collection and disposal of fly ash removed by the ESP. The specific collection area of the ESP is 379 ft^2. An SO_3 flue gas conditioning system is installed and is used depending on coal type.

3
OPTIMIZATION ISSUES

The overall goal of the Hammond project was to demonstrate the feasibility of on-line optimization techniques, individually at the component level and collectively at the unit level. While component-level optimization utilizing neural-network or statistical based tools has been well proven, at the start of the project, coordinated unit optimization had not yet been demonstrated on utility units.

What is Optimization

In its most general sense, optimization is the procedure of finding a global extrema (either a maximum or minimum) of some characterization of a process or a design, possibly subject to one or more constraints applied to the inputs or outputs of the process or design. For all optimization problems, there are the following:

- *Objective Function* - A scalar or vector valued function that represents how well we are doing in approaching a goal. Also known as *Criteria, Payoff, Value,* or *Cost Function.*
- *Manipulated Variables* - A set of variables which affect the cost function. Also known as *Control* or *Decision Variables*.
- *Constraints* - Limits and other conditions placed on the control variables, whether directly, or through functions of these variables.

In its simplest sense, the problem may be stated as follows:

$$\text{Find a set of decision variables: } \mathrm{X} = [x_1, x_2, \ldots x_n]^T \in \Re^n$$

$$\text{To minimize the real valued cost function: } f(X)$$

subject to:

$$\text{Inequality constraints: } G(X) \leq 0$$

$$\text{Equality constraints: } H(X) = 0$$

A feasible point is an X that satisfies the equality and inequality constraints and X^* is the feasible point that minimizes $f(X)$.

Further, it is sometimes convenient to define a process model relating directly manipulated process inputs, X and non-manipulated process inputs, Z, to the process outputs:

$$Y = P(X, Z)$$

In which case, the slightly revised problem would be as follows:

$$\text{Minimize:} \quad f(Y)$$

where:

$$Y = P(X, Z) \in \Re^m$$

$$\mathrm{X} = [x_1, x_2, \ldots x_n]^T \in \Re^n$$

subject to the constraints:

$$G_Y(Y) \leq 0$$
$$H_Y(Y) = 0$$
$$G_X(X) \leq 0$$
$$H_X(X) = 0$$

An example of a one-dimensional optimization problem is shown in Figure 3-1. In this example, the feasible region (non-shaded area) is disjoint as a result of the inequality constraint relating X to Y. A factor adding to the complexity in finding the best solution is that there is a local minimum that is relatively far removed from the global minimum. If the initial starting point for the optimizer was in the region where the local minimum is located, depending on the methodology used to solve the problem, the region where the global optimum is located may not even be searched.

The efficient solution method for an optimization problem depends greatly on the nature of the decision variables (X) and relating system of equations (f, G, and H) and may be categorized into an *optimization tree* such as shown in Figure 3-2 [OTC03]. When the constraints and objective function are linear functions of the decision variables, the problem is a linear programming (LP) problem. When these are non-linear functions, it is a non-linear programming (NLP) problem and the solution generally requires an iterative procedure.

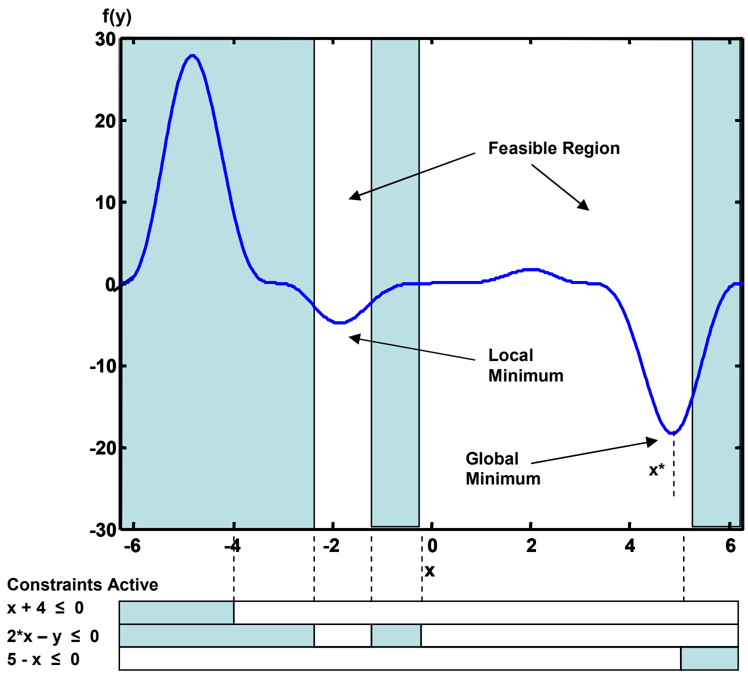
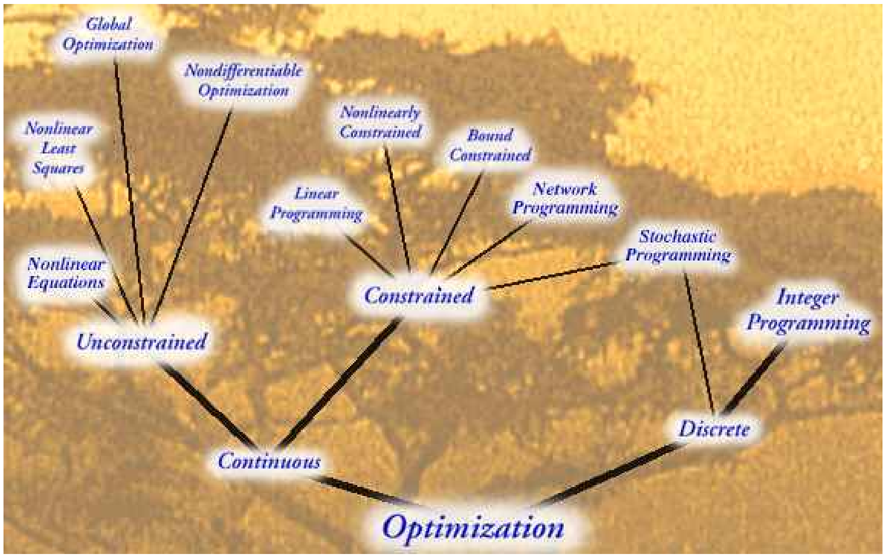


Figure 3-1 Graphical Representation of a One-Dimensional Optimization Problem



Source: US DOE Argonne National Laboratory, Optimization Technology Center, http://www-fp.mcs.anl.gov/otc/Guide/OptWeb/

Figure 3-2 Optimization Tree

In most real applications, generally only the simplest problems can be cast conveniently and completely into a single cost function. When done, although it may not be entirely inappropriate, the side effects of the optimization are oftentimes ignored. When these side effects cannot be ignored, it may be best to consider the problem as a collection of objective functions:

$$F(Y) = \left[f_1(Y), f_2(Y), \ldots, f_k(Y)\right]^T$$

The vector objective function $F(Y)$ implies multiple goals or criteria. The strategies for solving *multi-criteria optimization*[1] problems are numerous and include those based on "classical" optimization techniques and those that have their foundations in fuzzy logic, evolutionary computation, and game theory [MA03]. Regardless of the approach, the decision maker's preferences or values are modeled either explicitly or implicitly. In many real-world applications, the components of $F(Y)$ are often competing, meaning that improving one component is often at the expense of degradation of one or more of the other components.

When considering a collection of optimizers, there are a number of possible scenarios that may be found when performing unit optimization. These include the following.

<u>Case 1: Independent Processes</u> – The processes are completely independent, sharing no inputs or outputs:

$$c_1 = f_1(x_1)$$

$$c_2 = f_2(x_2)$$

In this case, the optimum may be found by optimizing each process independently. Although adding unnecessary complexity, the system could also be treated equivalently as a multi-criteria optimization problem with the vector objective function:

$$f(x_1, x_2) = \left[f_1(x_1), f_2(x_2)\right]^T$$

Example: Boiler and cooling tower

<u>Case 2: Processes Coupled Through Decision Variables</u> – The processes share one or more common decision variables:

$$c_1 = f_1(x_1, x_3)$$

[1] Also known as *multi-objective optimization* or *vector optimization.*

$$c_2 = f_2(x_2, x_3)$$

The processes could be optimized independently, but it is unlikely that the optimal decision value for the Process 1, $\hat{x}_3$, would be the same as that for Process 2, $\hat{x}_3$. Reconciliation of the two recommendations would require some type of multi-criteria technique. If the f_i could be cast as a cost or other common metric, then a reasonable approach may be to sum the cost functions:

$$c = f(x_1, x_2, x_3) = f_1(x_1, x_3) + f_2(x_2, x_3)$$

to obtain the total cost. Classical optimization methods could then be applied to this problem.

Example: Boiler performance and ESP performance, where the common decision variable is excess oxygen.

Case 3: <u>Dependent Variable Coupled Processes</u> – An output, though not the one of most interest, of one process affects the output of another process:

$$c_1 = f_1(x_1) \ ; \ z_1 = p_1(x_1)$$
$$c_2 = f_2(x_2, z_1)$$

As with Case 2, the processes could be optimized independently, but the optimum determined through this means would necessarily be inferior to that obtained when the two are considered a system. Also, if treated as independent, the optimum is likely to depend on the sequence in which the system is solved. Again, in general, reconciliation would require some type of multi-criteria technique. If the f_i are a cost or other common metric, then a possible approach would be:

$$c = f(x_1, x_2) = f_1(x_1) + f_2(x_2, z_1) = f_1(x_1) + f_2(x_2, p_1(x_1))$$

Example: f_1 represents boiler performance, f_2 represents ESP performance, and z_1 is fly ash unburned carbon.

More complicated combinations of the above cases are also possible:

Cross coupling of dependent variables:

$$c_1 = f_1(x_1, z_2), \ z_1 = p_1(x_1)$$
$$c_2 = f_2(x_2, z_1), \ z_2 = p_2(x_2)$$

Sharing of decision variables and cross coupling of dependent variables:

$$c_1 = f_1(x_1, x_3, z_2)\,,\ z_1 = p_1(x_1)$$
$$c_2 = f_2(x_2, x_3, z_1)\,,\ z_2 = p_2(x_2)$$

In contrast to single-objective optimization, there may not be one single, global solution in a multi-objective problem but a set of points which satisfy some definition for an optimum [MA03]. For two feasible points, X_1 and X_2, X_1 is said to dominate X_2 if $f_i(X_1) \leq f_i(X_2)$ for all i. For multi-objective optimization, a prevalent concept in defining an optimal point is that of *Pareto optimality*. A point, X, is *Pareto optimal* if it is feasible and it is the best that can be achieved in one of the f_i without adversely impacting the other $f_{j,\, j \neq i}$. The set of points that meet the Pareto optimality requirement (the *Pareto optimal set*) constitute the best that can be achieved without violating the given constraints. The decision maker then must choose by some method which of these constitutes the best solution to the problem at hand.

<u>Weighted-Sum Method</u> - Through sometimes maligned, a very common, straightforward attack is to use a weighted-sum strategy which maps the multi-objective problem into a scalar optimization problem that may be approached using more familiar constrained optimization methods [CDFK03]:

$$\text{Minimize:}\quad f(X) = \sum_{i=1}^{m} \left(w_i \cdot f_i(X) \right)$$

An important issue when using this technique is the selection of the w_i such that the summation is meaningful. In many cases, the most convenient scaling would be to currency (either profits or losses). For example, if C_1 is the production rate of product *A* and C_2 is the production rate of product B, then a reasonable selection of the w_i would be the unit profit margin of each product and the overall goal would be to maximize the overall profit. In many instances, the selection of the weights, w_i, is not particularly obvious. For example, if, as in the above example, C_1 is the unit profit margin but C_2 is the environmental impact, what would be the best choice for the C_i? The best selection would likely depend greatly on the decision maker's values.

<u>Lexicographic Method</u> - Another utilized approach is the hierarchical-based, lexicographic method. In this approach, the decision maker ranks the objectives in order of importance from 1 (most important) to k, where k is the number of objective functions. Starting with the most important, each individual objective function is optimized sequentially, ignoring the less important objectives. As the less important objective functions are optimized, additional

constraints are incorporated which prevent deterioration of the more important objectives. This may be expressed as follows:

$$\text{Find: } X = [x_1, x_2, x_3, \ldots x_n]^T \in \Re^m$$

$$\text{To minimize: } f_i(X)$$

$$\text{Subject to: } f_j(X) \le f_j(X_j^*)\,;\, j = 1,2,\ \ldots,\ i\text{-}1;\ i > 1$$

$$i = 1,2,\ \ldots,\ k$$

Combustion Optimization

As an introduction to multi-process, coordinated optimization, it is useful to consider one process optimization, such as combustion optimization. Combustion optimization is the procedure by which NOx emissions, combustion performance, and safety are balanced to achieve or approach a predetermined goal. In most instances, the goals are defined in terms of performance inequality constraints such as:

- Boiler performance - Maximize considering other constraints.
- NOx - Reduce to below guarantee value and/or compliance limit.
- Fly ash loss-on-ignition (LOI) - Hold below guarantee value and/or state imposed utilization limit.
- Safety - Increasing the safety margin.

These goals may be defined for one or more operating conditions. Only when all constraint goals are clearly met, will further NOx optimization be performed. Another possible scenario is as follows:

- Boiler performance - Maximize.
- NOx - Minimize.

Since the control setting for optimum boiler performance is unlikely the optimum NOx setting, a value judgment must be made as to what performance indicator to sacrifice.

Combustion optimization has historically been a difficult problem for a number of reasons. Unlike SO_2 emissions which are primarily a function of the sulfur content of the fuel, NOx emissions are highly dependent on a number of parameters. NOx emissions are formed in the combustion processes through the thermal fixation of atmospheric nitrogen in the combustion air producing "thermal NOx" and the conversion of chemically bound nitrogen in the fuel producing "fuel NOx." NOx emissions can be reduced by lowering: (1) the primary flame zone oxygen level, (2) the time of exposure at high temperatures, (3) the combustion intensity, (4) primary flame zone residence time, and (5) flame temperature. As a result, NOx emission rates are strongly influenced by the apportionment of the air to the burners and overfire air system, burner adjustments, air-to-fuel ratio, and other controllable parameters.

An example of the interdependencies and conflicting goals which must be considered in boiler tuning can be observed in Figure 3-3. As shown, as excess air (or equivalently, excess oxygen) decreases, NOx decreases while LOI increases. High LOI values are indicative of poor combustion and therefore possibly poor boiler performance. Also, on units which sell their ash, an increase in fly ash LOI can change the fly ash from a marketable commodity to an undesirable byproduct. A decision must be made as to what is the optimum operating condition based on economic and environmental considerations. Similar compromises must also be made when optimizing boiler efficiency. In this case, the optimum operating condition is clear as long as the performance index is defined as boiler efficiency and other parameters (such as NOx emissions) are not considered.

A source of difficulty in optimizing the combustion process is the degrees of freedom resulting from the availability of a number of tuning points. Historically, combustion optimization for the boilers equipped with low NOx technologies (such as low NOx burners and overfire air) is considerably more difficult than that required for setup of boilers without these technologies. This added difficulty is in part due to (1) the result of the increase in the number of adjustments and sensitivity of these burners to operating conditions and (2) conflicting goals [SCS98a].

In addition to variations with excess oxygen levels and load, boiler combustion parameters also vary significantly during long-term operation and it is evident that a number of uncontrolled and unidentified variables greatly influence combustion performance. These influencing variables include mill operating conditions (primary air temperatures, air/fuel ratios, flows, grind, and moisture), secondary air non-uniformity (air register settings, forced draft fan bias, and windbox pressure differential), coal variability, etc. Since these factors vary with time, optimization conducted at one time may not be optimal over a longer period.

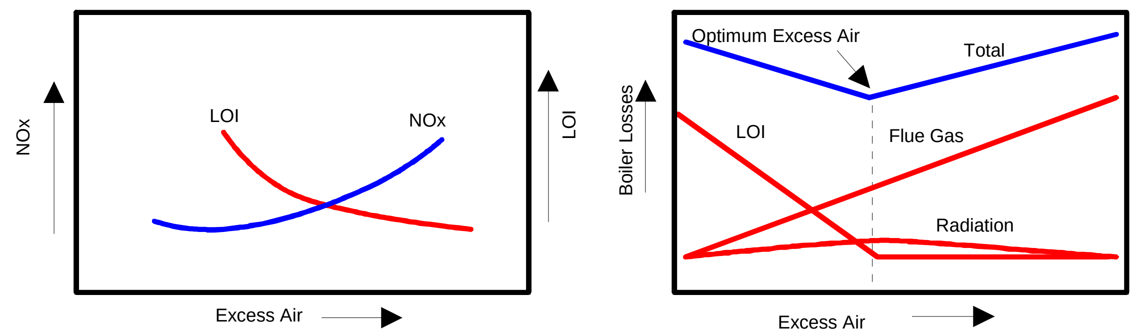


Figure 3-3 Tradeoffs in Boiler Optimization

Unit Optimization

Online boiler optimization is often considered the first focus when improving the overall performance of a unit in that in many respects this process has the most flexibility and potentially the most returns. Expanding the optimization envelope beyond the boiler provides additional opportunities but with the expense of additional complexity. These complications are the result of several factors:

Process Interactions – Like most process facilities, a coal-fired power plant is effectively a collection of components interconnected to form a closed-loop interactive system. This interconnectivity is such that adjustments made on one component may affect the performance of one or more downstream components. Examples of these interactions are shown in Figure 3-4 and Figure 3-5. As shown in the former, changes in the boiler affect conditions in the steam cycle and ESP while sootblowing decisions may also be affected. To a degree, the steam cycle conditions also affect boiler conditions. For example, in T-fired units, steam temperatures are controlled by furnace tilts which also have an impact on NOx emissions and LOI. The interaction between excess oxygen, an important boiler tuning parameter, and the ESP is shown in Figure 3-5. For example, decreasing excess oxygen typically increases LOI, which increases (again, typically) stack fly ash emissions.

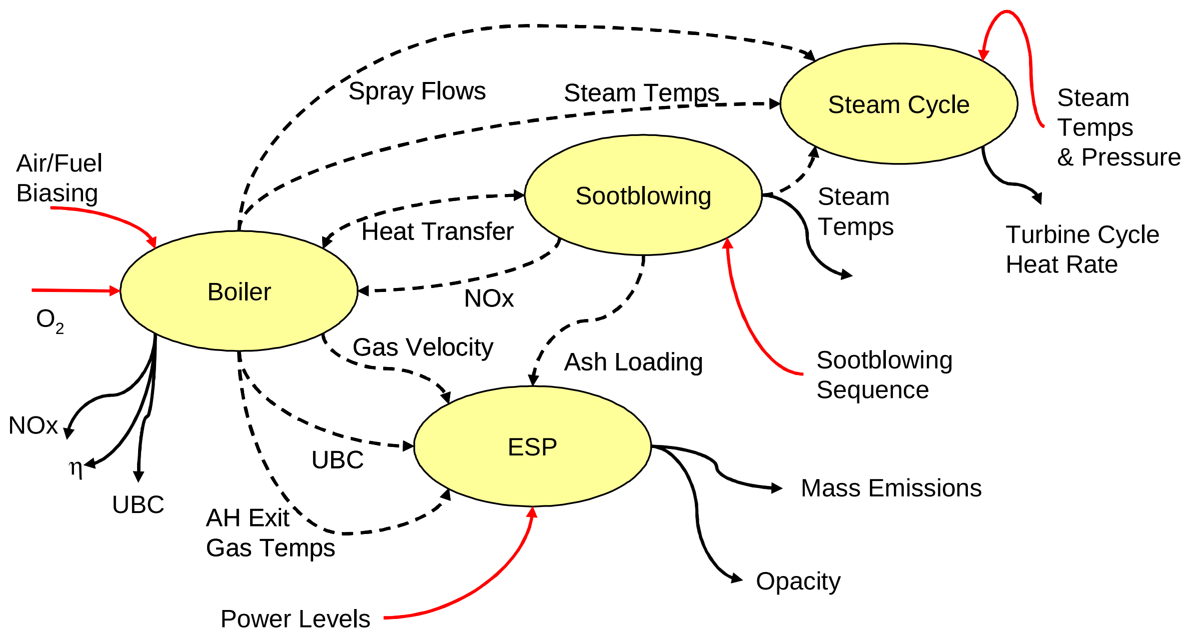


Figure 3-4 Process Interaction

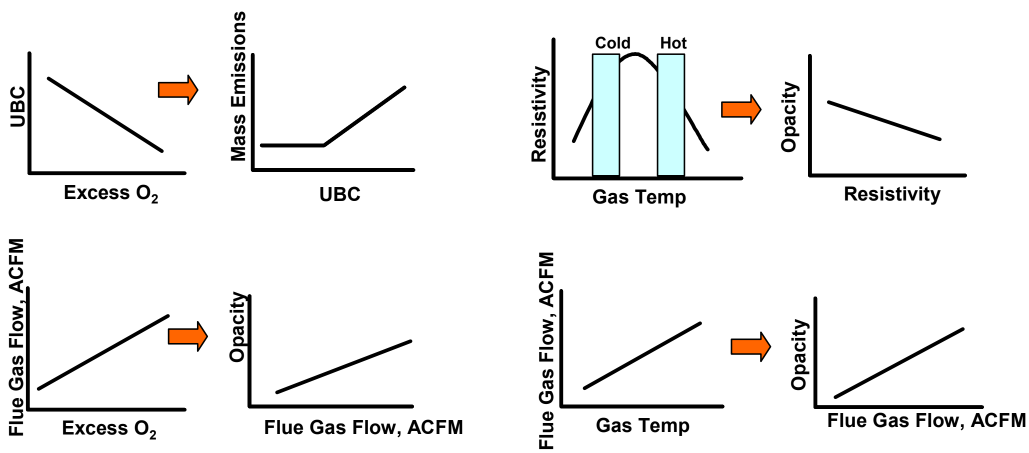


Figure 3-5 Example of Process Interaction between Boiler and ESP

Complex, non-linear non-stationary processes – Many of the processes in the power plant are highly non-linear with respect to the controllable inputs. An example of this is shown in Figure 3-6. These non-linearaties occur not only in the combustion process, but also particularly in post-combustion emissions control equipment such as SCRs and ESPs. The non-stationary response is the result of many factors including equipment degradation between maintenance intervals, varying and marginal fuels, and changing ambient conditions. An example of this is shown in Figure 3-7 in which a typical profile for plant cooling water temperature is shown during an approximately one-year period. Although cooling water temperature is not generally considered directly controllable, unit efficiency is greatly dependent on this temperature and its effect may need to be considered when performing unit optimization. Another example is shown in Figure 3-8 in which is shown an example of coal higher heating value variation over a year.

Difficult Measurements – Many important process performance indicators in a power plant are difficult to measure, particularly on a continuous basis. Examples of this include unit heat rate, boiler efficiency, fly ash LOI, and LP turbine efficiency. For many of these, the timeliness of the reading is not sufficient or there is suspect precision and repeatability.

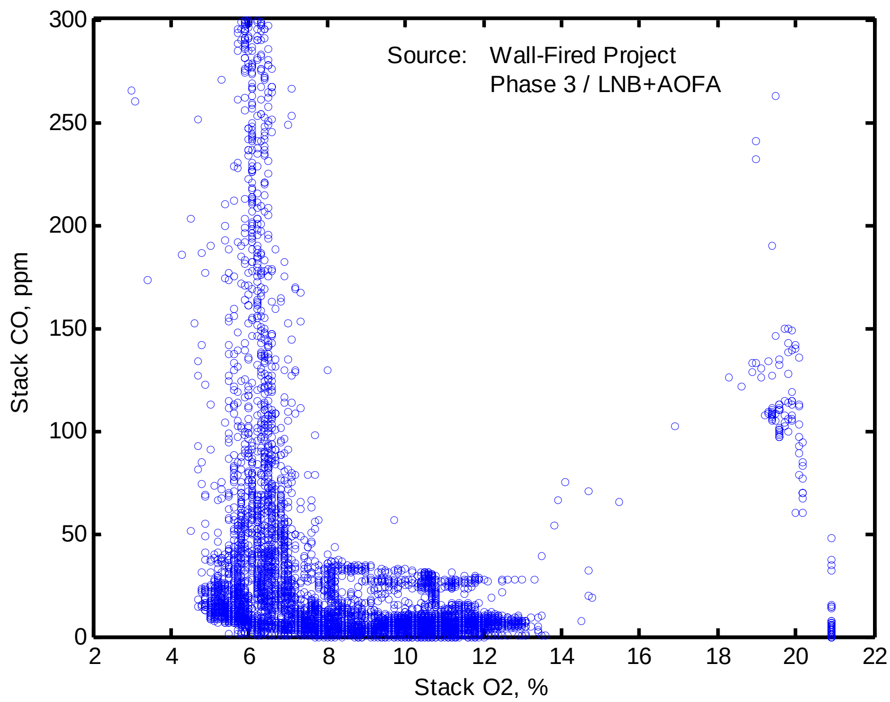


Figure 3-6 Example of Stack CO versus Stack O2

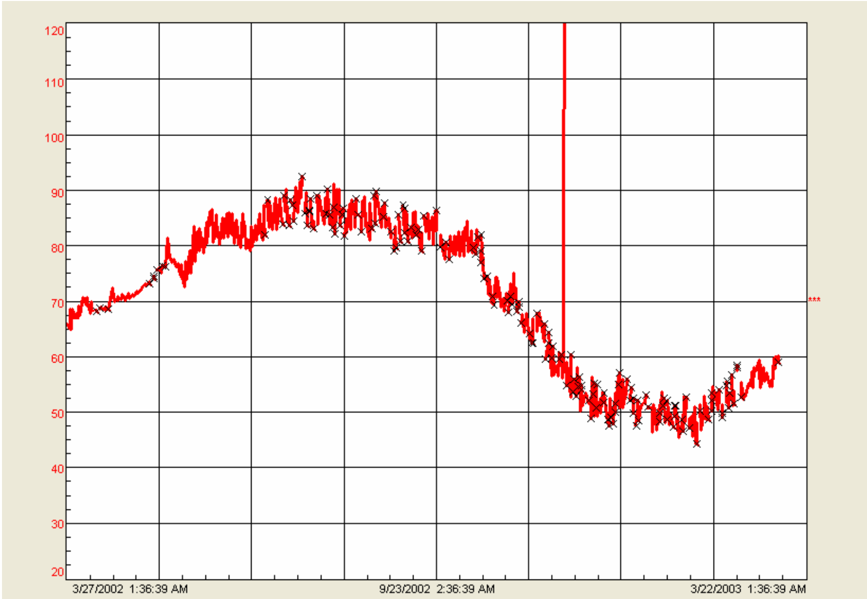


Figure 3-7 Example of Cooling Water Inlet Temperature Over an Extended Period

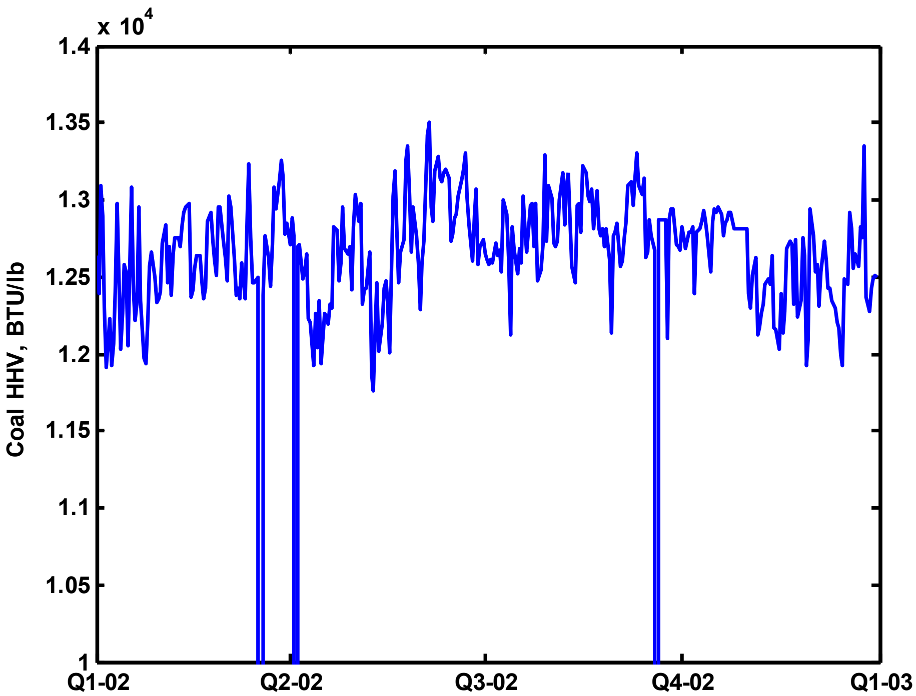


Figure 3-8 Example of Coal Higher Heating Value Over a Year

An example of some optimization considerations is provided by boiler excess oxygen. The control of excess oxygen is a very important consideration on any unit. Sufficient oxygen is necessary for the proper combustion of fuel; yet efficiency is adversely affected when too much excess oxygen exists in the furnace (Figure 3-9). Excess oxygen at most units is measured at multiple points at the economizer outlet. The selected excess oxygen signal is used for reference in calculation of a fuel/air ratio multiplication factor which is used to adjust the total air flow demand signal. For most units, the desired excess oxygen is a function of unit load and possibly other factors, such as the number of pulverizers in operation. The DCS adjusts the combustion air to maintain this excess oxygen set point. As shown, in addition to boiler efficiency, there are a number of other factors to consider when setting the excess oxygen. Too low of excess oxygen results in reduced safety margins, low steam temperatures, and higher LOI, while if excess oxygen is too high, the tendency is for lower boiler efficiency and increased stack particulate emissions.

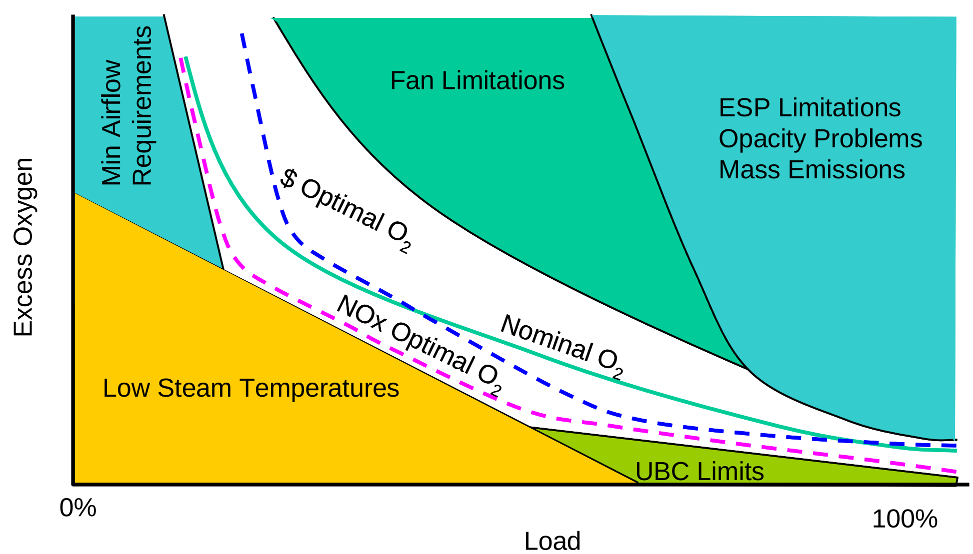


Figure 3-9 Optimization Envelope

4 APPLICATION OF EPRI'S TPCO GUIDELINES

The overall goal of the project at Hammond was to demonstrate the feasibility of on-line optimization techniques individually at the component level and collectively at the unit level.[1] While component-level optimization utilizing neural-network or statistical-technique based tools has been well proven, overall unit optimization based on operating costs has not been demonstrated [EPR02]. Issues associated with overall unit optimization are:

- While each component of the unit affects the performance of other individual components and the overall unit performance, it is not easy to prioritize the effects of the various operating parameters.
- The number of operating parameters which need to be monitored and taken into account in the overall unit optimization can be very high (more than 100-300). Appropriate problem formulation could reduce this effort by selecting only the components and operating parameters (input and output variables) which are important for the optimization.
- While the costs of an optimization system can be summarized with a reasonable accuracy, the benefits from such a system are difficult to estimate. As a result, utility management does not have adequate information to decide whether to proceed with an optimization system.

To address these issues, EPRI developed the Total Plant Cost Optimization (TPCO) Guidelines, a software tool. This software provides guidance on the benefits of on-line optimization and the selection of the process and control variables. The specific goals for the Hammond project were to provide guidance on the design of an on-line, closed-loop system which optimizes overall unit operating costs, and to estimate the benefits from such an optimization. With regard to problem formulation (optimization system design), the purpose of this task was to identify the most important components and operating parameters to be included in the overall unit optimization. Based on this analysis, a component was to be identified, which would be included in the optimization analysis, along with the boiler, ESP, and sootblowing system, which were already included. A beta version of this software was utilized in this project.

[1] This section is a summary of the final report submitted to SCS and EPRI by ENTEC in performance of their work at Hammond.

The purpose of EPRI's TPCO Guidelines is to assist utilities in:

- Assessing whether total plant cost optimization is appropriate and cost-effective for their plant,
- Identifying the most important control variables and plant components for inclusion in the optimization system, and
- Estimating the benefits from such an optimization.

This software product is intended for utilities considering performance optimization based on plant operating costs. In this situation, it is important to identify the key variables and processes which affect these costs. Prioritization of the variables helps formulate the optimization problem in such a way that it is easier and less time-consuming to calibrate the model.

The TPCO Guidelines consist of a written guideline and an Excel spreadsheet. The report provides guidance on the suitability of total plant cost optimization for each plant under consideration, as well as how to use the TPCO spreadsheet to identify the control variables and estimate the benefits due to optimization.

Problem formulation consists generally of the following aspects:

<u>Determination of the Objective Function</u> - This is the function which will be minimized or maximized. In the case of the current project, the optimization objective is assumed to be: minimize variable operating costs of the unit, defined to also include the value or penalties associated with emissions, when available.

<u>Control Variables</u> - These are the variables that are manipulated to obtain an optimum. These may be independent or not independent.

<u>Outputs</u> - Key performance indicators such as efficiencies, emissions, and operating costs.

<u>Models</u> - These are the numerical formulations of the process. The selection of the models depends on the type of components being included in the optimization and the availability of models based on first principles, empirical equations, or neural networks.

The TPCO Guidelines help the user prioritize the plant components and determine the most important inputs (control variables). The analyses which can be carried out using the TPCO guidelines can be classified into the following three types:

<u>Quick Assessment</u> - Working in the "Plant-Level" sheet, the user has the flexibility to change only a few key parameters (e.g., generation capability of unit, plant elevation, capacity factor and coal type) for quick results; or to change many more parameters, including the key control

variables, and obtain a more site-specific assessment.

Detailed Analysis - The user works in the "Plant-Level", "Coal Library" and "All Inputs & Results" sheets to adjust the key inputs ("calibration") to reflect plant-specific characteristics. However, the user does not change the default correlations of the spreadsheet (e.g., correlations between reheat spray and plant heat rate, or excess air and NOx emissions). This type of application is the most common for the TPCO spreadsheet. The user provides enough site-specific inputs, so that the analysis adequately reflects the characteristics of the plant under evaluation, is not time-consuming and does not require extensive changes of the spreadsheet.

Spreadsheet Customization - In addition to the variables involved in the previous two analyses, the user may customize the spreadsheet by modifying the default correlations between key inputs and outputs ("All Inputs & Results" sheet, input groups B through I).

In the case of Hammond, the utilization of the TPCO Guidelines involved detailed analysis and customization of the spreadsheet to reflect Hammond operating characteristics.

The results of the analysis are shown in Table 4-1 and Table 4-2. Table 4-1 is a summary of the unit performance and costs for the "baseline" and "optimum" operating conditions. The difference between baseline and optimum is an estimate of the annual cost reduction due to optimization and provides a basis for deciding if optimization is a cost-effective investment for the power plant under consideration. Table 4-2 illustrates how the key control variables are ranked based on their impact on plant operating costs ($/year).

Analysis and Key Findings for Hammond

As stated previously, the objectives of this task were to utilize EPRI's TPCO Guidelines to:

- Support the formulation of the overall unit optimization at Hammond;
- Recommend a component to be added to the optimization in addition to the boiler, sootblowing, and ESP, which have been included already; and
- Estimate the benefits due to overall unit optimization.

The approach followed was first to calibrate the TPCO Guidelines to reflect the performance of Hammond, and to carry out various scenarios reflecting key operating parameters (e.g., cost of fuel and value of emission allowances) in order to identify the process and the benefits due to optimization.

Calibration

Careful examination of the unit design and operating characteristics was critical. As a first step, the key control variables were identified in all the unit components. At the time of this study, Hammond bought coal mostly in the spot market; therefore there is significant variability in coal quality. However, there is no provision to control coal quality through coal blending. For this reason, coal quality was not considered to be a variable.

Excess air at full load was determined to have a lower limit at 19% to limit CO emissions and an upper limit of 27% due to FD fan capacity limitation.

The boiler has low-NOx burners with overfire air system. The operating range of the overfire air dampers was determined to be 0-100% open. Considering that the TPCO Guidelines were still under development when the Hammond project started, it was decided to modify the NOx prediction correlations of the spreadsheet utilizing actual data from Hammond. As a result, NOx emissions being predicted by the spreadsheet reflected closely the actual performance of the unit.

Superheater spray flow is used to control the superheater outlet temperature, but the reheat spray is not used. Therefore, the operating range for the superheater spray was: 0-75,415 lbs/hr (max; 1% of the total steam flow rate).

Key operating variables related to the steam turbine were determined to have the following operating ranges:

Main steam pressure:	2300 - 2400 psig
Main steam temperature:	990 - 1010^{o}F
Reheat steam temperature:	990 - 1010^{o}F
Cycle make up water:	0.0 - 1.0% of the main steam flow rate

Hammond does not have a cooling tower. The condenser is usually cleaned during scheduled outages. When the cleanliness factor reaches the 50-55% range, the plant reduces load to 40% and backwashes the condenser (usually done overnight). The cleanliness factor ranges from 55 to 70%.

Analyses

Base Case

For the base-case inputs, the performance and cost summary is shown in Table 4-2. The first estimate for cost savings due to optimization is $273,000 per year.

With regard to the prioritization of the key variables (Table 4-2), steam turbine inlet conditions (main steam temperature and pressure, and reheat temperature) and make-up water have the most significant impact on operating costs. This highlights the importance of the steam turbine and the key set points which determine its efficiency. It should be noted that the priorities of factors such as excess air and overfire air damper position are low, mainly because in this scenario NOx emission allowances were valued at zero $/ton.

Sensitivity Analyses

To assess the impact of key parameters on the priority list of the control variables and the savings due to optimization, a number of sensitivity analyses were carried out. For illustration purposes a few selected sensitivities are described in the following paragraphs.

<u>Coal Price</u>

The price of coal was increased by 150% over the baseline value. The impact on the prioritization of the control variables was minimal. The ranking of the excess air increased from 8^{th} to 7^{th}, but the rank of most of the control variables did not change. The main change was the increase in the total benefits due to the optimization from $273,000 per year (baseline) to $525,000 per year.

<u>NOx Emission Allowance</u>

In the baseline case, it was assumed that NOx emission reduction does not have any monetary value for the utility. As of first quarter 2003, NOx emission trading is not an option for most domestic US markets. However in conducting this study, it was felt that in the future that just as there is now trading in SO_2, at some point in the future, NOx trading would also be a possible compliance option. When this study was conducted, it was felt that $1500/ton NOx emission would be a reasonable NOx emission allowance and this is the value used for this study. As Table 4-3 shows, this change affected significantly the prioritization of the control variables. More specifically, the overfire air and excess air moved to 1^{st} and 2^{nd} place in the ranking order from 10^{th} and 9^{th}, respectively. Also, the benefits due to optimization increased to $1.87 million per year. As mentioned above, $1500 was used as the NOx emission allowance. More recent estimates of this allowance are in the range of $5000 to $7000/ton NOx removed and expectations are that these will go into effect for most of the US market between 2004 and 2005. If this is the case, the benefits due to optimization could increase proportionately.

<u>Condenser On-line Cleaning</u>

The condenser was perceived to be a very important component with significant impact on the unit heat rate. For this reason, it was decided to explore some operating and design options associated with the condenser. Controlling the water velocity would require re-design, such as replacement of the existing fixed-speed water circulation pumps with larger variable speed pumps. Eventually, it was decided that such a design change was too expensive for the expected benefit.

Key Findings and Recommendations

"Process X"

The analyses carried out suggest that the steam turbine is a critical component and should be included in the overall unit optimization system. The setpoints of the inlet condition of the steam turbine (main steam temperature and pressure, and reheat temperature) are the most critical parameters for the overall unit optimization. While these parameters are not considered typical control variables, they are setpoints and can be changed by the plant operator. Of course, attention should be paid so that the set points are not outside the recommended operating range to avoid adverse long-term effects (e.g., thermal stress). As a result, the steam turbine and its operating variables are high enough in the priority list, relative to other plant components, and should be included in the unit optimization system.

Suggested Problem Formulation for Hammond

At Hammond, it has been decided to use the following models/tools for the various plant components:

- GNOCIS for the boiler
- Powergen developed model for the sootblowing system
- ESPERT for the ESP

These choices seem appropriate considering the objectives of the project and the available options at Hammond.

While there may be a number of predictive models for steam turbine performance, it was suggested that a combination of empirical correlations and a neural-network based system (e.g., GNOCIS) be used to model the steam turbine at Hammond. More specifically, the following control variables need to be added to the unit optimization system:

- Superheat and reheat steam outlet temperatures (setpoints)
- Main steam throttle pressure (setpoint)

Predictive model(s) need to be developed which portray the effects of these variables on unit heat rate and operating costs.

The steam turbine manufacturers provide empirical correlations which predict the impact of steam outlet temperatures and throttle pressure on the heat rate. These correlations offer a good approximation and could be used as the basis for predicting steam turbine performance. If a more detailed prediction were needed at Hammond, these correlations could be used as a starting point and then be improved by using a neural-network-based model.

Estimated Benefits Due to Unit Optimization

Considering the uncertainty of key operating parameters, it was estimated that the benefits from an overall unit optimization system at Hammond are in the range of $250,000 - $350,000 per year, assuming that NOx emission allowances are not taken into account. As the value of NOx emission allowances increases from zero to $1,500 per ton, the benefits from the optimization will increase, possibly as high as $2 million per year.

Summary

The objectives of this study were achieved. More specifically, "*Process X*" was determined, guidance was provided on overall problem formulation of the optimization system, and the benefits due to optimization were estimated. The evaluation of alternative options for the condenser (e.g., installation of an on-line cleaning system) proved valuable, demonstrating that the TPCO Guidelines could be used for such assessments.

The overall assessment was done over a period of five months (June-October 1999), much longer than the typical application of the TPCO Guidelines, because the spreadsheet had to be developed and calibrated. Such calibration is not expected in most applications. The level of effort for data collection, calibration, and analysis was approximately 40 labor days. The majority of this effort was for detailed calibration.

Typical applications of the TPCO Guidelines are expected to take two to five days for data gathering and inputting into the spreadsheet, depending on the familiarity of the plant engineer with the plant and the availability of the relevant data. For example, a plant performance engineer who is very familiar with all the plant components would not need more than a couple of days to collect and input the required data. However, if many different plant engineers need to be consulted, it may take longer.

Table 4-1 TPCO – Plant Performance and Cost Summary

Description of Item Provided	Units	Values			Values
C. PLANT-LEVEL SUMMARY – PERFORMANCE & COSTS		**Baseline**			**Optimum**
Gross Plant Output	MWgross	503.00			502.89
Net Plant Output	Mwnet	477.14			477.14
Net Plant Heat Rate	Btu/kWh	9,959			9,895
Plant Auxiliary Power	MW	25.86			25.75
Boiler Efficiency	%	86.61%			86.68%
Turbine Cycle Gross Heat Rate	Btu/kWh	8,182			8,138
Main Steam Flow	lb/hr	3,770,740			3,749,670
Emissions					
- SO2	lb/MMBtu	1.16			1.16
- NOx	lb/MMBtu	0.42			0.38
- CO2	lb/MMBtu	201.39			201.27
- Particulates	lb/MMBtu	0.09			0.08
Emissions					
- SO2	ton/yr	18,650			18,531
- NOx	ton/yr	6,683			6,017
- CO2	ton/yr	3,227,300			3,204,770
- Particulates	ton/yr	1,363			1,274
Change in Emissions					
- SO2	ton/yr	-			-119
- NOx	ton/yr	-			-666
- CO2	ton/yr	-			-22,530
- Particulates	ton/yr	-			-89
Revenues					
- Sale of electricity	K$/yr	Note 1			Note 1
- Sale of flyash	K$/yr	Note 1			Note 1
Total Operating Revenues	K$/yr	Note 1			Note 1
Difference in Total Operating Revenues	K$/yr	Note 1			Note 1
Difference in Emission Cost or Credit		Note 1			Note 1
- SO2	K$/yr	Note 1			Note 1
- NOx	K$/yr	Note 1			Note 1
- CO2	K$/yr	Note 1			Note 1
- Particulates	K$/yr	Note 1			Note 1
Difference in Total Emission Cost/Credit	K$/yr	Note 1			Note 1
Variable Operating Costs		Note 1			Note 1
- Fuel (Coal)	K$/yr	Note 1			Note 1
- Fly ash disposal	K$/yr	Note 1			Note 1
- Bottom ash disposal	K$/yr	Note 1			Note 1
- Boiler Makeup Water	K$/yr	Note 1			Note 1
- Limestone (FGD)	K$/yr	Note 1			Note 1
- FGD waste disposal cost	K$/yr	Note 1			Note 1
- Ammonia (Post-Combustion NOx)	K$/yr	Note 1			Note 1
Total Variable Operating Costs	K$/yr	Note 1			Note 1
Difference in Total Variable Operating Costs	K$/yr	Base			Note 1
Operating Income	K$/yr	Note 1			Note 1
Overall Impact on Oper. Income (Alt. - Base)	K$/yr	**Base**			Note 1
Overall Impact on Oper. Income (Alt. - Base)	$/kWnet-yr	Base			Note 1

1. Proprietary information removed.

Table 4-2 TPCO – Prioritization of Control Variables and Estimate of Benefits (Baseline)

	D. PRIORITIZATION OF CONTROL VARIABLES		Ranking	Impact of Each Variable on Operating Income			
	Ranking of Parameter Impacts	**Units**	**of Variable**	$ x 1,000	% of Overall	**Range (Low to High)**	
1	Coal Quality (Name & % in Blend, A/B/C)	Name & %	11	Note 1	0.0%	100% KY-HammB to 100% KY-HammB	
2	Excess Air Downstream of Economizer	%	10	Note 1	0.2%	27 to 19	
3	Air Heater Leakage	%	11	Note 1	0.0%	11 to 13	
4	Overfire Air Damper Setting	% Open	9	Note 1	0.4%	0% to 100%	
5	Superheater Spray Flow	lb/hr	8	Note 1	2.8%	0 to 75415	
6	Reheater Spray Flow	lb/hr	5	Note 1	13.4%	0 to 37707	
7	Main Steam Throttle Pressure	psig	2	Note 1	18.9%	2300 to 2420	
8	Main Steam Temperature	° F	1	Note 1	20.0%	990 to 1010	
9	Reheat Steam Temperature	° F	4	Note 1	16.2%	990 to 1010	
10	Cycle Makeup Water	% of MS Flow	3	Note 1	17.9%	0% to 1%	
11	Condenser Cleanliness	Fraction	6	Note 1	7.2%	0.55 to 0.70	
12	Velocity of Water in Condenser Tubes	ft/sec	7	Note 1	3.0%	6.0 to 7.0	
13	Limestone Stoichiometry	moles CaCO3/mole SO2	11	Note 1	0.0%	#N/A	
14	Ammonia-to-NOx Ratio	moles NH3/mole NOx	11	Note 1	0.0%	#N/A	
	OVERALL IMPACT OF CONTROL VARIABLES (TOTAL IMPACT ON INCOME)			Note 1	**100%**	Note 1	
	TABLE D NOTES:						
	1. All costs are absolute values						
	2. Rank is based on absolute value of costs						

1. Proprietary information removed.

Table 4-3 TPCO – Prioritization of Control Variables and Estimate of Benefits (with NOx Credits)

D. PRIORITIZATION OF CONTROL VARIABLES		Ranking	Impact of Each Variable on Operating Income		
Ranking of Parameter Impacts	**Units**	**of Variable**	$ x 1,000	% of Overall	**Range (Low to High)**
Coal Quality (Name & % in Blend, A/B/C)	Name & %	11	Note 1	0.0%	100% KY-HammB to 100% KY-HammB
Excess Air Downstream of Economizer	%	2	Note 1	29.3%	27 to 19
Air Heater Leakage	%	11	Note 1	0.0%	11 to 13
Overfire Air Damper Setting	% Open	1	Note 1	44.8%	0% to 100%
Superheater Spray Flow	lb/hr	10	Note 1	0.7%	0 to 75415
Reheater Spray Flow	lb/hr	7	Note 1	3.5%	0 to 37707
Main Steam Throttle Pressure	psig	4	Note 1	5.0%	2300 to 2420
Main Steam Temperature	° F	3	Note 1	5.2%	990 to 1010
Reheat Steam Temperature	° F	6	Note 1	4.3%	990 to 1010
Cycle Makeup Water	% of MS Flow	5	Note 1	4.5%	0% to 1%
Condenser Cleanliness	Fraction	8	Note 1	1.9%	0.55 to 0.70
Velocity of Water in Condenser Tubes	ft/sec	9	Note 1	0.7%	6.0 to 7.0
Limestone Stoichiometry	moles $CaCO_3$/mole SO_2	11	Note 1	0.0%	#N/A
Ammonia-to-NOx Ratio	moles NH_3/mole NOx	11	Note 1	0.0%	#N/A
OVERALL IMPACT OF CONTROL VARIABLES (TOTAL IMPACT ON INCOME)			Note 1	**100%**	Note 1

1. Proprietary information removed.

5
UNIT OPTIMIZATION PACKAGE

Problem Definition and Solution Approach

The overall goal of the project at Hammond 4 was to develop and demonstrate quasi-steady state, online, optimization techniques to power plant processes and to the unit as a whole. An important part of the project was the design of the unit optimizer. At least two approaches could have been taken.

Single Optimizer/Model Approach - One possible approach would be to cast it into a single criteria optimization problem rather than a multi-criteria optimization problem, creating a single model of the processes to be optimized and then optimizing this model (Figure 5-1).

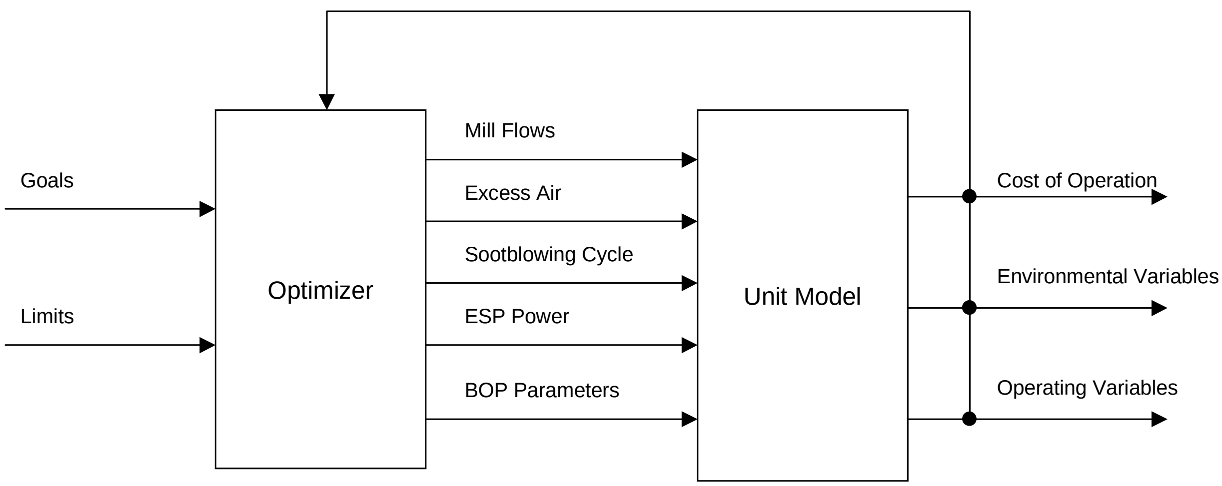


Figure 5-1 Single Optimizer/Model Approach

If the model and optimizer are sufficiently robust, this approach has the potential to produce a global optimum operating point for the unit process. Although attractively simple, straightforward, and appealing in its approach, this method presents several significant problems. Since there is one model, an expansive and complex model will likely be required to portray the process to accuracy necessary to obtain optimum performance. Generally, complexity tends to lead to fragility and increased likelihood of failure. If there are problems with any input, the model can fail. A consequence of creating this type comprehensive optimizer is that if a particular sub-process (such as the boiler) cannot be optimized unless the entire model is operating and valid. The model will have numerous inputs representing the "manipulated" control variables (excess air, mill coal flows, sootblowing steam flows, ESP power levels, etc) and non-controllable noise variables (ambient conditions, coal properties, cooling water temperatures, etc). For a typical power plant, the number of important inputs could conceivably be 50 to 100 variables or more (there are sometimes up to 50 inputs in a boiler optimizer). In addition to tending to increase the fragility of the model, increasing the inputs increases the degree of difficulty of finding the optimum solution. Also, the data required to develop the model increases exponentially with the number of model inputs unless measures are taken to decouple the input variables through either (1) apriori process knowledge or (2) model simplification. Another problem is that this approach effectively forces similar treatment of possibly dissimilar processes more appropriately handled by different optimization methods.

Hierarchical Optimizer/Model Approach - Another possible approach is to break down the process into a collection of optimizers in which there is a top-level optimizer and subsidiary process optimizers (Figure 5-2). This hierarchical approach, although structurally more difficult to implement than the more straightforward single optimizer approach, offers a number of potential advantages:

- The models are reduced in scope mitigating the difficulty of developing and maintaining each model.
- The number of inputs to each model is significantly less than the global model thereby reducing the likelihood of model collapse due to bad inputs.
- Each model should be more robust (precision and reliability) than the global model.
- Each process optimizer functions independently but with guidance (i.e. constraints and goals) imposed from the unit optimizer. This partitioning gives great flexibility and robustness to plant problems.
- If a process optimizer-model fails, the system could be designed such that the other process optimizer-models can continue to operate, although the unit optimization will run at reduced functionality. Optimizer modules can be added or removed as necessary only causing partial degradation of the unit optimization.
- The optimizer and model for a particular process can be specifically selected to best match process characteristics.
- The hierarchical approach facilitates module testing and adding new functionality is greatly simplified.

- This method allows the use of existing, commercial process optimization platforms such as GNOCIS and NeuSight.
- The formation of representative cost model should be more straightforward since the input variables to this model are more closely related to actual cost. For example, it is easier to obtain the functional relationship of incremental operating cost to station service than it is of incremental operating cost to excess oxygen level.
- The same framework could be carried both upward (e.g. to the plant level) and downward (e.g. to the burner level) in the process (Figure 5-3) and allows adding process details as needed.

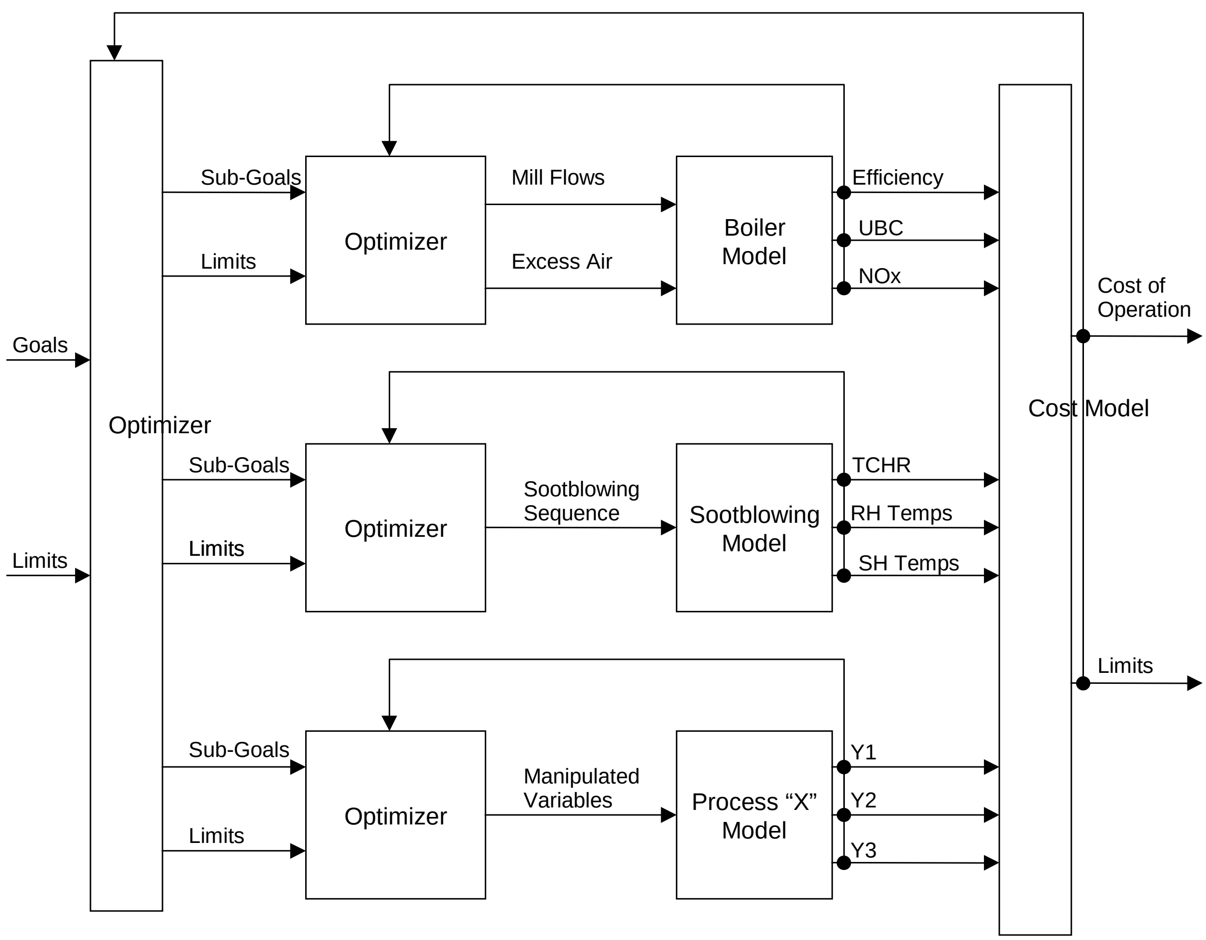


Figure 5-2 Hierarchical Model Approach

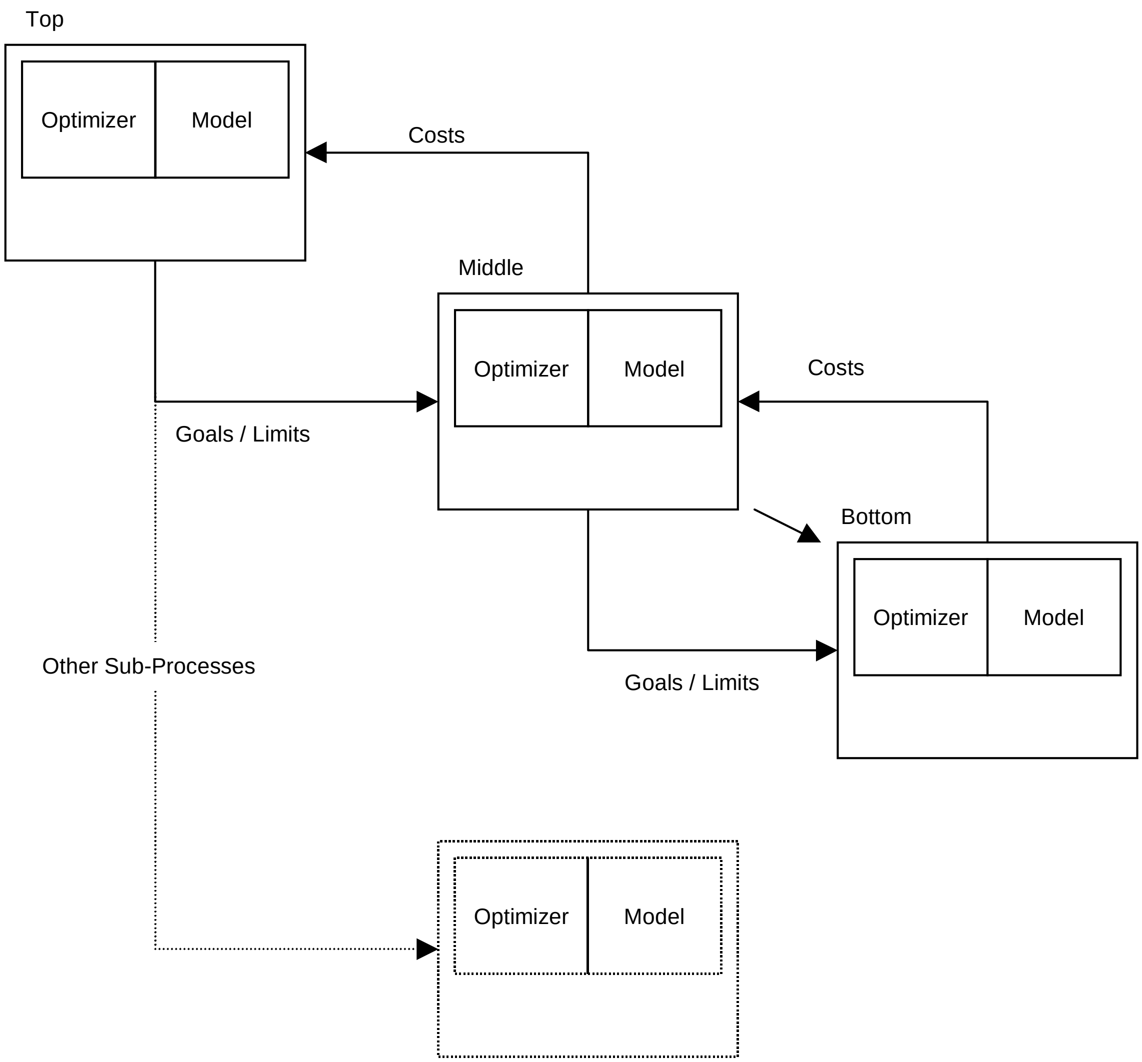


Figure 5-3 Expansion of Hierarchical Model Approach

Considerations in this approach are (Figure 5-4):

- The roll-up of costs from the lower levels to the higher levels
- The roll-down of effective goals and limits from the higher levels to the lower levels
- Reconciliation of recommendations from different sub-optimizers

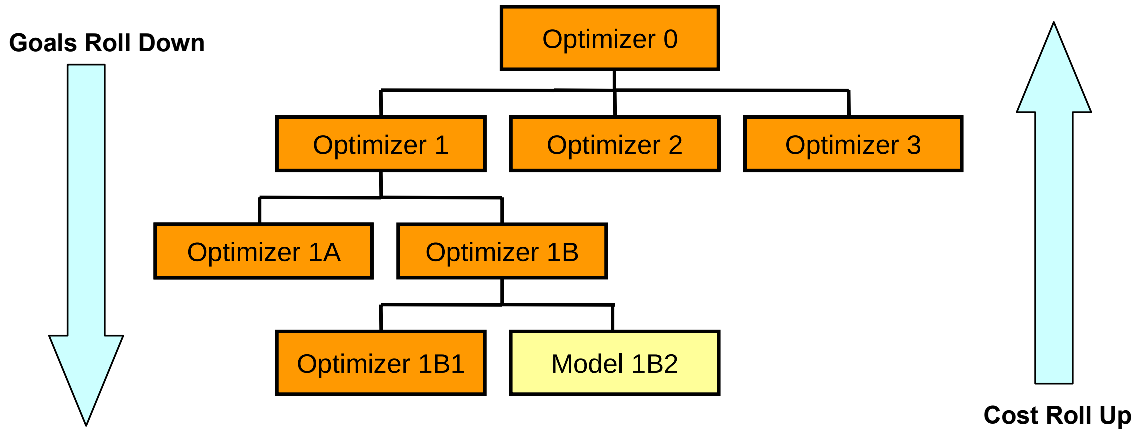


Figure 5-4 Goal and Cost Propagation

This hierarchical approach cast itself naturally to a multi-criteria optimization problem. As discussed elsewhere, there are numerous methods that may be utilized to attack multi-criteria optimization problems. One possible approach is to formulate the global optimizer objective function such that a common metric (as an example, cost) is created from the sub-optimizers objective functions. This approach corresponds to the weighted-sum method. Since the sub-optimizers may share decision variables, if run independently, it is likely that the recommendations from each optimizer will need to be reconciled. For example, considering the system in Figure 5-4, if *Optimizer 1* optimum solution is $\hat{X}_1$, *Optimizer 2* optimum solution is $\hat{X}_2$, and *Optimizer 3* optimum solution is $\hat{X}_3$, what is the optimal solution for the system? One possible approach, though not necessarily yielding an optimal solution for the system, is to average recommended solutions to get a new operating point:

$$\hat{X} = \frac{\hat{X}_1 + \hat{X}_2 + \hat{X}_3}{3}$$

and this decision vector would be implemented.

Software Overview

The three major efforts in the unit optimization package were:

- Development of a software framework to coordinate optimizers and sub-optimizers (*Unit Optimization Framework*).
- Development of global optimizer algorithm and software that potentially greatly reduces the number of manipulated variables (*Powergen Optimizer*).
- Inclusion of the Synengco SentinentSystem Global Optimizer software (*SentinentSystem*).

These efforts are discussed in the following paragraphs.

Unit Optimization Framework

The objective of this task was to develop a framework, including software, to coordinate multiple, hierarchical optimizers. Conceptually, this is shown in Figure 5-5. The most important aspects of the software are as follows:

- Optimizers are encapsulated as software objects providing a common interface.
- Various optimizer technologies can be accommodated.
- The framework supports hierarchical optimizers and models, resolving common inputs and outputs.
- The approach is flexible in that the optimizers being configured through initialization files and not necessarily programmatically (though it can be).
- The optimizers may be distributed across multiple computers running the same or different operating systems.

The unit optimization software in relation to other project software is shown in Figure 5-6 and the major components of the software are shown in Figure 5-7. Details of the software are provided in the appendices and a brief summary follows.

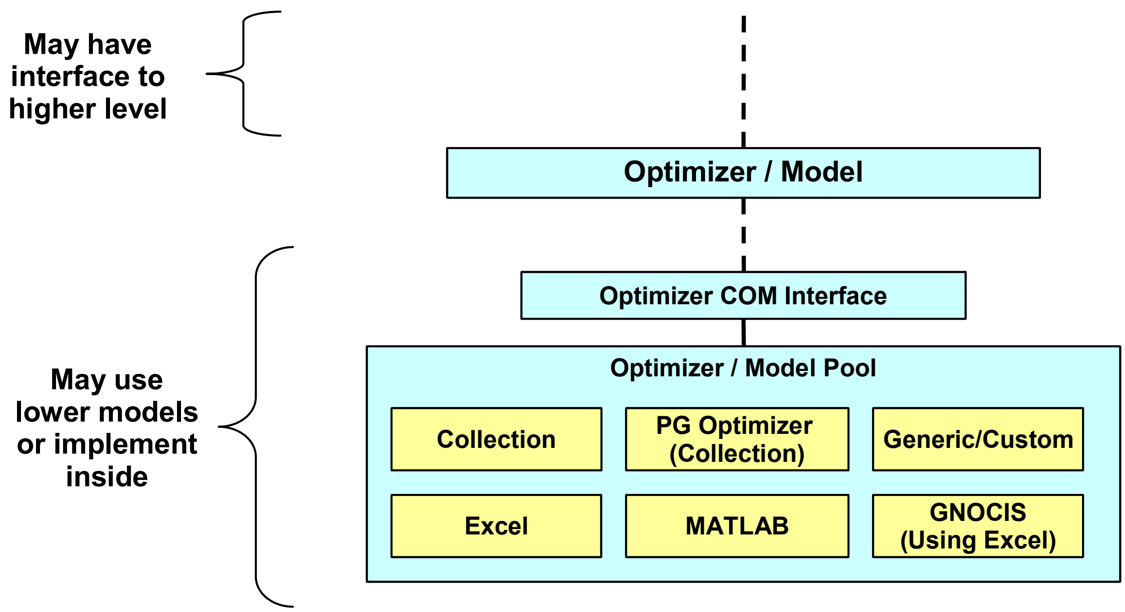


Figure 5-5 Software Structure

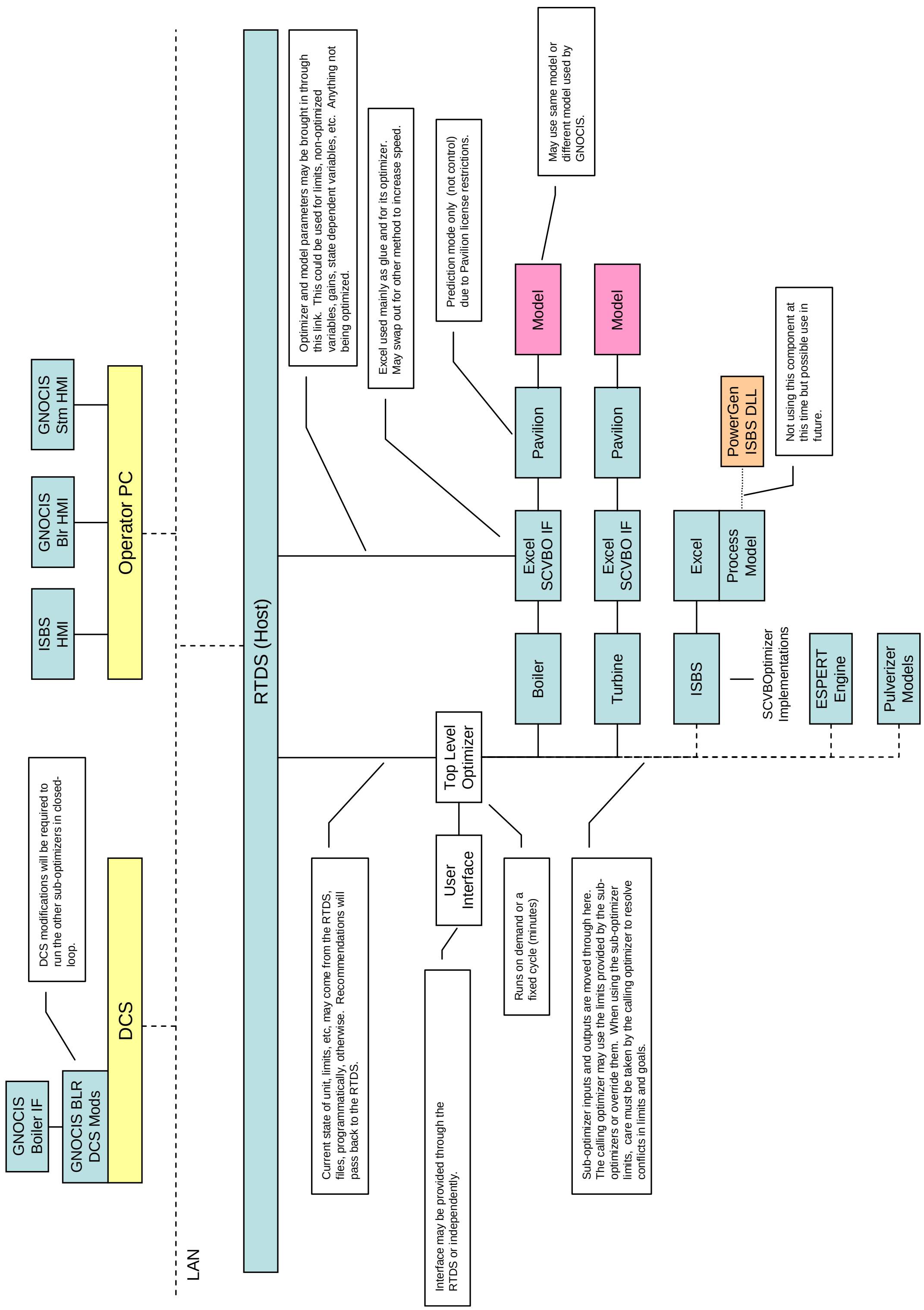


Figure 5-6 Unit Optimization Software in Relation to Other Project Components

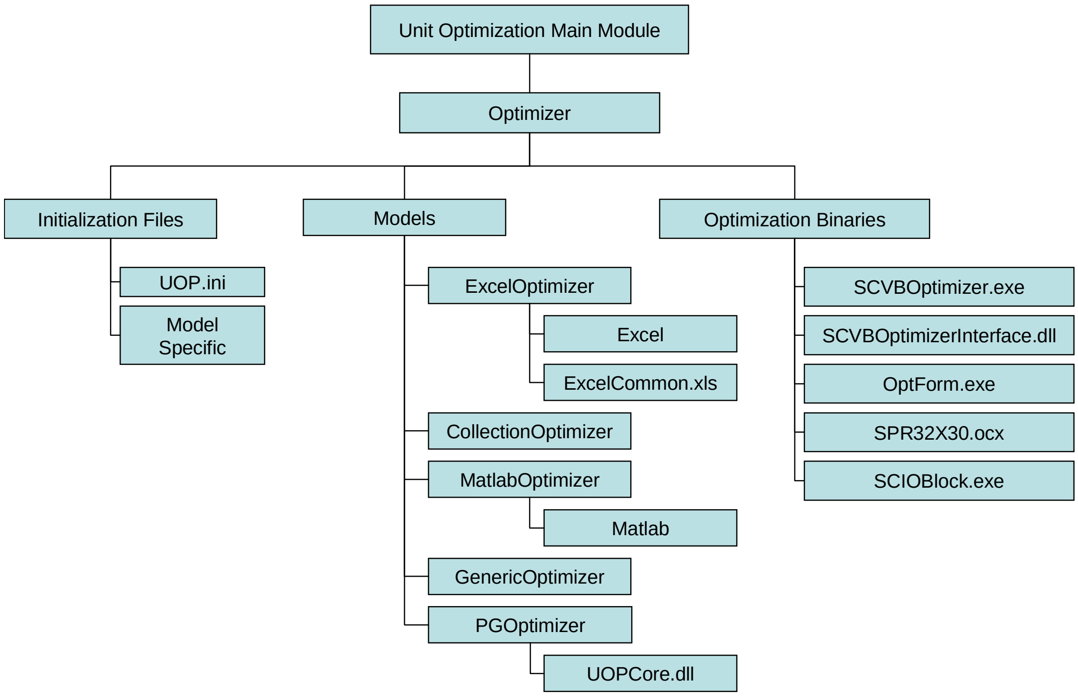


Figure 5-7 Unit Optimization Framework Software Overview

Optimizer Objects

An optimizer object consists of the following:

- Inputs – A set of extended inputs to the model
- Model – A relationship between the inputs and the outputs
- Outputs – A set of extended outputs from the model
- Optimizer – A method for determining the optimal set of inputs with respect to the cost
- Cost – A scalar metric which is a function of the model outputs
- Sub-Optimizers – Zero or more sub-optimizers that make up this optimizer

The framework supports multiple optimizer objects. These optimizers may be implemented in various programming languages. The optimizers support a common interface, a subset of the methods being listed in Table 5-1. The inputs, outputs, and the cost are themselves objects of the *SCInput*, *SCOutput*, and *SCCost* class. The members of the *SCInput* class are shown in Table 5-2 (the outputs and cost definitions are similar). The inputs and outputs may also be put into collections (*SCIOBlock* class). More details on the class definitions may be found in the appendices.

Table 5-1 Subset of SCVBOptimizer Class Methods

Method	Description
Run	Run the model based on the current set of inputs
Optimize	Optimize the model based on the current set of inputs
getCost	Get current cost
getReachedMin	Get flag which indicates whether optimizer reached minimum
setInputs	Set the current set of inputs
getInputs	Get the current set of inputs
getOutputs	Get the current set of outputs
setOutputs	Set the current set of outputs
getSubOptByName	Get sub-optimizer by name
setDebugState	Enable/disable debugging state

Table 5-2 SCInput Data Members

Member	Description
Name	Name given to the input
Value	Current value of the input
UpperBound	Upper bound for the input (for optimization)
LowerBound	Lower bound for the input (for optimization)
OptimumValue	Optimum value of the input from an optimization
Type	Type of the input

SCVBOptimizer

The *SCVBOptimizer* is an implementation of an optimizer and has been used exclusively to date. The *SCVBOptimizer* acts as a wrapper object for one of five different types of models: *Excel*, *Matlab*, *Collection*, *PGCollection*, and *Generic*.

Excel - This type uses Microsoft's Excel as the optimization interface. Calculations can be entered into a spreadsheet and automated through the SCVBOptimizer interface. In order for a spreadsheet to be used in the SCVBOptimzer Excel model type, it must have two macro functions in the workbook. The *LocalRun* routine is called by the optimizer container when the model is run. This routine should contain any other calculations necessary to complete a model run (from the spreadsheet). The *LocalOptimize* routine is called when the SCVBOptimizer calls its optimization routine, and should contain any steps necessary to complete the optimization of the contained spreadsheet model. Different optimizers may be used through specification of fully qualified path and filename to the workbook that contains the model calculations.

Generic - This model is currently not used, but is intended to provide a template for future models that will be coded in Visual Basic, and contained within the *SCVBOptimizer.exe* binary.

Matlab - This model uses The Mathwork's Matlab software as the calculation engine. Similar to the *Excel* model type, a Matlab style class definition is specified in the initialization file.

Collection - This model may contain an agglomeration of several SCVBOptimizer objects. The intention of this type is to create a set of models that interact with each other to provide the optimum point among all models. This type may have an established algorithm that resolves conflicts among sub-models. It can also be used as simply a container for multiple model sets. Collections may contain collections.

PGCollection - This is a special version of the *Collection* model that uses the Powergen optimization routine to resolve conflicting optimum points among the contained models. Otherwise, it is similar to the Collection model.

A simple interface allows the user to view the current inputs, outputs, cost, and sub-optimizers that are applicable to the current model being executed.

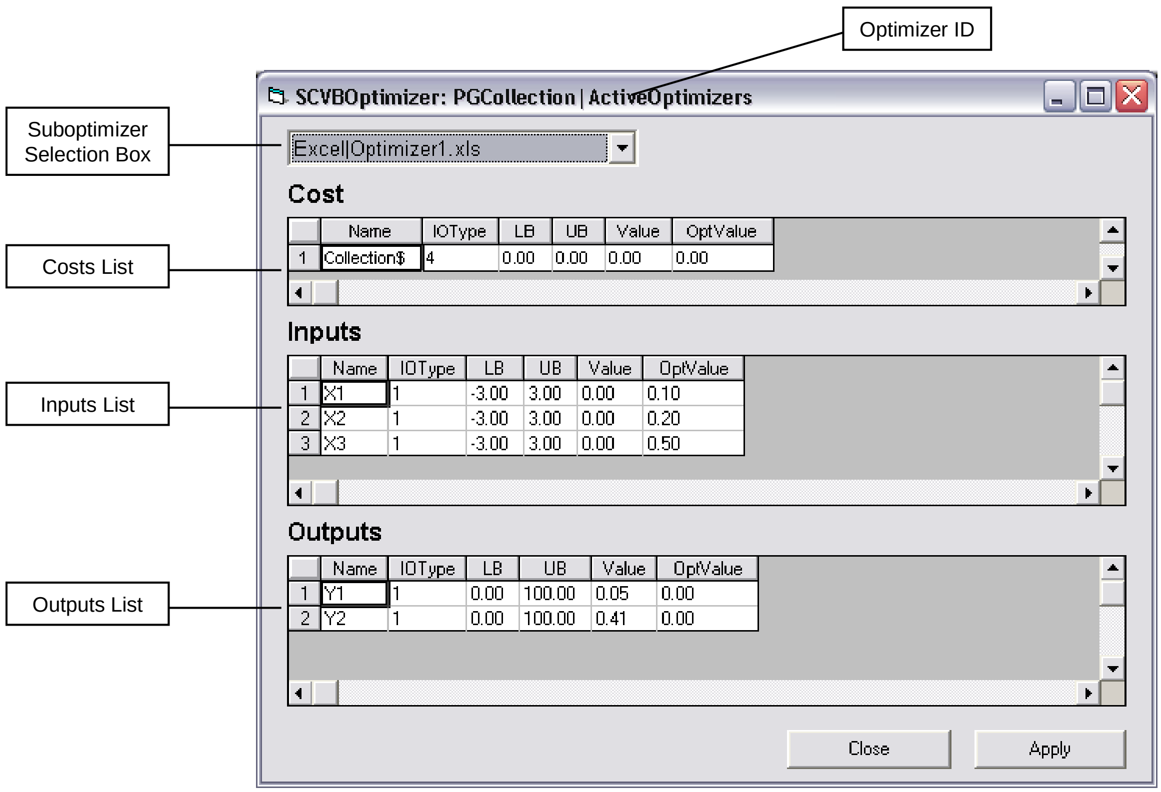
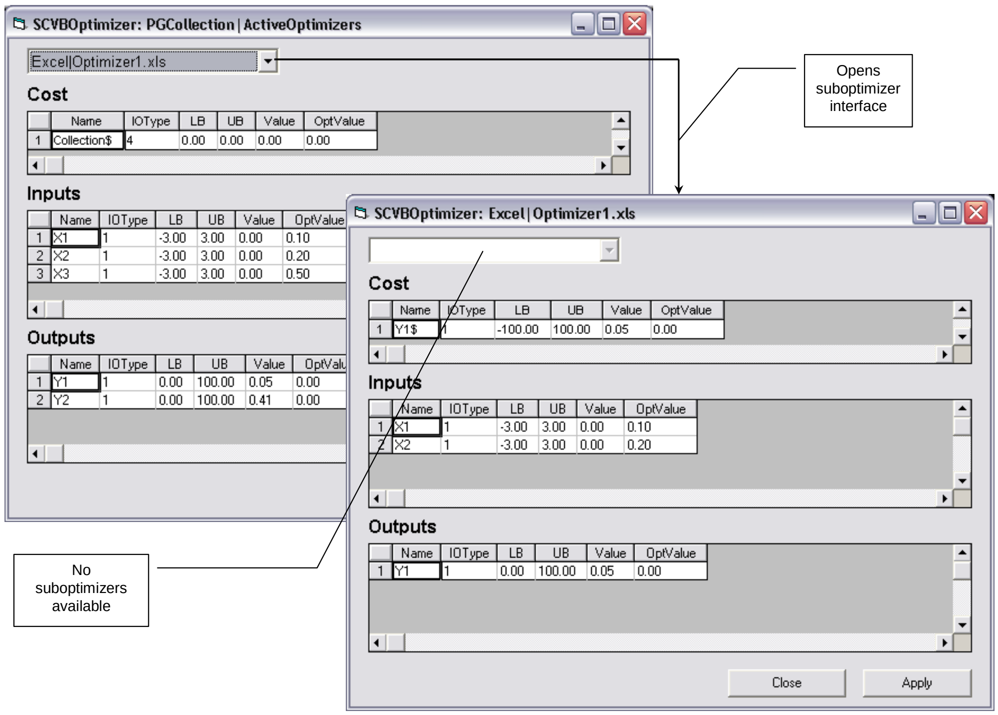


Figure 5-8 SCVBOptimizer User Interface



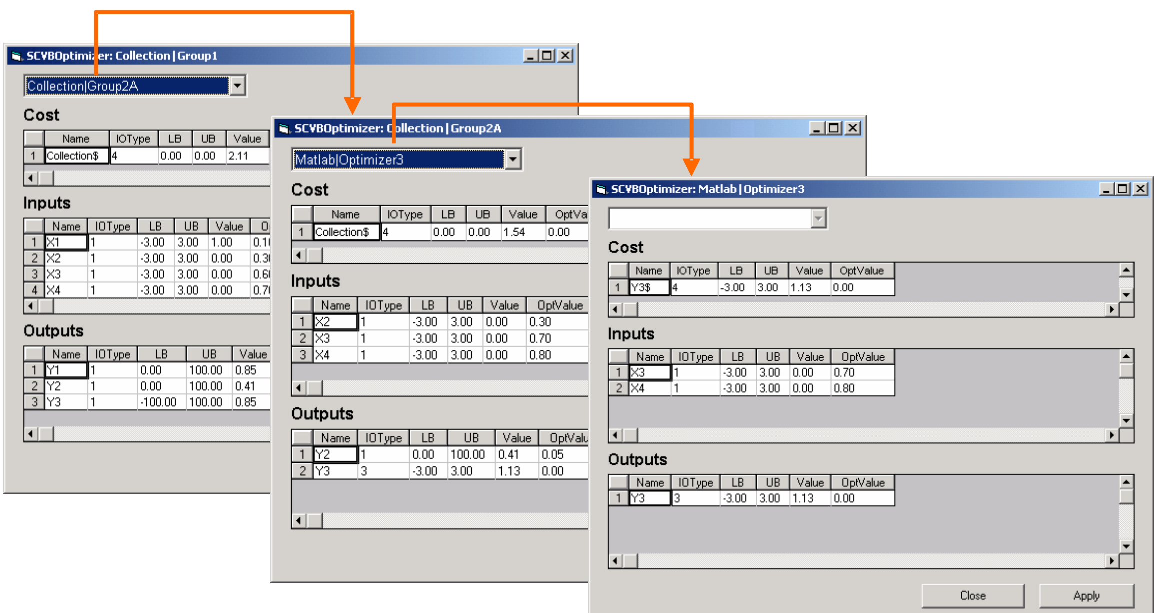


Figure 5-9 Cascading of Optimizers and Models

Initialization File

Selection of the optimizers and other options are performed though an initialization file. An example of an initialization file for the optimizer structure in Figure 5-10 is provided in Figure 5-11. The top level optimizer references three sub-optimizers (Optimizer1, Optimizer2, and Collection1). Collection1 is a collection of optimizers consisting of Optimizer3 and Optimizer4. All four optimizers and the collection are implemented using the *SCVBOptimizer* client, with Optimizer1 and Optimizer3 using Excel and Optimizer2 and Optimizer4 using Matlab.

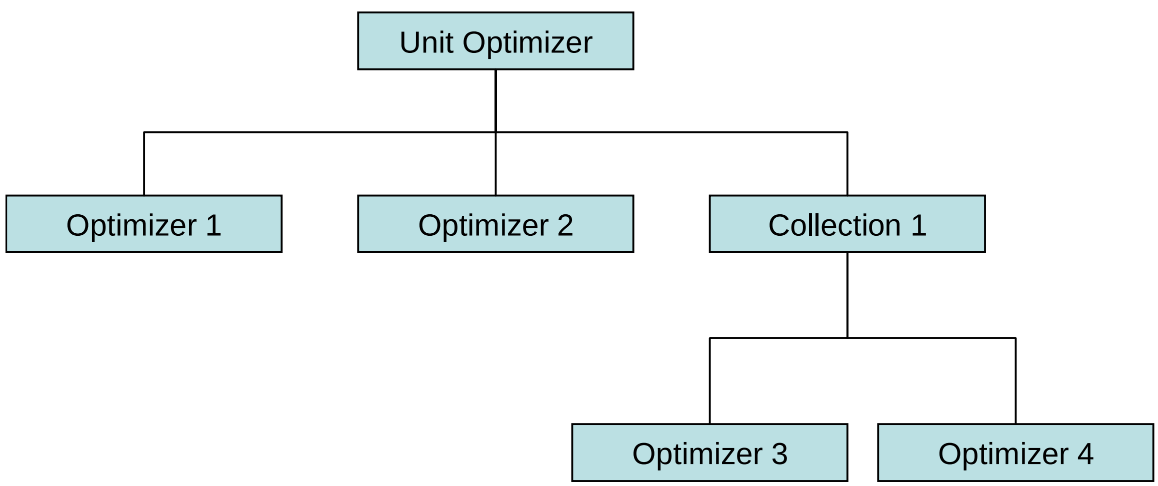


Figure 5-10 Model Structure for Example Initialization File

```
[General]
sRTDSHost = 127.0.0.1
sLoadTag = "4CP001_I:MAIN_4.PNT_3"

[ActiveOptimizers]
Optimizer1
Optimizer2
Collection1

[Collection1]
DEBUG = 1
DATA = "Collection|Collection1A"

[Collection1A]
Optimizer3
Optimizer4

[Optimizer1]
DEBUG=1
COMID = "VBOptimizers.VBOptimizer"
DATA = "Excel|Optimizer1.xls"

[Optimizer2]
DEBUG=1
COMID = "VBOptimizers.VBOptimizer"
DATA = "Matlab|Optimizer2"

[Optimizer3]
DEBUG=1
COMID = "VBOptimizers.VBOptimizer"
DATA = "Excel|Optimizer3.xls"

[Optimizer4]
DEBUG=1
COMID = "VBOptimizers.VBOptimizer"
DATA = "Excel|Optimizer4 "
```

Figure 5-11 UOP Initialization File

Powergen Algorithm

The problem of several local optimizers giving conflicting control settings for the unit can only be solved by considering the total unit costs or objectives. Whereas each local optimizer has its own cost function to minimize[1] and only has knowledge of its own local restricted environment, the unit optimizer has to integrate the advice from all the local optimizers to produce an overall control strategy. In order to do this, some means of compromising individual advice must be found. Since the objective functions for the local optimizers involve different high level plant variables, a common factor needs to be found to enable appropriate recommendations to be made. This factor has to be total unit costs and a unit cost function has to be defined in terms of high level plant variables such as NOx, carbon-in-ash, boiler efficiency, etc. It is important that costs can be associated with the high level plant variables otherwise it is not possible to fully define a unit cost function.

The definition of this cost function may pose no problems in some cases and be quite difficult in other situations and is a custom issue. In order to optimize this unit cost function it can be considered, via the high level plant variables, to be a function of the plant control variables and optimized using a conventional optimizer package. This involves using the local optimizers as predictors of the high level plant variables and a considerable amount of numerical traffic across the network linking the various local optimizers. The number of controllable variables is typically 20 ~ 30 and numerical evaluations of gradients impose a high computational burden on the network system. Note that with this approach, the optimization functionality of the local optimizers is not used, the main computational burden being with the unit optimizer.

Another potential problem with this approach is local minima. While the unit cost function, expressed in high level plant variables, may have a well defined single minimum, when expressed in terms of low level plant variables, multiple minima may arise. This is more likely to happen as more high level plant variables are included in the unit cost function. For these reasons, optimizing the unit cost function directly in terms of the low level plant control variables is not considered to be the most appropriate method of achieving the unit optimization.

In order to reduce the amount of calculation involved with the unit optimization, the effective dimension of the unit optimization calculation needs to be reduced from the number of plant control variables (*Design Space*) to substantially less. The observation that if there was no conflicting advice, then all local optimizers would give the same plant variable recommendations provides an indication of how this may be achieved.

[1] In this report, we assume a minimization. Maximization is achieved by negating the appropriate cost terms.

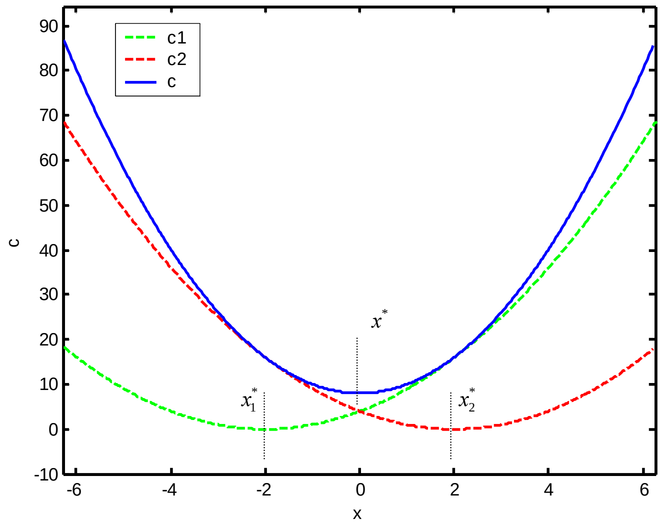


Figure 5-12 Example of Unit Cost for a M = 2, N = 1 System

A simple example of this is provided in Figure 5-12 for which the local objective functions are:

$$\textit{Optimizer 1:}\quad \mathrm{y}_1 = f_1(x) = (x+2)^2$$

$$\textit{Optimizer 2:}\quad \mathrm{y}_2 = f_2(x) = (x-2)^2$$

and the unit cost is:

$$\mathrm{c} = f_1(x) + f_2(x)$$

The optimum solution for *Optimizer 1* is x = -2, the optimum solution for *Optimizer 2* is x = +2, and for this system, clearly the unit optimum lies in the design space between the local optimizers optimal solutions.

Another representation of the disparity between three local optimizers (or sub-optimizers) is shown in Figure 5-13. In this figure, the c_i represent the value of the unit (global) objective function at optimum design points for the individual local optimizers: X_1 is the optimum design point for *Optimizer 1*, X_2 is the optimum design point for *Optimizer 2*, and X_3 is the optimum design point for *Optimizer 3*. A measure of the conflict is given by the size of the shaded triangle.

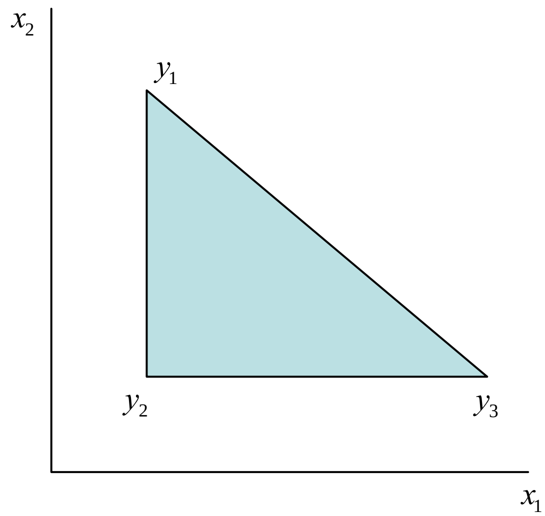


Figure 5-13 Representation of Conflict in Advice in Design Space for M = 3, N = 2 System

As with the prior example, it is not unreasonable to envisage that the unit optimum point lies somewhere in the shaded region, the exact position depending on the unit objective function. It turns out that this is not quite the situation when the y_i refer to the fully optimized local solutions but a modification of this idea can be used to solve the optimization problem as described in the next section. The situation shown in Figure 5-13 is that where the number of local optimizers is greater than or equal to the dimension of the plant control variables. Clearly, if there were a much greater number of sub-optimizers than the dimension of the design variables, then the various local optimal plant coordinates would be expected to define the boundary of the actual high level optimum, similar to the triangle in Figure 5-13. This is not strictly mathematically true for arbitrary functions but is expected to be true for reasonable global cost functions, such as a weighted-sum global cost.

However, if the number of local optimizers is less than the plant control variable dimension, then the unit optimum value is not necessarily in the *convex set*[1] defined by the local optimum points. Referring to Figure 5-13, if there were only two local optimizers, yielding the local optimums X_1 and X_2, then the high level optimum need not lie on the line joining X_1 and X_2. However, if the X_i are not the locally optimum plant coordinates, but are coordinates of the local optimizers that reduce their output compared to current values, then an iterative scheme can be constructed which converges to the solution. This is described in the next section.

[1] A set is convex if, given two points in the set, the straight-line segment joining the two points is also contained in the set. For example, the set of all real numbers from 0 to 5 {(0,5)} is convex, whereas the union of real numbers of 0 to 2 and 3 to 5 {(0,2) ∪ (3,5)}is not.

Description of Algorithm

The unit objective function is expressed in high level plant variables which are themselves outputs of the local optimizers. If there were no constraints or conflicts in manipulating these high level variables, then the unit optimization could be carried out using these variables with no reference to the plant control variables. Only after the final values of the high level variables have been obtained would the plant control variables be specified in order to achieve these desired high level values. This situation is not generally the case and we consider a situation where the unit optimization is carried out subject to constraints on the high level variables. These variables are constrained to lie within a hyper-sphere centered on the current values of the variables. This constrained optimization is carried out assuming that the high level variables are independent. An example is shown in Figure 5-14.

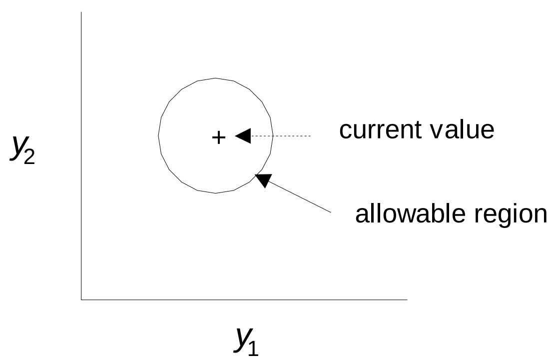


Figure 5-14 Schematic Showing Restrictions Around the Current Value in Criterion Space

This constrained minimization provides changes in the high level plant variables (y_1 and y_2). These changes are then used as constraints for the local optimizers which are required to carry out their own specialist optimization using the low level control variables, *X*, subject to the change in their output variables (y_1 and y_2) being no greater than that specified by the unit optimization.

If the required changes in the high level variables are not in conflict then the plant control values calculated by the local optimizers will be identical. However in the more realistic option of some conflict between the different plant settings a similar situation to that shown in Figure 5-13 arises. Again, the shaded area is a measure of the degree of disparity and for well-defined unit objective functions, we expect the minimum of the unit objective function to lie within this shaded area. Note that unlike the situation described above where the points X_i represented the final locally optimized plant setting for each optimizer with the unit cost function not being involved at all, the situation here involves the unit objective function through the constraints put on the local optimizers. The constrained unit optimum is assumed to be in the shaded region and this region is searched for the unit optimum, which may occur on the boundary. Mathematically, we assume that the individual plant coordinates returned by the local optimizer are the extreme points of a convex set and then search this set for the optimum solution.

The vector of inputs that satisfy the individual local optimizer criteria set by the unit optimizer are denoted by X_i and represented by the red dots in Figure 5-15. The convex hull, *C*, of the X_i is the set of all points X_θ which may be written as:

$$C \equiv \left\{ X_\theta = \sum_{i=1}^{i=m} \phi_i X_i \quad , \qquad \sum_{i=1}^{i=m} \phi_i = 1, \quad \phi_i \geq 0 \right\}$$

Alternative, less formal, though equivalent definitions of a complex hull are:

- The convex hull of a set of points is the smallest convex set that contains the points.
- The convex hull is the intersection of all convex sets containing the points.
- The convex hull is the area contained by a rubber band wrapped around the "outside" points.

In Figure 5-15, the set of points X_i are known as the extreme points of the set and are the red points and the convex hull defined by the X_i is represented by areas marked with "1".

The set of plant input constraints form a hyper-cube represented by the yellow rectangle in Figure 5-15. An important property of the convex set is that if the extreme points of the set satisfy a set of upper and lower bound constraints, then any member of the set also satisfies these constraints. This is demonstrated in Figure 5-15 where it is clear that the green convex set is wholly contained within the yellow rectangle representing the plant constraints. Thus any X_θ defined by the above equation automatically satisfies the plant constraints. The dimension of the search space is equal to the number of local optimizers minus one. This is because the constraint on ϕ removes one dimension from the search space. A full optimization is now carried out using ϕ as the search variables together with the appropriate constraints.

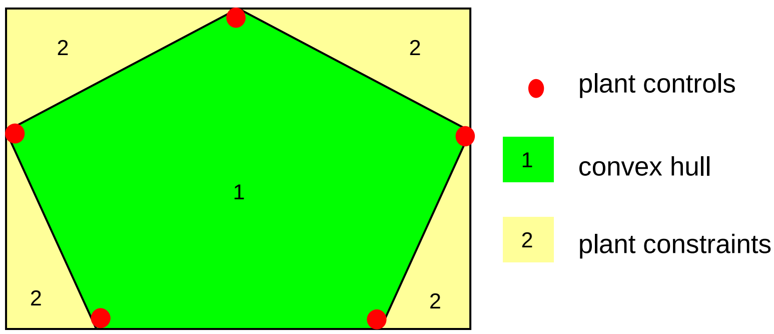


Figure 5-15 Schematic Showing the Convex Set within the Set of Plant Input Constraints

Once the minimum has been found the new plant values are used to calculate new high level variables and the procedure repeated until no further improvement in the objective function is obtained. If, during the iteration, any plant variable reaches an upper or lower bound then this is enforced by the local optimizers. If a high-level plant variable reaches its minimum then this is subsequently enforced by the high level optimization. During the iteration, if the new value of the unit objective function obtained from the current convex set is greater than the previous value, the hyper-sphere radius is reduced and a new convex set calculated. This is repeated until the new objective function is below the previous.

It has not been possible to prove that this iteration converges to the unit optimum but it is possible to show that if this iterative scheme is regarded as the mapping:

$$x_{n+1} = A(x_n)$$

then the optimum value, $\hat{x}$, is a fixed point of the above equation. The iteration is terminated when a convergence criteria based on the last three values of the objective function is satisfied.

In order to test the above algorithm, Powergen developed prototype code in Matlab, a listing of which is provided in the appendix.

Application Example

A simple example was constructed using two high-level plant variables and a three dimension plant variable. In this example, the high-level plant variables are called *NOx* and *Eff* (representing efficiency) and are given by the following formula:

$$NOx = \frac{\sqrt{x_1}\left(x_2 + x_3^{1.5}\right)}{1.0e4}$$

$$Eff = \frac{x_1 (x_3 - 200)^2 + \dfrac{1.0}{\sqrt{x_2}}}{1.0e6}$$

The high-level cost function used was:

$$UC = \lambda \cdot NOx^2 + (1 - \lambda) \cdot Eff^2$$

This is a simple cost function combining *NOx* and *Eff*. In practice, if efficiency were expressed in percent then *Eff* would be 100 – *Eff*, since the aim is to maximize it. The lower and upper limits on the *x* were set to (200, 100, 50) and (500, 600, 400), respectively. Initially, only the components x_2 and x_3 were allowed to vary, x_1 being fixed at 500. For this example, $\lambda = 0.5$ and the optimum is at $\hat{x}$ = (500, 100, 134.0) for which $NO_X = 3.69$, *Eff* = 2.17, and *UC* = 9.19.

The constraint variable in Figure 5-16 is the radius of the constraint hyper-sphere used in the high level optimization. In order to clarify the algorithm, the steps for this example will be explicitly given. Note only the second and third components were allowed to vary in this example. The constraint radius was set at 20.

Start x = (500, 346, 245), $NO_X = 9.35$, *Eff* = 1.01, *UC* = 44.2

Step 1.

A. High level optimization.

Carry out high level optimization - new *NOx* = 4.9, new *Eff* = 0.53.

This was done using the MATLAB function *fmincon* which implements the sequential quadratic programming algorithm.

B. Local optimizer optimization.

These new high level values are set as lower bound limits for the local optimizers which are then run for both high level variables. The local optimizers again use the MATLAB function *fmincon.* This then gives a set of plant coordinates for each local optimizer, namely *NOx* plant values (342, 150.7), *Eff* plant values = (346, 232.6). These form the vertices of the first convex set which is then searched for the unit cost minimum. This corresponds to line 1 in Figure 5-18.

C. Optimization over the convex set.

Carry out full optimization over above convex set to get the optimum coordinates (342, 150.7).

This optimization is also done with *fmincon.* The algorithm uses *fmincon* three times during each iteration, first with the high level optimization using the high level plant variables, second, for each local optimizer using plant control variables, and third for optimization of the unit cost function using the convex set variables θ_i. This then gives *NOx* = 4.9 and *Eff* = 1.2, *UC* = 12.75.

Step 2.

A. High level optimization.

Carry out high level optimization - new *NOx*=0.56, new *Eff* = 0.14.

B. Local optimizer optimization.

These new high level values are set as lower bound limits for the local optimizers which are then run for both high level variables. This then gives a set of plant co-ordinates for each local optimizer, namely *NOx* = (100 ,50), *Eff* = (341.98, 183.17), these form the vertices of the second convex set which is then searched for the unit cost minimum. Note the *NOx* coordinates are now at their lower limit. This corresponds to line 2 in Figure 5-16.

C. Optimization over the convex set.

Carry out full optimization over the above convex set to get the optimum coordinates (246.8, 130.8). This then gives *NOx* = 3.9 and *Eff* =2.39, *UC* =10.46.

Step 3.

A. High level optimization.

Carry out high level optimization - new *NOx* = 1.01 new *Eff* = 4.0e-009.

B. Local optimizer optimization.

These new high level values are set as lower bound limits for the local optimizers which are then run for both high level variables. This then gives a set of plant coordinates for each local optimizer, namely *NOx* =(100,50), *Eff* = (246.88,199.98).

C. Optimization over the convex set.

The convex set which is then searched for the unit cost minimum. Note the *NOx* coordinates are now at their lower limit. This corresponds to line 3 in Figure 5-16.

Step 4.

A. High level optimization.

Carry out full optimization over the above convex set to get the optimum coordinates (180.6, 132.29). This then gives *NOx* =3.8 and *Eff* =2.29, *UC* = 9.87.

This procedure is repeated until convergence is achieved. The *x* coordinate comes up against a lower bound and the convex set 'swings' around to converge on the optimum solution. Note how the objective function reduces rapidly in value and then changes very little for subsequent iterations. Figure 5-16 shows the various convex sets used for each iteration, the dashed line being the actual trajectory taken by the iterates. The converged solution is point 11 (500,100,117,6). Also note in the above that the maximum change in $NOx^2 + Eff^2$ is 20, corresponding to the hyper-sphere constraint of 20 that was used in this example. If the individual components of the unit cost function are of different magnitudes they can either be suitably scaled prior to use in the unit cost function or if not, the hyper-sphere constraint can be replaced by a hyper–ellipsoidal constraint to account for the differences in scale. This is easily incorporated in the non-linear constraint formulation.

The method was then tried allowing all components of *x* to vary. For this configuration, the optimum is $\hat{x}$ = (200,100, 117.6) for which NO_X = 2.43, *Eff* = 0.78, and *UC* = 3.20. The method converged to the optimum point (200.0, 100.6, 117.6) in 10 iterations, the path to convergence is shown in Figure 5-19.

Discussion

The algorithm presented has been subject to a number of simulations, all of which converged to the correct solution. A convergence proof has not been found although it has been shown that the optimum is a fixed point of the iterative mapping. The central feature of the method is the use of a unit cost function expressed in high-level plant variables. This cost function may prove difficult to define in certain circumstances, but this is only a reflection of the difficulty in formalizing some aspects of plant performance. The resolution of conflicts of advice between different local optimal policies cannot be determined by mathematics alone, but some extra criteria must be supplied. Since the various local optimization processes are dealing with different plant variables, a way of combining disparate variables must be found. The most general of these, and the most flexible, is to associate a cost with each of the local variables. Difficulty in doing this for any plant variable indicates that a deeper study of the impact of this variable on plant and environmental behavior. The use of such a unit cost function does not require that the iterative algorithm described has to be used to effect the optimization; a direct method could also be used. This has the disadvantage of using the local optimizers only as variable predictors and also that the dimension of the optimization problem is that of the number of plant control variables being used. The iterative procedure described in this report uses the optimization capability of the local optimizers and reduces the main optimization dimension to the number of local optimizers –1. This is usually a considerable reduction.

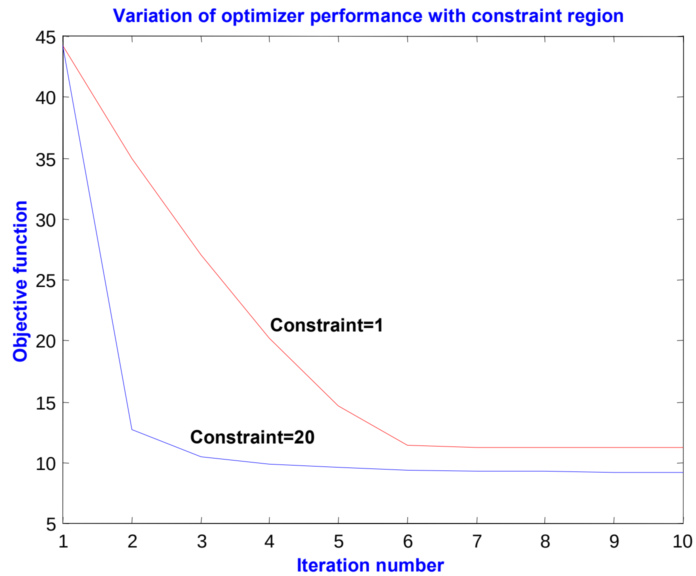


Figure 5-16 Variation of Optimizer Performance with Constraint Region

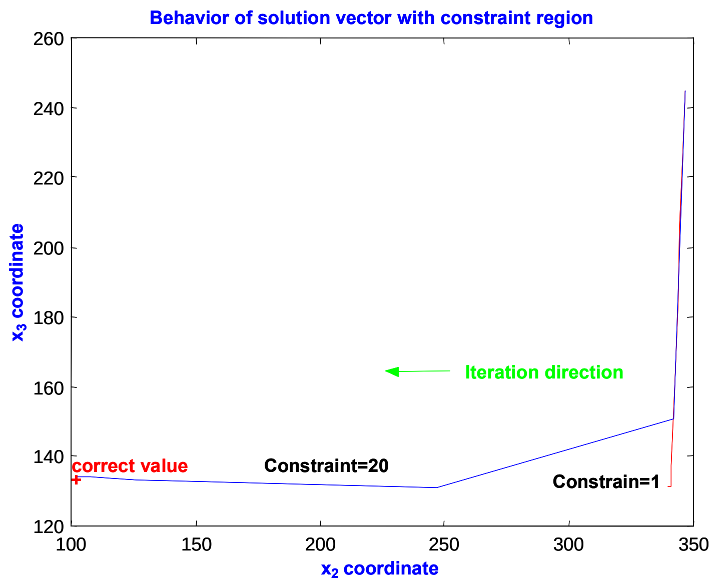


Figure 5-17 Behavior of Solution Vector with Constraint Region

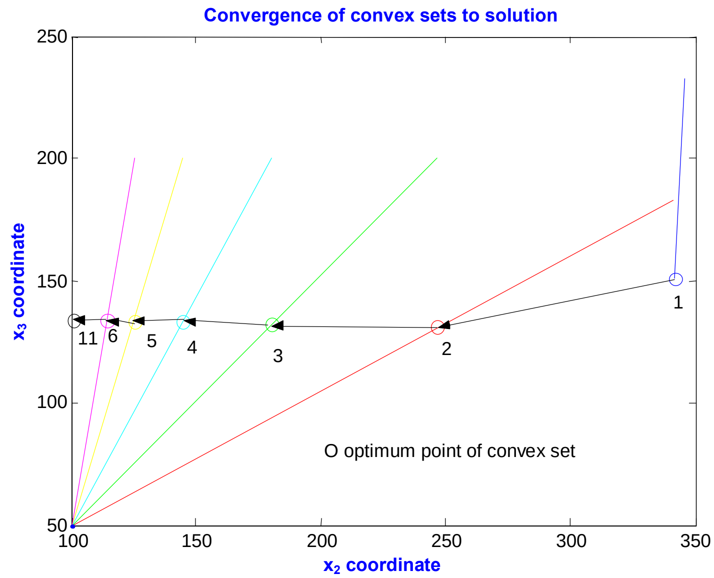


Figure 5-18 Convergence of Convex Sets to Solution

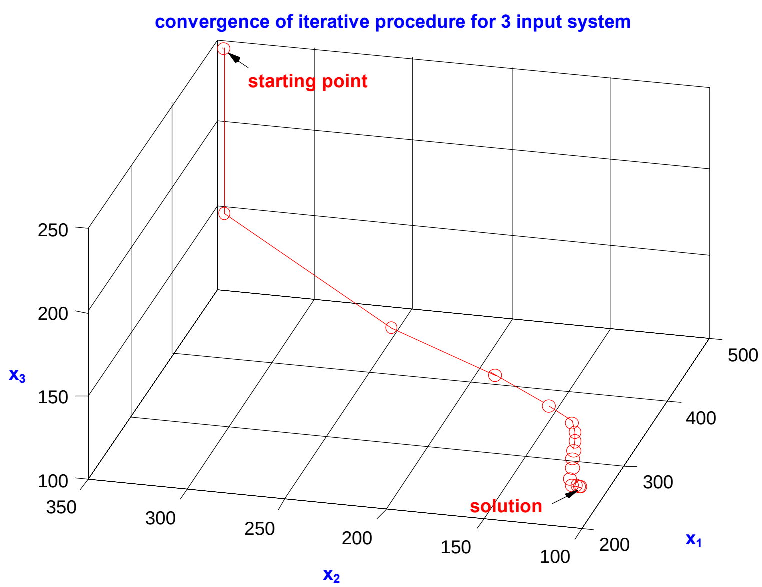


Figure 5-19 Convergence of Iterative Procedure for Three Input System

Software Implementation

As discussed earlier, Powergen developed a prototype, proof-of-concept software (in Matlab). SCS was responsible for the conversion to production code and implementation. Figure 5-20 shows the relationship of the Powergen optimizer software to the other software components. The major efforts consisted of:

- Migrating the Matlab code to C++.
- Adapting the migrated software to fit within the optimization framework.
- Adding enhancements to the originally developed code so that it could be configurable (through initialization files) and support sub-models in which the inputs and outputs are unknown until run-time.

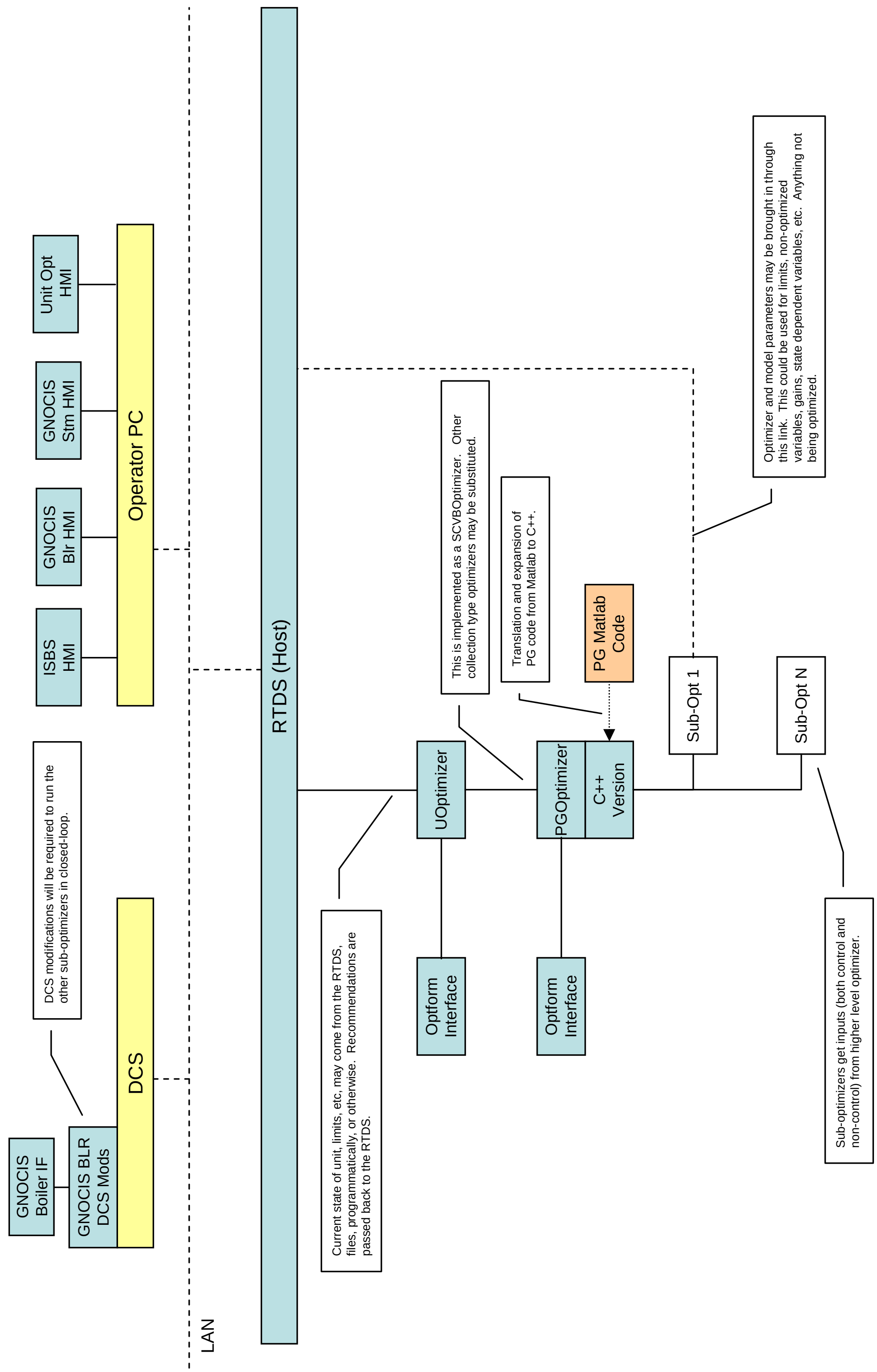


Figure 5-20 Relationship of Powergen Optimizer Software to Other Project Software

Synengco SentinentSystem Software

The Hammond 4 Global Optimization project was initiated with the aim of developing a system to allow global optimization of the Hammond 4 unit. To achieve this requires running multiple optimization models in a global optimization framework. One of the approaches taken was to apply the Synengco's SentinentSystem Global Optimizer. Its relationship to the other software in the project is shown in Figure 5-21. The SentinentSystem software and its application in the project are discussed in the following paragraphs.

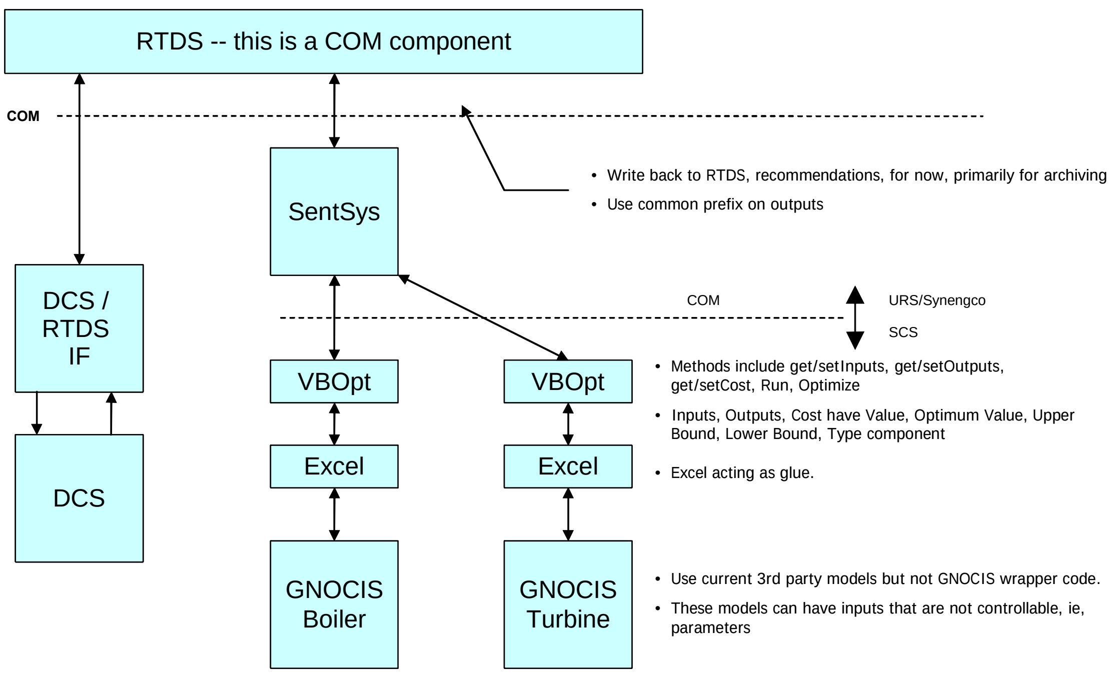


Figure 5-21 Relationship of Sentinent System Software to Other Project Software

Synengco developed wrapper code for the existing boiler and turbine optimizers so they could be incorporated into SentinentSystem and provide a consistent framework for all the optimization models to contribute to a single objective function. The framework was also used to control the various models and optimization models to achieve global optima. A single optimization algorithm could have provided a global optimum and although the SentinentSystem framework is capable of solving it this way it would have required extensive processing power due to the number of manipulated variables available to achieve global optimum.

A feature of SentinentSystem is the ability to cascade optimization where sub-optimization can be used within a global optimization. In some optimization problems there is a requirement to cascade optimization (i.e. optimize subunits and then the whole unit). The concept is to optimize at the subcomponent level and then use this information to obtain the starting point for the global optimization. This provides the benefits of processing efficient generation of solutions as each subunit is solved individually, a smaller problem, while still being part of the global optimization as well as allowing the optimization technique for each subunit to be selected independently to best match the requirements of the subunit. Using the cost function as the objective function for all models allows for the summation of the sub-models in the global optimization and therefore provides the essential link for global optimization. The linkage between the models is established within the unique SentinentSystem framework and the manipulated variables brought out for the global optimizer.

This technique allows faster conversion to the optimum solution, as the number of manipulated variable is small for each sub-optimization.

SentinentSystem has a number of standard optimization algorithms such as:

- Nelder-Mead Simplex algorithm
- Broyden-Fletcher-Goldfarb-Shanno algorithm
- Newton-Conjugate-Gradient
- Full Hessian
- Hessian product
- Least-square fitting
- Bounded minimization
- Root finding

This range of solutions ensures that the best optimizer in the global framework is selected and tuned to meet the individual requirements of each component within the global optimizer.

SentinentSystem also has a number of hybrid optimization algorithms such as:

- Genetic algorithm which uses a number of different techniques to simulate what happens in natural selection process (survival of the fittest) to find the optimum solution. This includes mutations, which try selectivity within the entire solution space to ensure a global optimum is being found rather than a local optimum. If a mutated solution is close to the current optimum it will search the area for a better solution than the current optimum. Mutations that do not improve the solution cease to exist while those that improve the solution spawn other solutions.

- A hybrid optimizer was developed for Hammond 4. This optimizer incorporates some of the above benefits as well as addresses some of the problems with the previous optimizer in terms of speed to converge for Hammond 4 (see following section).

The existing models from Hammond were adapted to fit into the SentinentSystem Model Framework so the model results can be incorporated into the global optimization objective function as well as be controlled by the SentinentSystem to achieve global optimization.

Various optimization techniques can be used with the adapted models and switched in and out from a simple pick box. A problem that became apparent was that it was taking too long for the optimizer to converge to a global optimum. This led to the development of the Hammond 4 hybrid optimization model. This approach reduced the number of calls to the model and allowed the rate of change from current operation to optimized operation to be done in a controlled and definable manner.

Hammond Hybrid Optimizer

A hybrid optimizer was developed for Hammond 4 to ensure that the global optimum converged in an acceptable time frame. The standard optimizer algorithms considered made numerous calls to the model and, due to the length of time and the frequency of the calls, resulted in an unacceptable delay in finding the optimum solution. In addition there is a requirement to rate limit the changing of parameters and as a result it takes some time for the plant to get to the optimum (i.e. don't want to move the plant around too fast). The outcome of this is that by the time the plant has been moved, the optimum solution has changed due to changes to the ambient and/or operating conditions. To address these problems, a hybrid optimizer was developed with the following design objectives:

- Minimize the number of calls to the models
- Rate-limit the movement of controllable parameters
- Provide closed loop control capacity
- Capacity to provide open loop control capacity by recommending the movement of only one process parameter at a time
- Provide feedback from the actual plant so that it is possible to move outside the trained model bounds with lower risk
- Only move controllable parameters if the impact is above a specified threshold

The *h4optimizer* is a solver developed specifically for the project at Hammond. Each manipulated variable has a lower and upper limit and a minimum and maximum allowable change. For each manipulated variable, the optimum value in the range satisfying the lower and upper limits and the maximum allowable value is found. If the optimum from the first step does not satisfy the minimum allowable change constraint, the optimizer then searches in the two regions inside the allowable limits (defined by the lower and upper bounds and the maximum allowable change), but excluding the range defined by the minimum change constraint. These search results are combined into a single result, which is the optimum value within the legal domain for that manipulated variable. This procedure is performed for each manipulated variable. For one complete optimization, the optimum considering the complete set of manipulated variables is that change of one manipulated variable which minimizes the objective function. Using this procedure, there will be only one recommended manipulated variable change per time step.

6
INTELLIGENT SOOTBLOWING SYSTEM

Overview

It is generally recognized that boiler sootblowing has an effect on NOx emissions, boiler efficiency, and steam temperatures. Sootblowing also affects boiler tube life. Boiler tube failures are the leading cause of unplanned outages on of coal-fired units. Most utilities sootblow: (1) at regular intervals based on the plant's staff past experience or (2) based on operator interpreted feedback from the furnace (such as difficulties in maintaining steam temperature or pressure). It was the goal of this task to identify and install suitable commercial or near commercial suppliers of optimization packages (suitable for this problem domain) and interface this system to the unit optimization package. Commercial packages available at the start of the project included those from Applied Synergistics Incorporated (ASI) and Diamond Power. Near commercial packages or those with substantial prior development included those from DHR, Powergen, and Westinghouse [EPR98b]. Since then, other packages are also being offered commercially including those from Clyde Bergemann, URS/Synengco, Invensys, and B&W. These packages can be broadly categorized as either: (1) instrument based requiring additional instrumentation such as furnace tube heat flux sensors, flue gas temperature measurements, etc, or (2) model based using computing technologies such as neural networks, fuzzy logic, and expert systems.

The Powergen ISBS software was selected for this demonstration due primarily to its low cost as compared to the other technologies and it required no additional instrumentation. Powergen had previously tested this system at Powergen's Kingsnorth Station Unit 1, a tangentially-fired unit [EPR99].

Sootblowing Hardware Description

The existing sootblowing system was supplied by FWEC and manufactured by Copes-Vulcan. The type, quantity, and location of the sootblowers are shown in Table 6-1, Figure 6-1, and Figure 6-2. The sootblowers use steam extracted from the boiler at the partial division wall superheater as the sootblowing medium. This steam has a nominal pressure and temperature of 2550 psig and 870ºF, respectively. A pressure reducing value regulates the steam to a pressure of approximately 500 psig upstream of the sootblowers. According to the type and location, the pressure is further reduced to a pressure of between 125 psig to 200 psig for sootblowing. The flow through the system when blowing is on the order of 50,000 lb/hr. The sootblowing control logic control program is executed in an Allen-Bradley Programmable Logic Controller (PLC). This PLC is interfaced to the DCS through a gateway in order to provide the operator and DCS full access to the sootblowing controls from the DCS. The operator interacts with the sootblower equipment primarily through two graphics on the DCS console (Figure 6-3 and Figure 6-4). From these displays, the operator may initiate sootblowing and monitor its operation.

Table 6-1 Sootblower Types and Locations

Type	Quantity	Location
Long retractable	34	Furnace nose (2)
		Pendant superheaters (12)
		Furnace rear wall screen (4)
		Convection superheater (4)
		Reheater (10)
		Secondary economizer (4)
Half travel	4	Primary economizer (4)
Wall deslaggers	49	Side walls (30)
		Front wall (10)
		Rear wall (9)
Water injectors	15	Side walls (12)
		Rear walls (4)

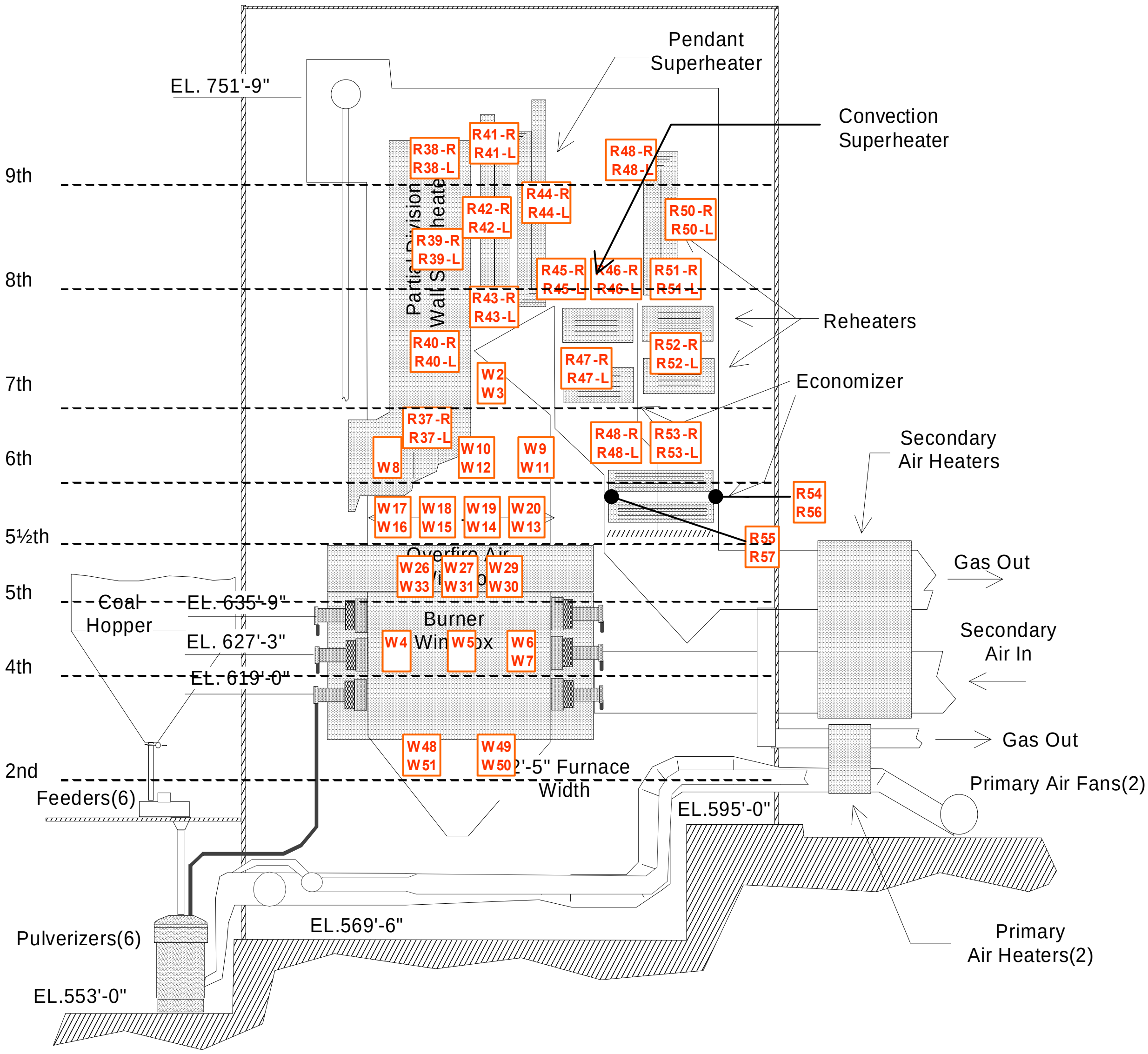


Figure 6-1 Location of Sootblowers in Furnace

Group		Sootblower
1	38L	37L, 37R, 38L, 38R, 39L, 39R, 40L, 40R
2	41R	41L, 41R, 42L, 42R, 43L, 43R, 44L, 44R
3	45R	45L, 45R, 46L, 46R, 47L, 47R, 48L, 48R
4	49R	49L, 49R, 50L, 50R, 51L, 51R, 52L, 52R, 53L, 53
5	W54	W54, W55, W56, W57
6	W13	W2-W15
7	W17	W16-W51

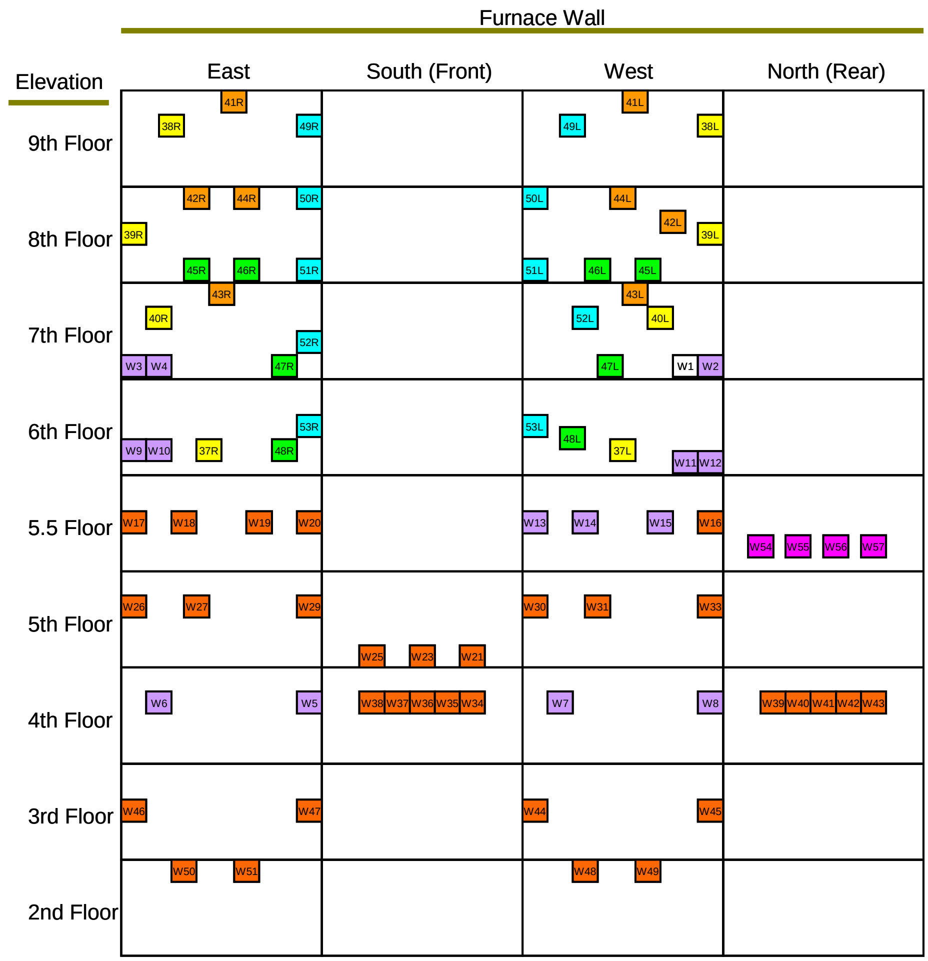


Figure 6-2 Location of Sootblowers in Furnace

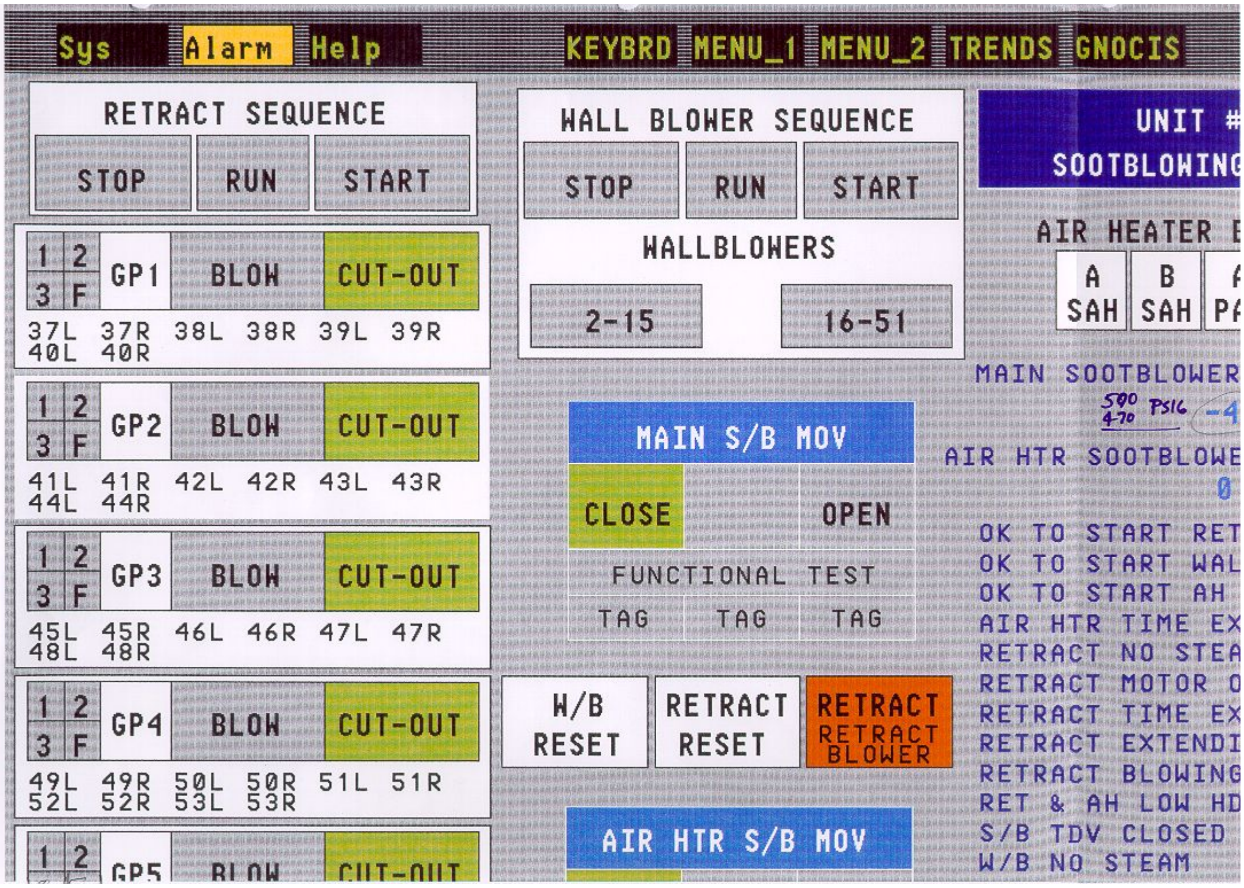


Figure 6-3 Sootblower Control Interface on the DCS (Panel 1)

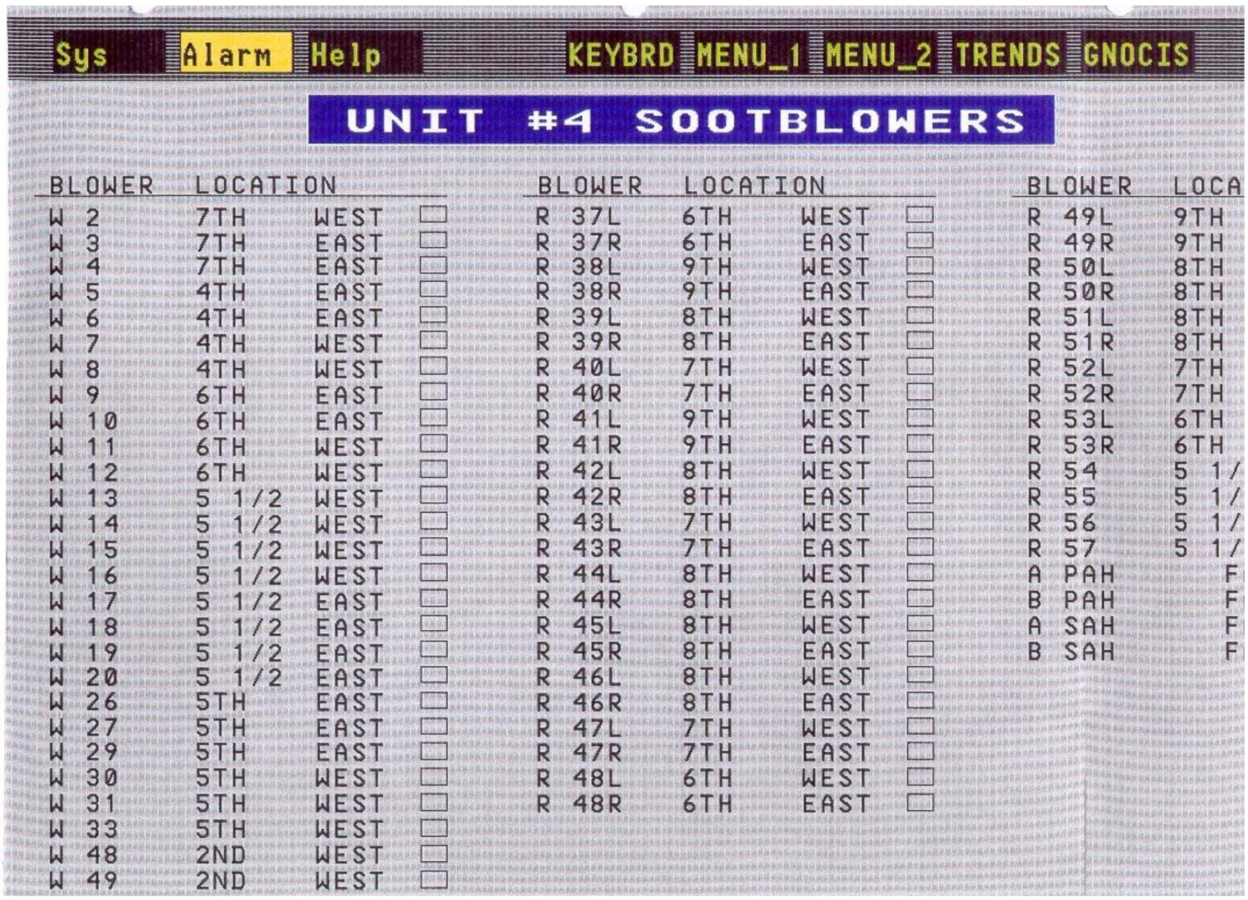


Figure 6-4 Sootblower Control Interface on the DCS (Panel 2)

Review of Sootblower Operation and Operational Impacts

During second quarter 2000, a review of sootblower operations and impacts was performed by Powergen, consulting with plant operating personnel. General guidance for sootblowing is provided in the applicable operating procedures, an excerpt of which is shown in Figure 6-5. Under normal operation, sootblowing is performed once per eight-hour shift, at the unit operator's discretion. This sootblowing is performed automatically, in groups, as shown in Figure 6-2. An operator usually walks round the boiler at least once a shift and the boiler is visually inspected to determine cleanliness. The main hatch through which the tubes can be seen is on the eighth level (Figure 6-1) and allows the operator to see the pendant superheater and any ash that may have deposited near the neck of the boiler. If that area is fouled, Groups 1, 2, and 3 are cleaned. Most operators would also clean Group 4 (everything except the waterwall sections). Visual inspection is not possible in the reheat area. Some operators sootblow the reheater when the load is reduced to help achieve the 1000°F reheat temperature design target. Furnace walls are never sootblown (Groups 5, 6, and 7). This is due to there being very little slagging on the waterwalls since the installation of the low NOx burners and difficulties in reaching design steam temperatures (1000°F) over the load range, particularly at lower loads (Figure 6-6). As shown, reheat temperature droop is more of an issue than superheat temperature. These deviations from design have significant detrimental impacts on both unit heat rate and load capacity.[1] Practices diverge among operators but if there is something common, it is that operators would rather sootblow regularly (too often) than not. Tube erosion is not perceived as a problem for the operators mostly because they are not aware that tubes are regularly replaced during outages.

An example of typical sootblowing operation over several days is shown in Figure 6-7. During this five-day period, sootblowing operation was generally initiated three times a day. As is normal practice with the operators, the four groups were cleaned in sequence automatically (Figure 6-8).

In that context, it is not possible to formalize the operators' judgments, as many of the clues for sootblowing are visual and are not easily automated. Moreover, superheater spray does not seem to be a constraint (they claim they have plenty of it and consider, correctly, that it does not significantly affect the unit efficiency). Whereas Kingsnorth reheat temperature constantly had to be watched so that it wouldn't exceed the design level, that problem does not exist at

[1] On a typical drum unit, a 10°F drop in main steam temperature results in a 0.2% increase in heat rate and 0.15% reduction in load capacity. Similarly, a 10°F drop in reheat steam temperature results in a 0.14% increase in heat rate and 0.14% reduction in load capacity.

Hammond and indeed they hardly ever exceed the 1000°F target reheat temperature. In that context, a simple extrapolation of Kingsnorth rules was ruled out as a possible approach at Hammond.

The operators recognize do not have sufficient information to assess the cleanliness of the reheater. Consequently some correlation must be found between the sootblowing activity and some boiler variables and this is discussed in the following paragraphs. The configuration of the gas and steam flow paths and control logic are provided in Figure 6-9 through Figure 6-12 to aid in the interpretation.

2125.210 BLOWING FREQUENCY

NOTE: Do Not blow soot out of habit.

NOTE: Below 30% load (150 MW), Do Not blow Retract or Wallblower sootblowers. Air heater blowers may be blown.

Normally all Wall Blowers and Retracts are blown one time per shift if conditions permit.

The following will indicate a need for blowing certain areas:

1. Visual Check
2. Low Superheat Temperature: This indicates this section is slagged up, or the Waterwall section is too clean.
3. Low Reheat Temperatures: This indicates this section is slagged up, or the Waterwall section is too clean.
4. High Reheat Temperatures: This could indicate that the Superheat section is slagged up and not absorbing the gas temperature before it reaches the Reheat section.

Figure 6-5 Excerpt from Hammond 4 Operating Procedures for Sootblowing

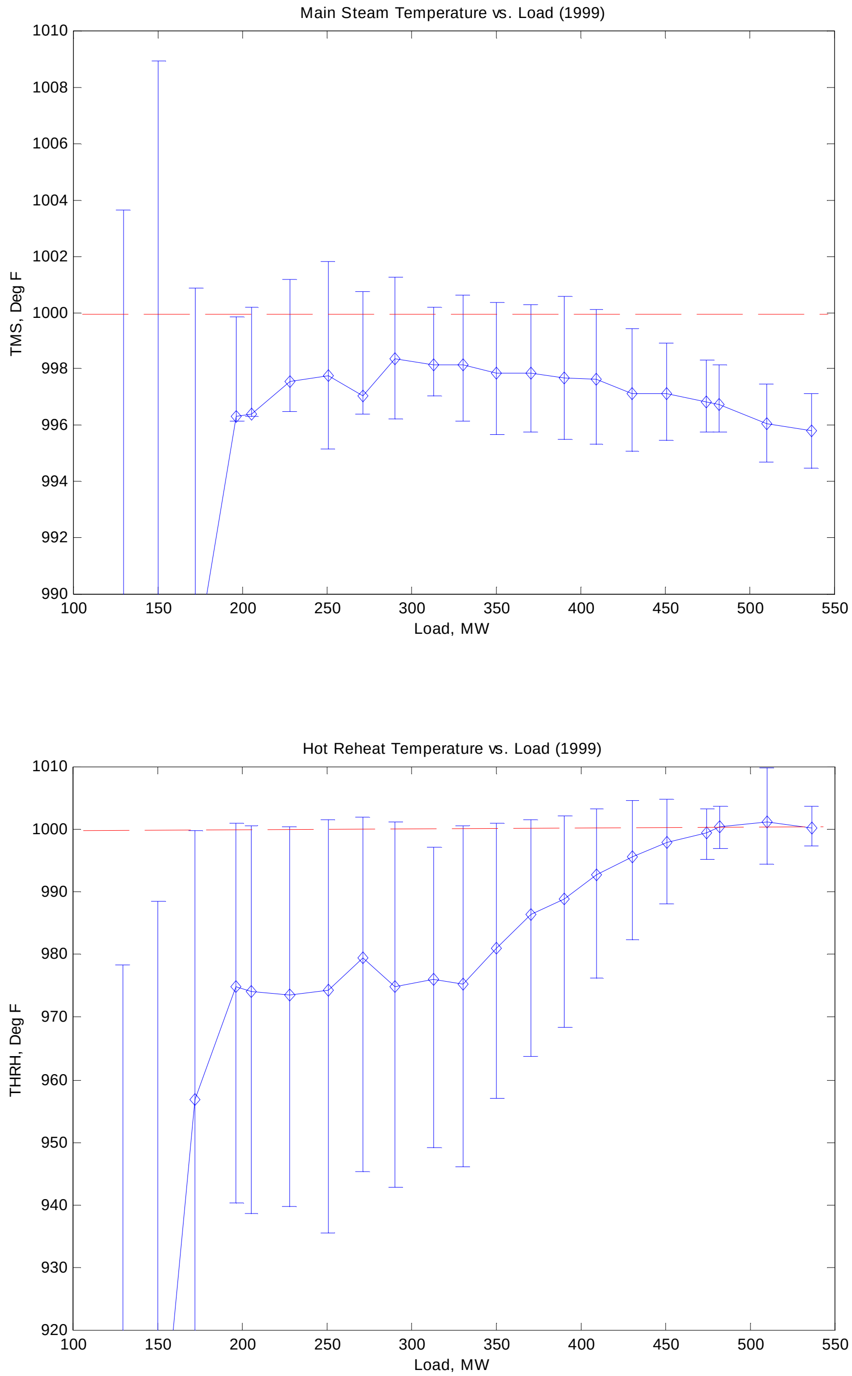


Figure 6-6 Main Steam and Hot Reheat Temperatures vs. Load for 1999

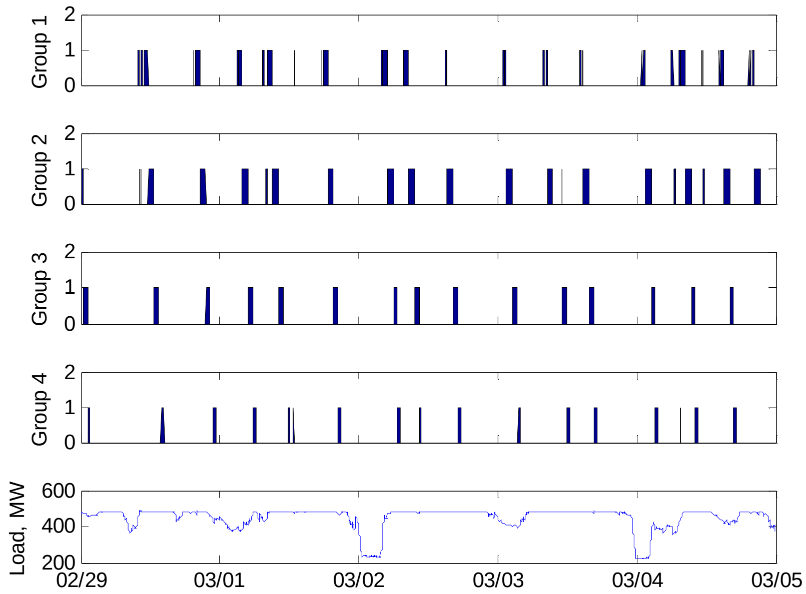


Figure 6-7 Sootblowing Activity by Group for February 29 through March 4, 2000

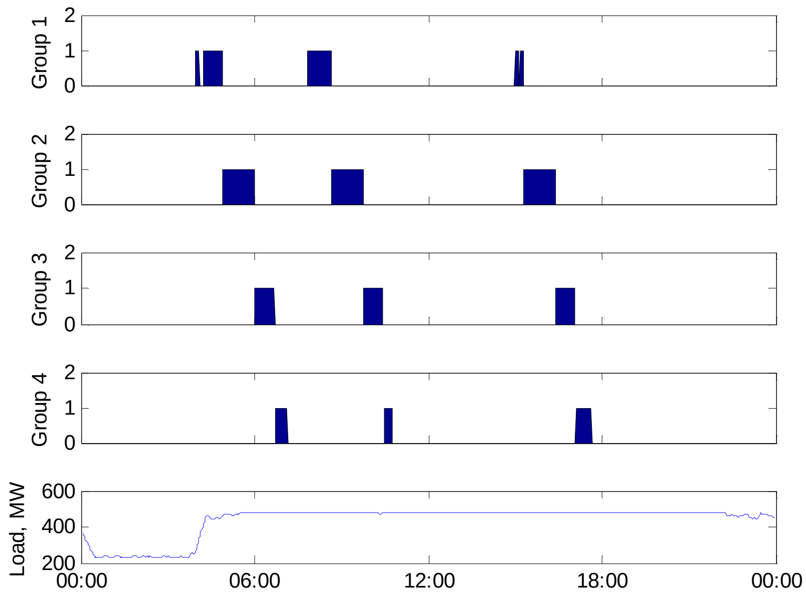
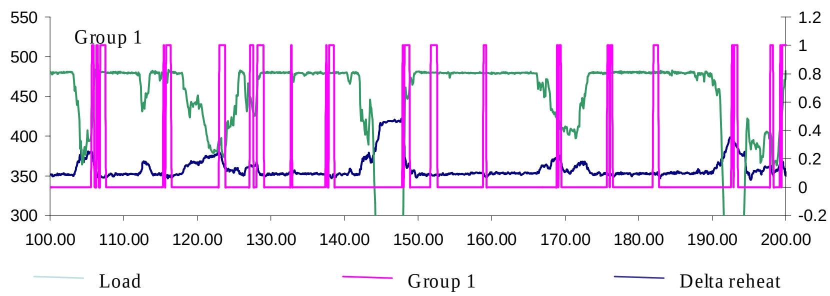


Figure 6-8 Sootblowing Activity by Group for March 2, 2000

Sootblowing vs. Boiler Cleanliness

Sootblowing Activity Compared to Reheat Heat Flux

It was expected that sootblowing Group 1 and Group 2 would result in more heat being transferred upstream of the reheat section leading to a decrease of heat being absorbed in the reheat section. This is not reflected in the change in fluid temperature flowing through the tube banks as can be seen from the plot below for the Group 1 sootblowers.



Where:

- The load in MWe (green line, left axis)
- The sootblowing activity of a specific group (pink line, right axis). A spike means the sootblowers are in use.
- The difference between the reheat outlet temperature and the reheat inlet temperature (in ^{o}F, blue line, left axis). This difference can be interpreted as an indication of the heat transfer to the reheat tubes.

The horizontal axis represents approximately 100 hours starting February 29, 2000 at 4 am through March 4, 2000 at 8 am (approximately 4 days).

The use of the sootblowers in Group 4 should result in more heat being absorbed in the reheater. Once more, no correlation of sootblowing with fluid temperature could be found. As discussed previously, existing practice is for each of the groups (Groups 1 through 4) to normally be blown sequentially starting with Group 1 and thus the same plot for Group 4 is very similar to the above. The changes in fluid temperature are related to load and not sootblowing.

There are two plausible explanations to this lack of correlation:

- The boiler is clean, so sootblowing does not affect the cleanliness of the boiler, hence, no effect observed in the reheat section. Although this could be true occasionally, this is unlikely to be the case forever.

- The reheat temperature is controlled by the reheat dampers that adjust the flux of the combustion flow automatically to keep the reheat temperature as constant as possible, masking all heat flux alterations due to fouling. This was thought to be the most likely explanation, and this idea is developed in the next section.

Sootblowing Activity Compared to Flow Split (Reheater/Superheater)

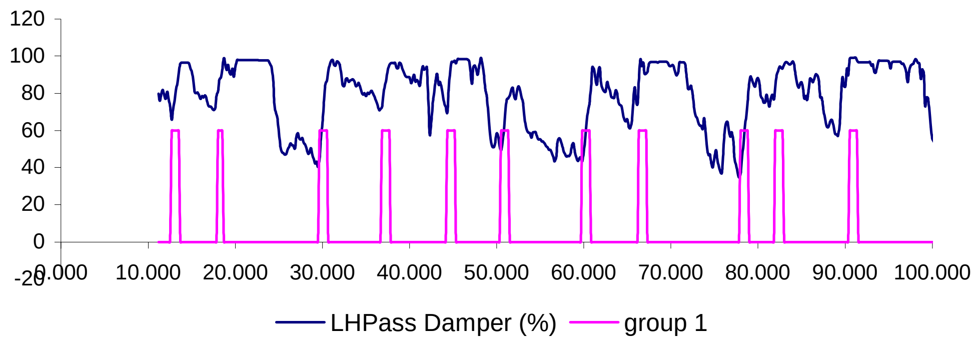
The following paragraphs explain how the superheat/reheat pass dampers operate.

Main steam enters the turbine at 1000°F and passes through the high-pressure section and then returns to the boiler to the reheater sections. There are two reheater sections located in the back pass of the boiler; the left-hand reheater section and the right-hand reheat section. The steam travels through these sections, and the flue gas passes over these sections, increasing the steam temperature in the reheater sections.

The back pass of the boiler is also divided into two sections, the reheater section and the convection superheater section. The outlet of these two sections have control dampers referred to as the superheater pass damper and the reheater pass damper. These dampers are mechanically linked and controlled by a single drive unit (Figure 6-11). These are referred to collectively as pass dampers. There are two sets of pass dampers; the left hand pass dampers (LH superheater pass damper and LH reheat pass damper) and the right hand pass dampers (RH superheater pass damper and RH reheat pass damper). These dampers move in opposite directions. When the reheat pass damper is 100% open, the superheater pass damper will be effectively closed. At a 50% demand, both dampers will be 50% open. By operating the dampers in this manner, the flue gas flow through the reheater pass can be regulated. The gas leaves the boiler by passing through the reheat pass and superheat pass sections. However, the amount of gas that passes through each pass is determined by the position of the pass dampers. By increasing the reheat pass damper position, the superheat pass damper will be decreased; thus, more gas will pass through the reheater pass section and less gas will pass through the superheat pass section.

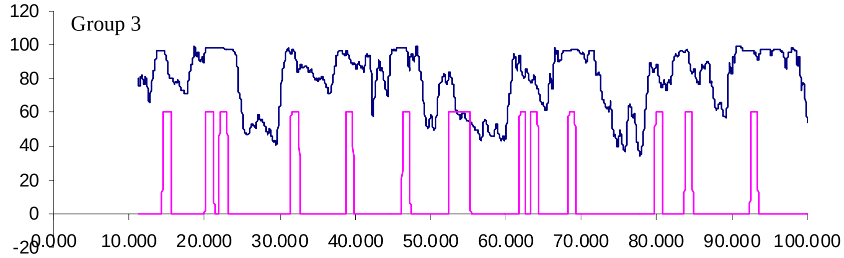
Reheat steam temperature is primarily controlled by the positioning of both the left hand and right hand pass damper positions. The DCS control logic is shown in Figure 6-12. The DCS compares the actual reheat steam temperature to the reheat temperature setpoint and adjusts the demand to both the left pass damper drive units. If the reheat steam temperature is lower than setpoint, the DCS will increase the pass damper demand, which will increase the reheat pass damper position, increasing gas flow through the reheat pass and increasing the hot reheat steam temperature. If the reheat steam temperature is higher than setpoint, the DCS will decrease the pass damper demand, which will decrease the reheat pass damper position, decreasing gas flow through the reheat pass and decreasing the hot reheat steam temperature.

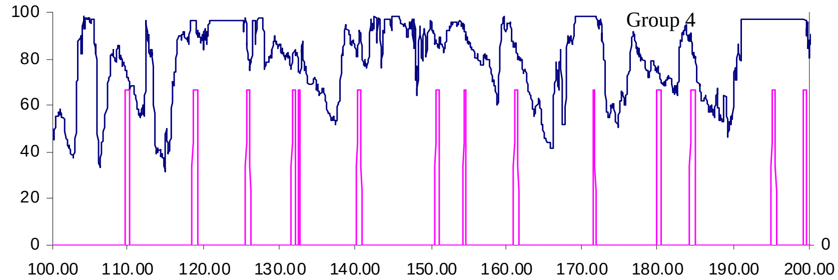
The following graph shows in blue, the LH pass damper position and in pink, the sootblowing activity (Group1):



It can be seen from the above graph that using the Group 1 sootblowers coincides with an opening of the LH pass damper. Since using the Group 1 sootblowers will clean part of the superheater tubes and remove more heat from the gas upstream of the reheater, the damper is opening to compensate and maintain reheater temperature. Since the sootblowers are used in sequence, starting with Group 1, the damper continues to open while Group 2 sootblowers are used.

As may be seen in the following two figures, the movement of the damper is less well correlated with the use of sootblowers in Group 3 and Group 4. When Group 4 is used then more heat will be taken from the reheater and the damper can be expected to close. This seems to be the case though starting to close the damper does not coincide with the start of using the Group 4 sootblowers.





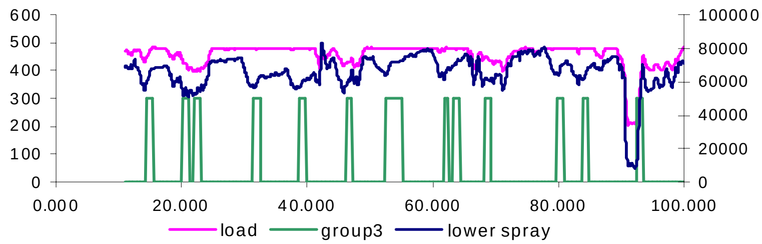
Sootblowing Activity vs. Use of Spray

It was also thought that the state of cleanliness of the boiler could be deduced from monitoring the use of spray. Presently, Hammond Unit 4 does not use reheat spray. The only sprays that are used are the lower and upper superheater sprays. Figure 6-10 should help to clarify their position in the steam cycle.

On the following two graphs:

- The pink line is the unit load in MWe (left axis)
- The amount of spray in blue is measured in lb/hr (right axis)
- Sootblower activity is in green
- Horizontal axis is the time in hours starting on May 25, 2000

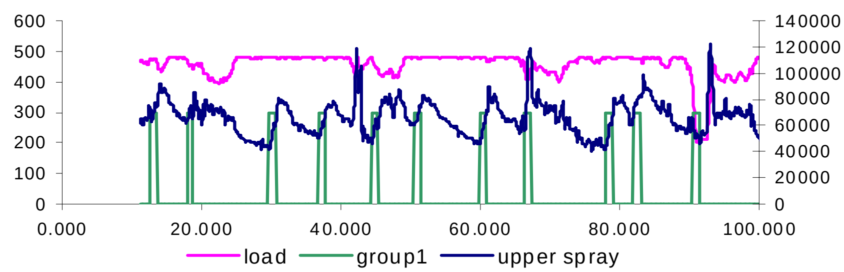
The first of these two graphs is an attempt to correlate the use of sootblowing Group 3 (convection superheater) with the amount of lower sprays. It is expected that cleaning the convection superheater would result in more heat being absorbed in that area, and as a result, an increase of the lower sprays. To a limited extent, this is true, but it is also apparent that these increases are much smaller than those observed following load increases.



A similar analysis with the upper sprays vs. sootblowing the division wall (Group 1) leads to interesting conclusions. Here, not only it is clear that sootblowing the division wall leads to more upper spray used, but also, the fluctuations due to load change are less important (once a few spikes are filtered out). Moreover, a rule such as:

if upper spray flow < 50000 lb/hr then sootblow Group 1

would have simulated the operators' use of the sootblowers most of the time, since currently the other groups follow the use of Group 1 in sequence. There are other rules that may also have modeled the operator sootblowing actions. It should be noted that this rule is likely a consequence of the operator action rather than an operator cue since upper spray flow rate is not explicitly mentioned in the Hammond 4 operating procedures for sootblowing.



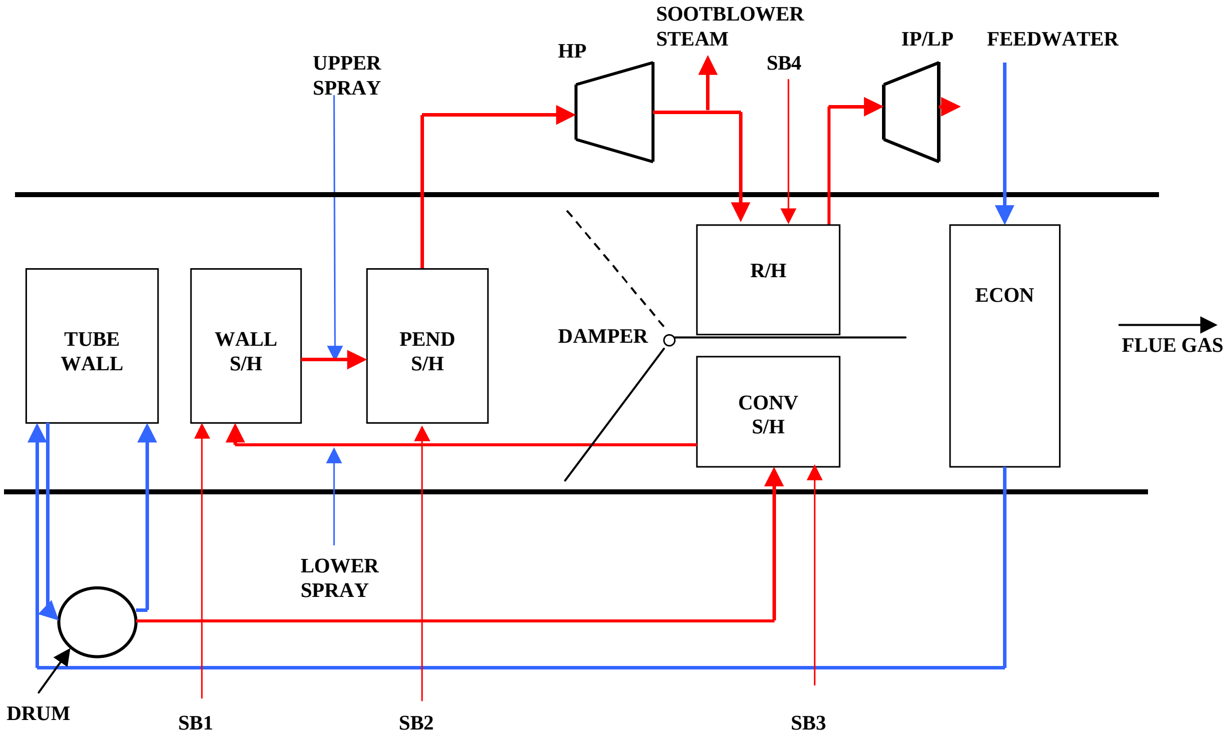


Figure 6-9 Steam and Gas Flow Schematic

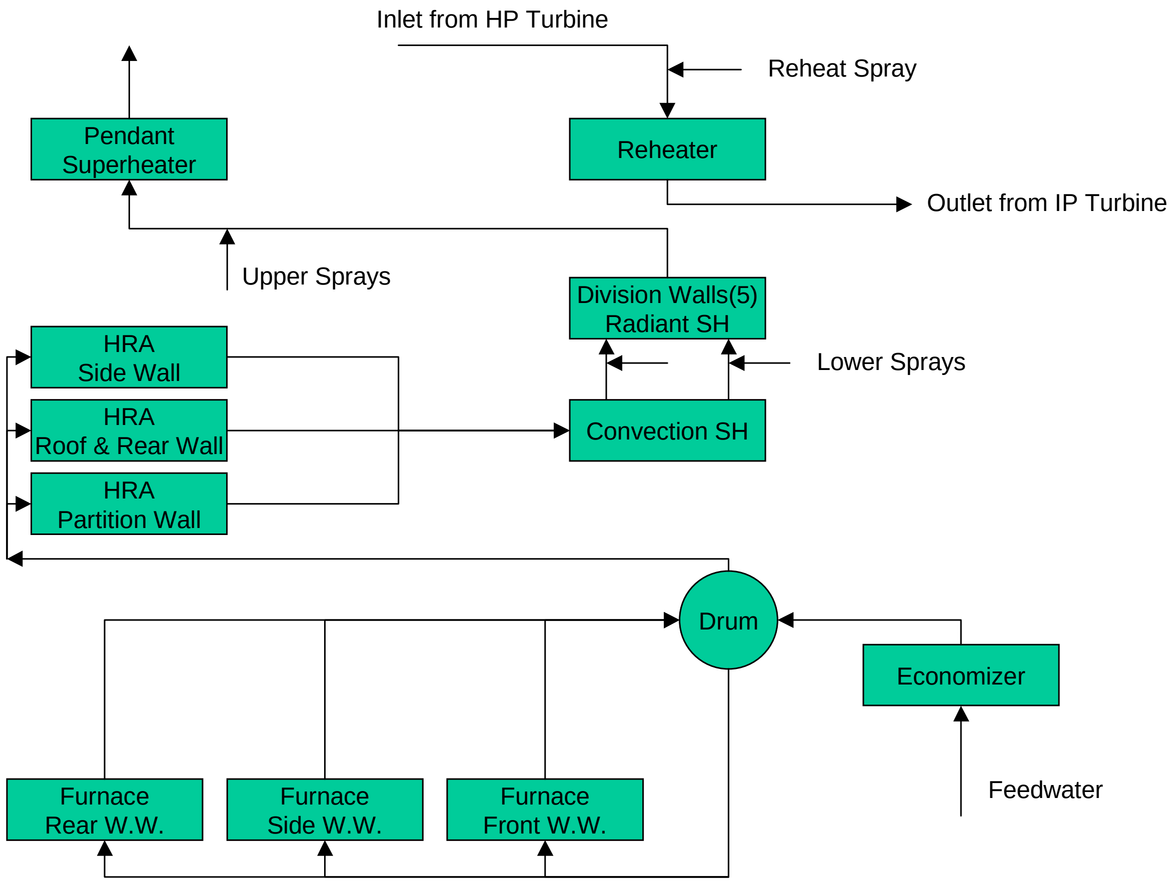


Figure 6-10 Steam and Spray Flow Paths

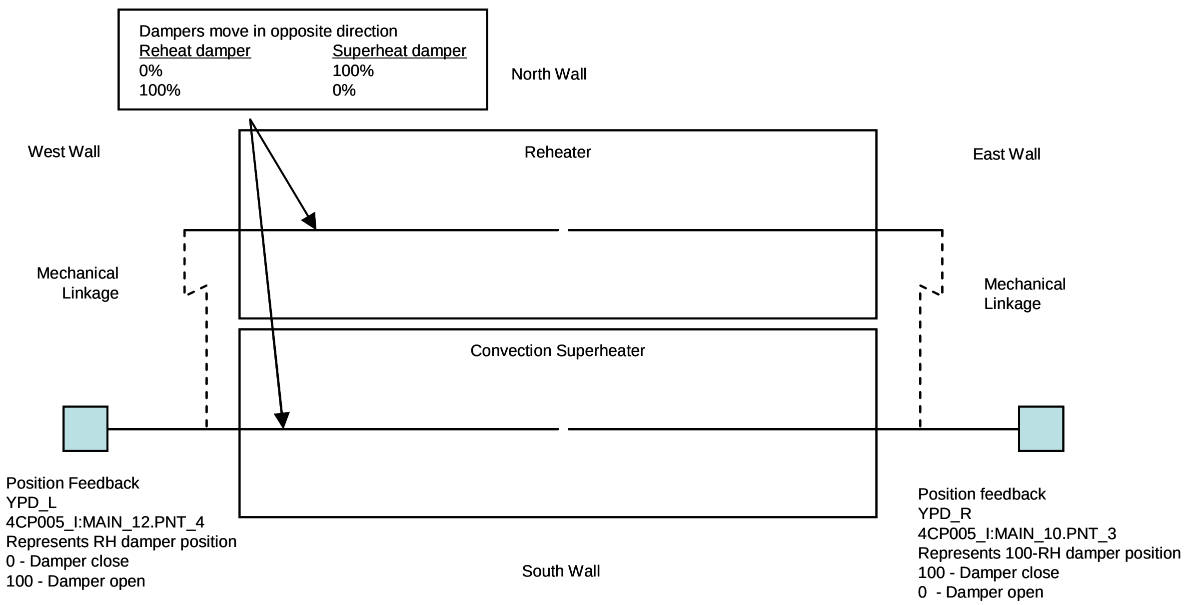


Figure 6-11 Operation of Pass Dampers

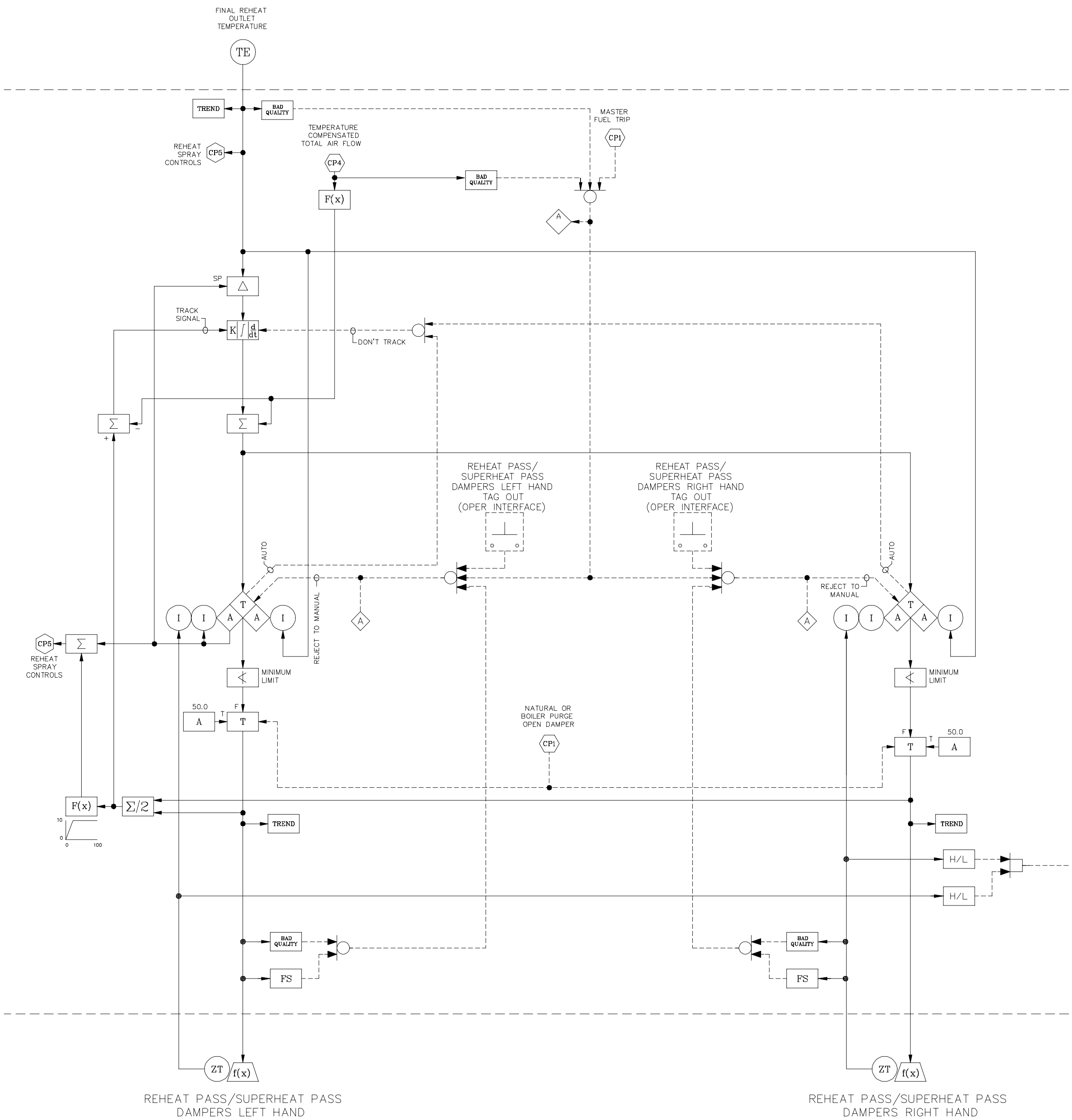


Figure 6-12 Superheat and Reheat Pass Damper Control Logic

Cleanliness factors

Fouling of the boiler has the effect of reducing the heat flux through the fouled surface. Therefore, if one is able to measure the heat flux through a surface and compare that flux against a known reference, then it may be possible to estimate the degree of fouling on the surface. However, heat flux sensors are not part of plant instrumentation and so this heat transfer must be inferred from other parameters.

The standard cleanliness factor calculation as previously applied to Kingsnorth is given below. Heat transfer can be computed section-by-section to produce a cleanliness factor. A classic definition of a cleanliness factor (CF) is:

$$CF = \frac{Q}{Q_{clean}}$$

where Q is the current heat transfer from the combustion gas to the water/steam of a particular section and Q_{clean} is the baseline Q for a clean surface.

The difficulty of assessing the baseline heat transfer for a clean surface is such that the alternative definition:

$$CF = \frac{Q}{Q_{av}} - 1$$

where Q_{av} is the average of Q over a long period is preferred. It is also thought to be more effective at capturing changes due to sootblowing.

With the notations:

$\dot{m}$	=	water/steam flow rate
c	=	specific heat of water/steam
T_i	=	inlet temperature of water/steam
T_o	=	outlet temperature of water/steam

the CF becomes:

$$CF = \frac{\dot{m}\,c(T_o - T_i)}{\left(\dot{m}\,c(T_o - T_i)\right)_{average}} - 1$$

Assuming that the mass flow (reheater, superheater water/steam circuit) is primarily a function of the load, it is then possible to derive CF for each load:

$$CF(load) = \frac{T_o - T_i}{(T_o - T_i)^{average}_{for\_that\_load}} - 1$$

where the subscripts *i* and *o* denote inlet and outlet measures.

The key behind this approach is to find a value of Q_{av} such that deviations from the average can be accounted from fouling alone and is thus insensitive to other boiler parameters, such as load, sprays, and damper position.

At Kingsnorth, the only variables required were the inlet and outlet temperatures of the sections concerned (superheater and the reheater). At Hammond though, these are kept constant by the control system actions and thus the cleanliness factors must be calculated differently.

Superheater Cleanliness Factors

From plant data, it can be seen that the main quantities varying during sootblowing are the lower and upper sprays. It is thus straightforward to base the cleanliness factor for the superheaters on these spray levels. The level of the lower spray will be used as a cleanliness factor for the wall superheater and the upper spray for the pendent superheater.

The minimum spray level at full load for each spray is about 50,000 lb/hr and at low load is 20,000 lb/hr. The baseline for each of these spray levels to define the cleanliness factor was thus taken as a linear variation between 20,000 and 50,000 lbs/hr with load.

With the notation:

$\dot{m}$ = steam flow rate

dm_s = spray flow rate

L = latent heat of spray

c = specific heat of steam

T_i = inlet temperature of steam

T_o = outlet temperature of steam

The cleanliness factor may be defined as follows:

$$Q = mc(T_o - T_i) + dm_s(c(T_o - T_i) + L)$$

$$Q_{av} = mc(T_o - T_i) + dm_{sav}(c(T_o - T_i) + L)$$

$$Q_{sprayref} = dm_{ref}(c(T_o - T_i) + L)$$

$$CF = \frac{Q - Q_{average}}{Q_{sprayref}}$$

$$CF = \frac{dm - dm_{average}}{dm_{sref}}$$

The above defines the superheater cleanliness factor purely in terms of the spray flows and can be applied separately to the upper and lower sprays. With the denominator being a reference to the heat transfer resulting from the sprays alone, the cleanliness factor as defined varies from 1 when clean to –1 when dirty.

Reheater Cleanliness factor

Calculating a cleanliness factor for the reheater is more difficult since the tube temperatures are held constant by the control system and reheater sprays are not used. Clearly the position of the reheater damper must be related to the fouling since this affects the volume of hot gas passing over the reheater while inlet and outlet temperatures remain constant. Unfortunately, the earlier plots show the position of the damper by itself does not correlate well with sootblowing.

The schematic of the Hammond boiler is useful to provide a better understanding of the problem (Figure 6-9). The schematic clearly shows the superheaters upstream of the reheater. The damper position and thus the volume of gas passing over the reheater depend upon the cleanliness of the superheater. If the superheaters are clean, then more gas will be required to pass over the reheater than if the superheaters are dirty. Thus for Hammond, the effect of the cleanliness of the superheater must be included in the model used to determine the reheater cleanliness factor.

The cleanliness of the superheater is determined by the level of spray and thus the following graphs (Figure 6-13 through Figure 6-16) show scatter plots of the bypass damper position against total superheater spray flow (sum of lower and upper superheat spray flows) for given unit loads. At higher loads (more than 400MW) there is clearly a trend in the scatter data of lower damper opening with lower spray. This is to be expected indicating that as the superheaters foul then less gas is required over the reheater to maintain the heat transfer. The plot is very similar to the scatter plot of reheat temperature obtained for Kingsnorth and thus the relationship below of damper position to total superheater spray can be used to determine a reheater cleanliness factor at high load. A trend line has been shown on the plot for load at 480MW. Operating above the trend line for a given level of spray flow, the reheater is dirty since the damper is more open and below the trend line the reheater is clean since the damper is relatively closed (Figure 6-17). Mapping the plant operation on the plots below thus enables the cleanliness of the reheater to be inferred.

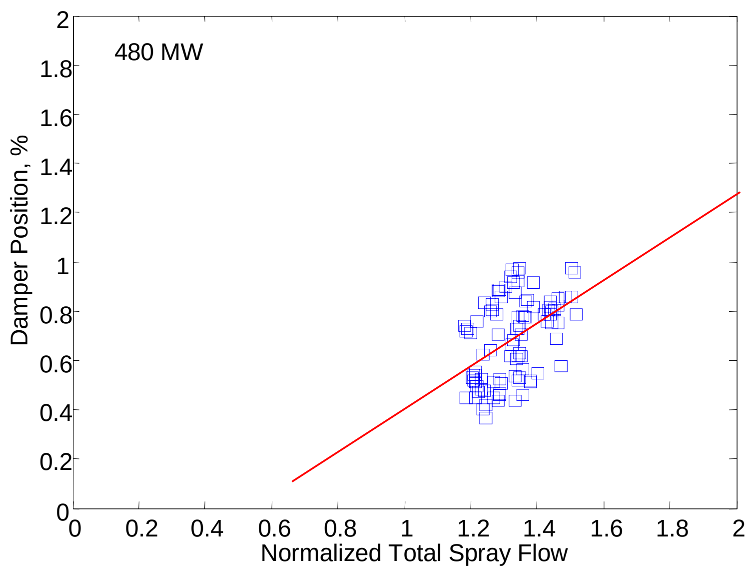


Figure 6-13 SH/RH Damper Position vs. Normalized S/H Spray Flow (480 MW)

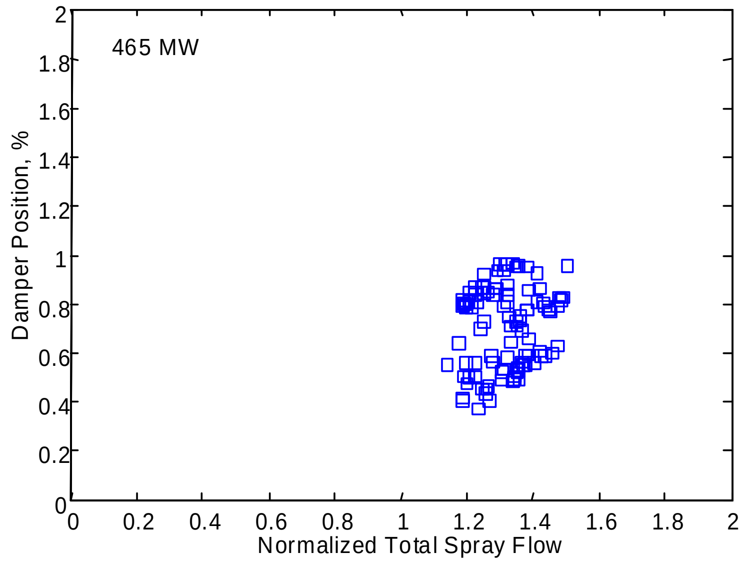


Figure 6-14 SH/RH Damper Position vs. Normalized S/H Spray Flow (465 MW)

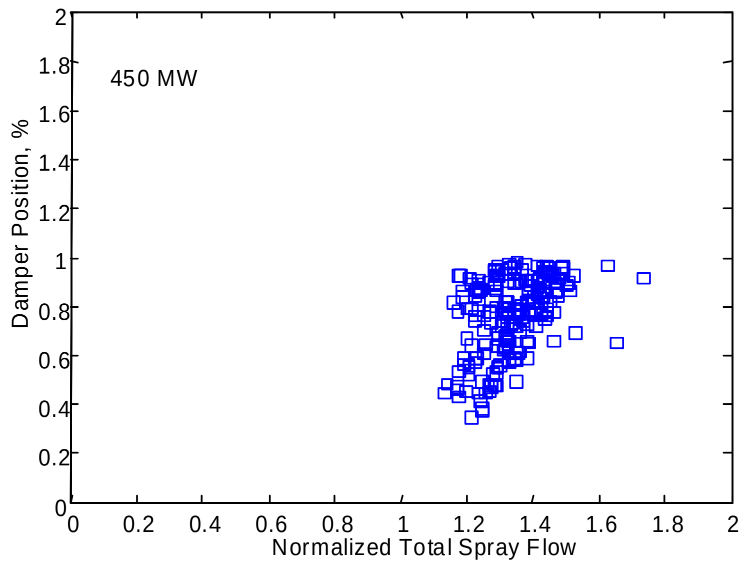


Figure 6-15 SH/RH Damper Position vs. Normalized S/H Spray Flow (450 MW)

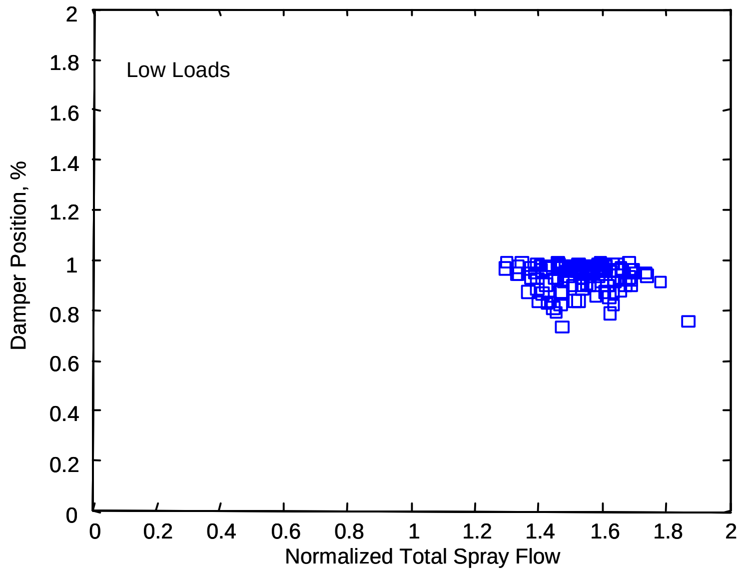


Figure 6-16 SH/RH Damper Position vs. Normalized S/H Spray Flow (Low Loads)

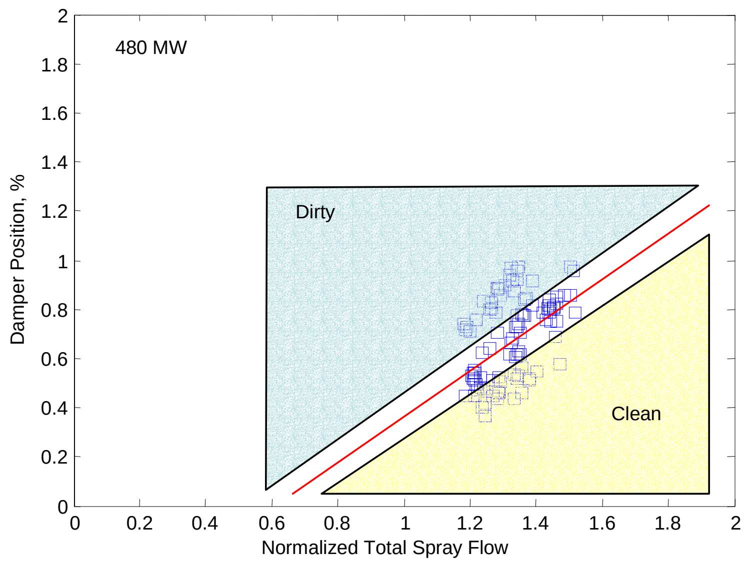


Figure 6-17 Evaluation of Reheater Cleanliness

Intelligent Sootblowing Model

The current work follows-up on prior work conducted by Powergen with funding from EPRI at Powergen's Kingsnorth Station [EPR99]. Based on the efforts and results at that site, it was decided to develop a fuzzy rule-base to generate recommendations. Using Matlab [Mat02c] and the Fuzzy Logic Toolbox [Mat02a] as a development platform, prototype rules were developed. These rules were subsequently translated into a Powergen developed library.

As at Kingsnorth, interviews were held with operators and other plant staff so that their method of working as it pertains to sootblowing could be understood and represented as a set of rules. This knowledge acquisition was considerably eased using fuzzy inference rules since the formulation closely follows the operators' conception of the process. The result of these interviews along with an evaluation of the process (discussed previously) was a small set of fuzzy rules (currently 8). These rules are described below.

The fuzzy system makes decisions based upon the following criteria:

- Reheat cleanliness factor
- Upper and lower spray flows
- Backpass damper position
- Reheat temperature
- Time since previous sootblowing

The elapsed time since last sootblow is corrected to account for periods when the unit does not operate and is reset to zero should the idle period exceed 15 hours. This period should be long enough for the furnace to cool and self-clean by contraction.

The operation of the backpass damper is entirely controlled by the control system. Although when putting the intelligent sootblowing model together there is a view on whether the damper should be opening or closing, there is no need for the routine to provide this advice since the damper operation is fully closed-loop.

It is assumed that the control system will call the ISBS routine about every five or ten minutes. There is thus a time since last sootblowing associated with each rule as otherwise the advice to sootblow may be given on several consecutive calls of the ISB routine since conditions in the boiler may not change sufficiently between calls.

It is not possible to distinguish between the use of the Group 1 and Group 2 sootblowers, thus in each case Group 2 is blown after Group 1.

The fuzzy rules are as follows:

Rule 1: If (Load is High and RHDamper_open is Narrow and PW_CF is Low and LastG12 is Long) then Blow Group 1 followed by Group 2

Rule 2: If (RH_CF is Low and LastG4 is Long) then Blow Group 4

Rule 3: If (Load is Low and RH_CF is Very_Low and Last G4 is Not_Recent) then Blow Group 4

Rule 4: If (Load is High and Conv_CF is Low and LastG3 is Long) then Blow Group 3

Rule 5: If(Load is Low and Conv_CF is Low and LastG3 is not Recent) then Blow Group 3

Rule 6: If (LastG12 is Very_Long) then Blow Group 1 followed by Group 2

Rule 7: If (LastG3 is Very_Long) then Blow Group 3

Rule 8: If (LastG4 is Very_Long) then Blow Group 4

where:

RH_CF - Reheat cleanliness factor

PW_CF - Pendent and wall superheater cleanliness factor based upon the upper spray flow

Conv_CF - Convection superheater cleanliness factor based upon the lower spray flow

RHDamper_open - Opening of the reheater damper

LastG12 - Last time Group 1 and 2 sootblowers were used

LastG3 - Last time Group 3 sootblowers were used

LastG4 - Last time Group 4 sootblowers were used

The first rule is the main decision rule for the use of the Group 1 and 2 sootblowers for cleaning the wall and pendent superheaters. With including the condition on the damper position, the use of the sootblowers will be delayed, possibly until the reheater has been cleaned. The model tries not to use the Group 1 and 2 sootblowers if the unit is struggling for reheat temperature. Since this is mainly at low load, this is facilitated by the high load constraint. At low loads, the use of Groups 1 and 2 is controlled by a time basis given by Rule 5.

The second rule is independent of load and is the main trigger for the use of the Group 4 sootblowers for cleaning the reheater.

At low load, the unit struggles to maintain superheat temperature and Rule 3 will initiate more cleaning of the reheater at low load if the unit is having difficulty maintaining reheat temperature.

Rules 4 and 5 trigger the cleaning of the convection superheater. Since the unit struggles for reheat temperature at low load, the reheat damper is generally wide open, limiting the flow over the convection superheater and the rate at which it fouls. Thus the convection superheater, like the other superheaters, is cleaned more sparingly at lower loads.

Rules 6, 7, and 8 ensure that the period between sootblower cycles cannot be too long. This is consistent with the operators' view that the tubes will be difficult to clean if sootblowing is too infrequent.

The fuzzy rules may be tuned by adjusting coefficients associated with the fuzzy membership functions. This library was written in C and compiled to a dynamic link library. The calling parameters are shown in Figure 6-18.

```
//INPUTS
        double Load             /* Load,  MW */,
        double RrhDamperpos             /* Reheat Damper Position, % */,
        double UpperSpray               /* Upper spray flow, lb/hour */,
        double LowerSpray               /* Lower spray flow, lb/hour */,
        double Treheat          /* Hot Reheat temperature, Deg F */,
        int             Group1          /* G1 sootblowers running */,
        int             Group2          /* G2 sootblowers running */,
        int             Group3          /* G3 sootblowers running */,
        int             Group4          /* G4 sootblowers running */,
//OUTPUTS
        int             *Baddata        /* 1= bad data 0= good data */,
        double *a1                      /* cleanliness indicator group1*/,
        double *a2                      /* cleanliness indicator group2*/,
        double *a3                      /* cleanliness indicator group3*/,
        double *a4                      /* cleanliness indicator group4*/
```

Figure 6-18 Intellisoot Library Call Parameters

Software Description

SCS was responsible for the developing the ISBS software interfacing with the Powergen developed Intellisoot.dll and the balance of the software project. This software in relation to the other software developed during the project is shown in Figure 6-19.

The system architecture is a client/server application (Figure 6-20). The client has many versions including a standalone engine control-based client, a standalone operator client, and separate ActiveX operator client controls. The server is written as a DCOM-compliant server. The ISBS server is the calculation engine for the ISBS, whereas the client modules offer limited control over the engine and provide feedback on the engine status. The primary development language for the ISBS package is Visual Basic. An overview of the software is provided in this section.

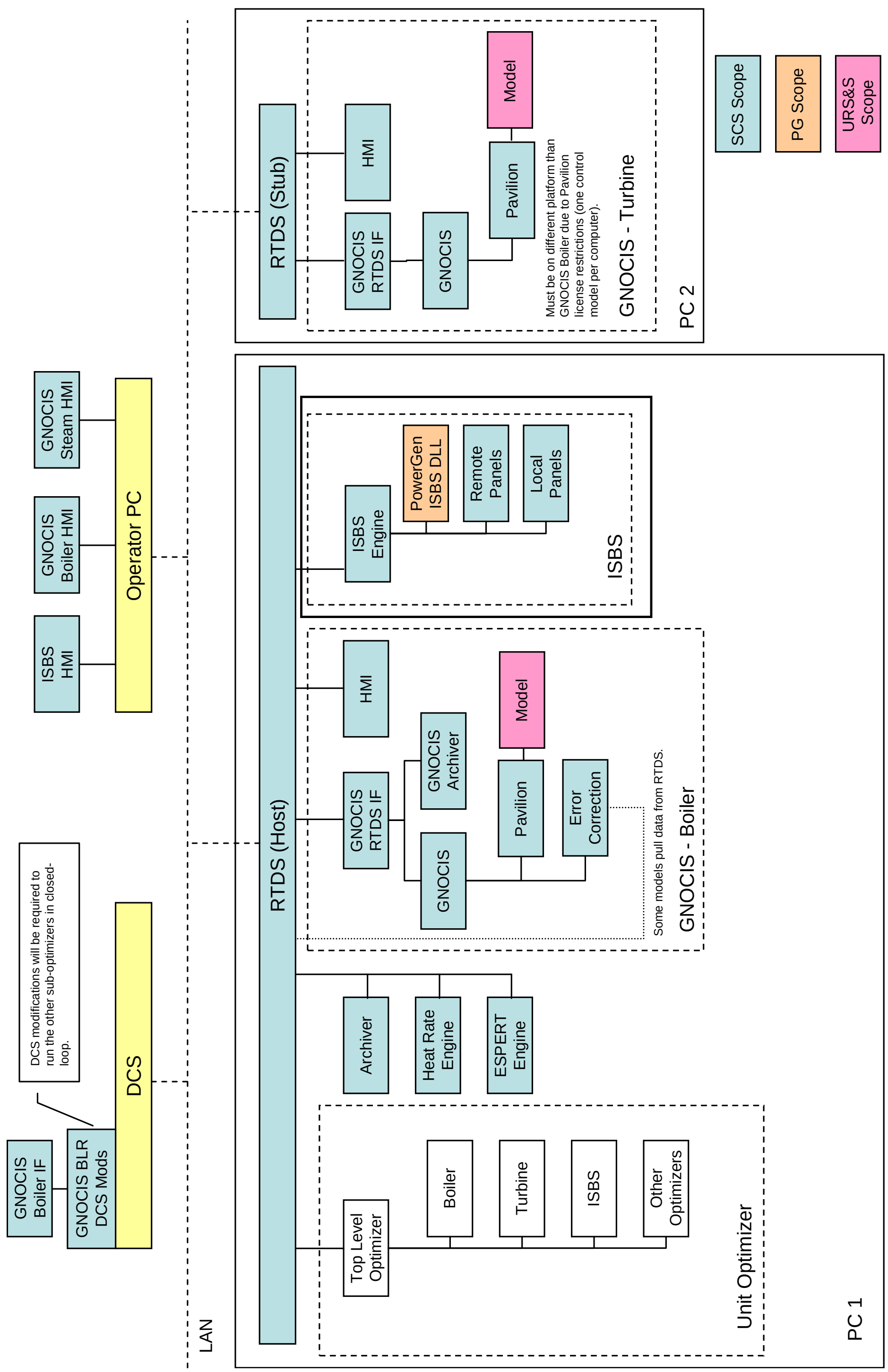


Figure 6-19 ISBS Software in Relation to Other Software Components

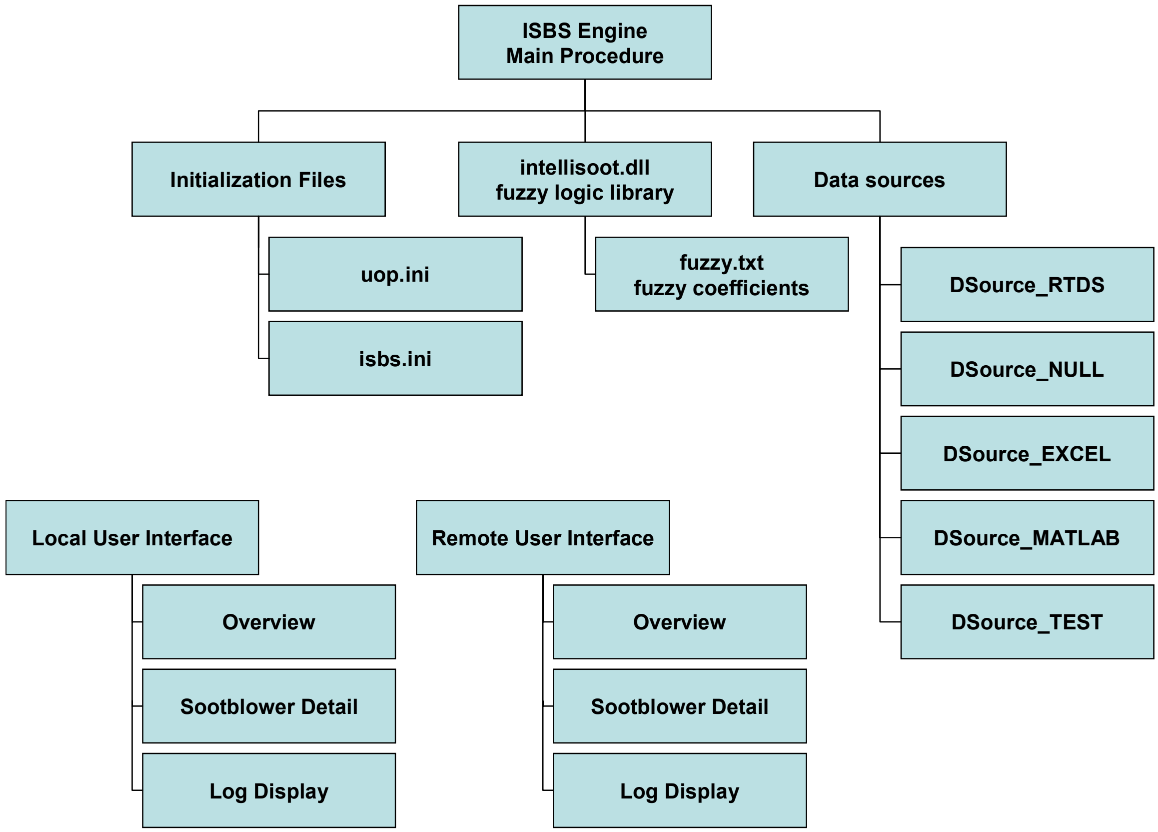


Figure 6-20 ISBS Software Overview

ISBS Engine

This software contains the core of the ISBS calculations as well as the client/server communications capabilities. There are several methods of connecting to this engine to obtain its operational and configuration data. It was designed to allow maximal access to data without compromising the stability of the running process. An internal timer executes the ISBS calculation at a regular, user-defined interval specified in an initialization file, an example of which is provided in Figure 6-21. Results are extracted via one of the many client software options, and are also written back to the RTDS. The ISBS Engine performs functions such as time-averaging of the inputs, determination of when a sootblower group is being utilized, and marshalling the data for calling the Powergen developed Intellisoot.dll.

```
[General]
RTDSHost = HostComputerName
LoadTag = "4CP001_I:MAIN_4.PNT_3"

CycleTime = 20
FilterConstant1 = 0.8
FilterConstant2 = 0.8
Level1 = 0.7

LogLevel = 200
LogFile = c:\temp\isbs_log.txt
LogToFile = 0

;Location of the fuzzy.txt file
WorkingDirectory=C:\DATA\Isbs2_VCOM\Code

ErrorLogFile=C:\Data\Isbs2_VCOM\Code\ISBSErrorLog.txt
ErrorLogSize = 255
```

Figure 6-21 ISBS Initialization File (Example)

Client User Interfaces

Description

There are several options for accessing the ISBS engine information during runtime.

- Master client
- Operator client
- Operator client controls

All clients display ISBS engine information, including group recommendations, current sootblower values, engine log messages, and other selected DCS tags such as unit load. The master client has additional control and information available to its user. All clients also possess the capability of starting the ISBS engine; however, with the DCOM configuration set as described above, this should only be possible from the server machine or with the correct administrative permissions.

The information presented by the user clients is divided into three panels: main panel, log panel, and the sootblower detail panel. All panels display the current ISBS engine/connection status specific to their individual process. The Master Client provides more control of the ISBS package than the Operator Client allowing re-initialization of the package.

The main panel shows a graphic display of the status of the sootblower recommendations (Figure 6-22). A bar chart is used to visually describe the current recommendations to activate a specific sootblower group. If a recommendation exceeds a predetermined threshold, the bar changes color to indicate a need to sootblow. Other information provided by the main panel includes unit load, reheat temperature and damper position, superheat spray flows and temperatures, and sootblower group activity. Also shown are time stamps indicating the last state of a sootblower group.

The log panel displays messages generated by the ISBS engine for the current log level (Figure 6-23). The log level is determined during initialization of the ISBS engine process (from an initialization file), or by the master client panel.

The sootblower detail panel displays the current state of the individual sootblowers within their respective groups (Figure 6-24).

Master Client

The master client is a standalone application that provides additional control over the lifetime of the ISBS engine process. The master client can display the same information as the operator clients, but can also clear the text log buffer, enable and set the text log buffer file, request a forced ISBS engine shutdown, restart a terminated ISBS engine process, and lock and unlock the ISBS engine into memory (the engine will then run regardless of the number of clients attached).

Operator Client

There are two versions of the operator client. The first is based upon the master client interface (Figure 6-25 and Figure 6-26). The second was constructed using the operator client ActiveX controls (Figure 6-27). This client may be imbedded in a web page or some other program (Excel or Word for example). Since the ActiveX controls create independent connections to the ISBS engine, each control will be counted as a client connection. For example, when the control-based client starts, three connections will be made to the ISBS engine, and the master client will display an additional three connections on its main panel.

There are three operator client controls: the main control, the log control, and the detail control. These controls may be embedded within a web page or within a custom application, such as the operator client. Each control has an independent connection to the ISBS engine, and therefore, will be counted as a connection when activated. If the control is loaded, the connection is active.

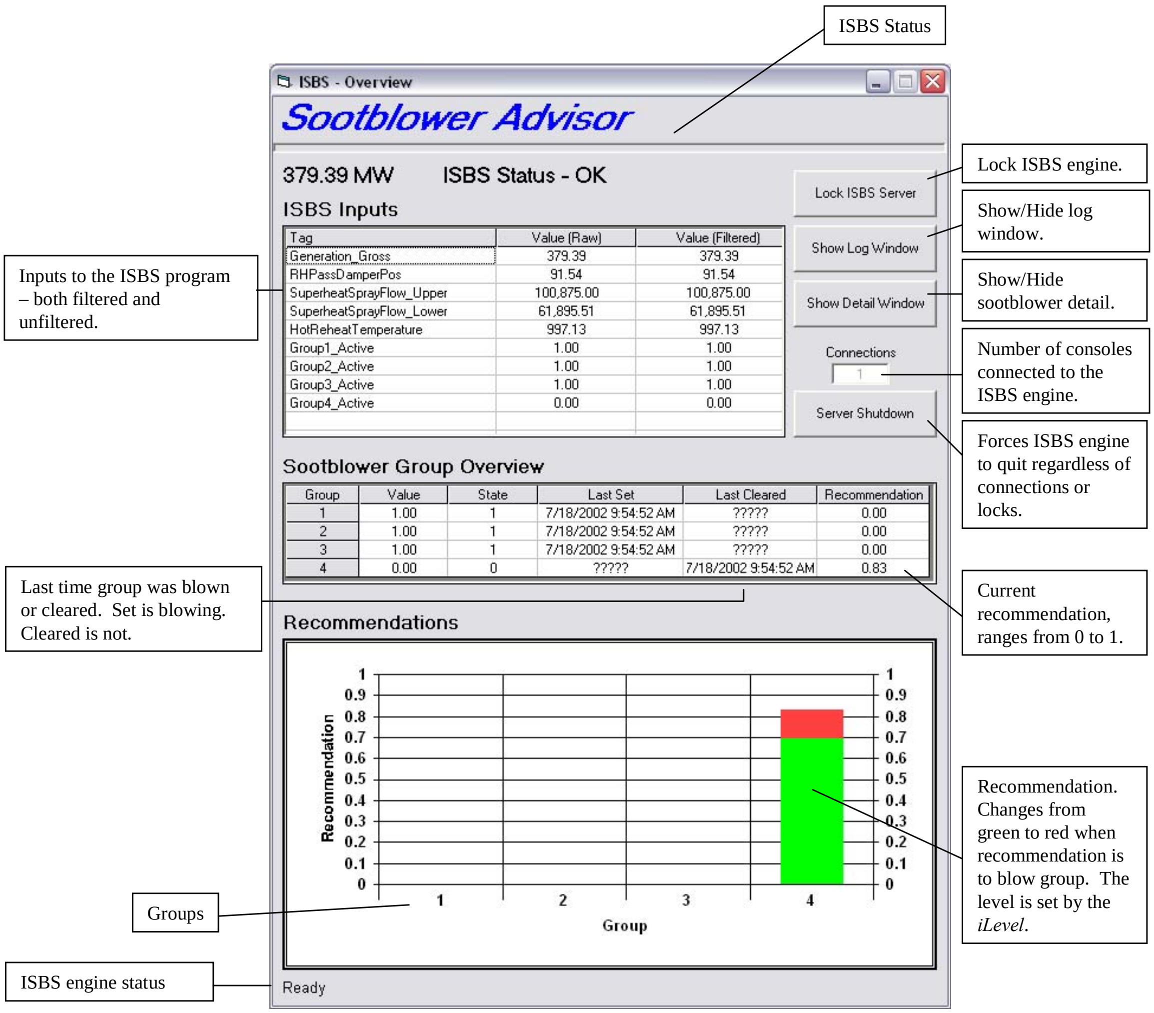


Figure 6-22 ISBS Master Client – Main Display

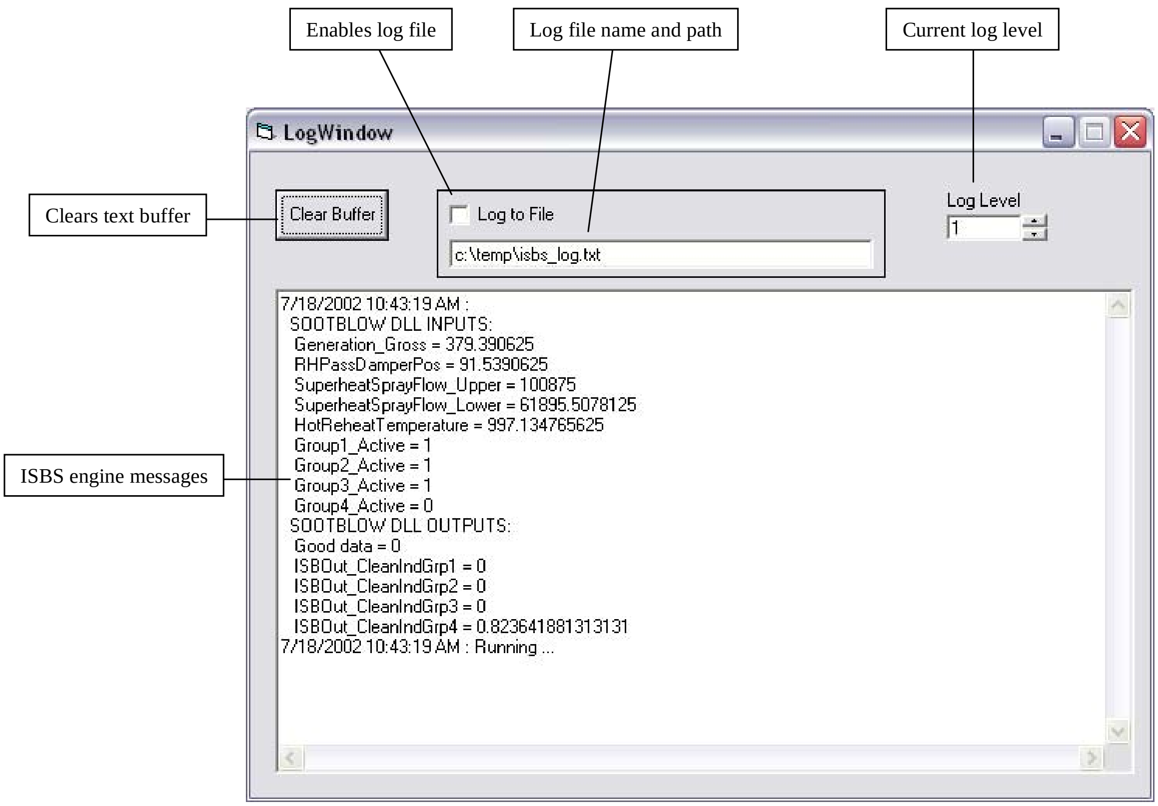


Figure 6-23 ISBS Master Client – Log Display

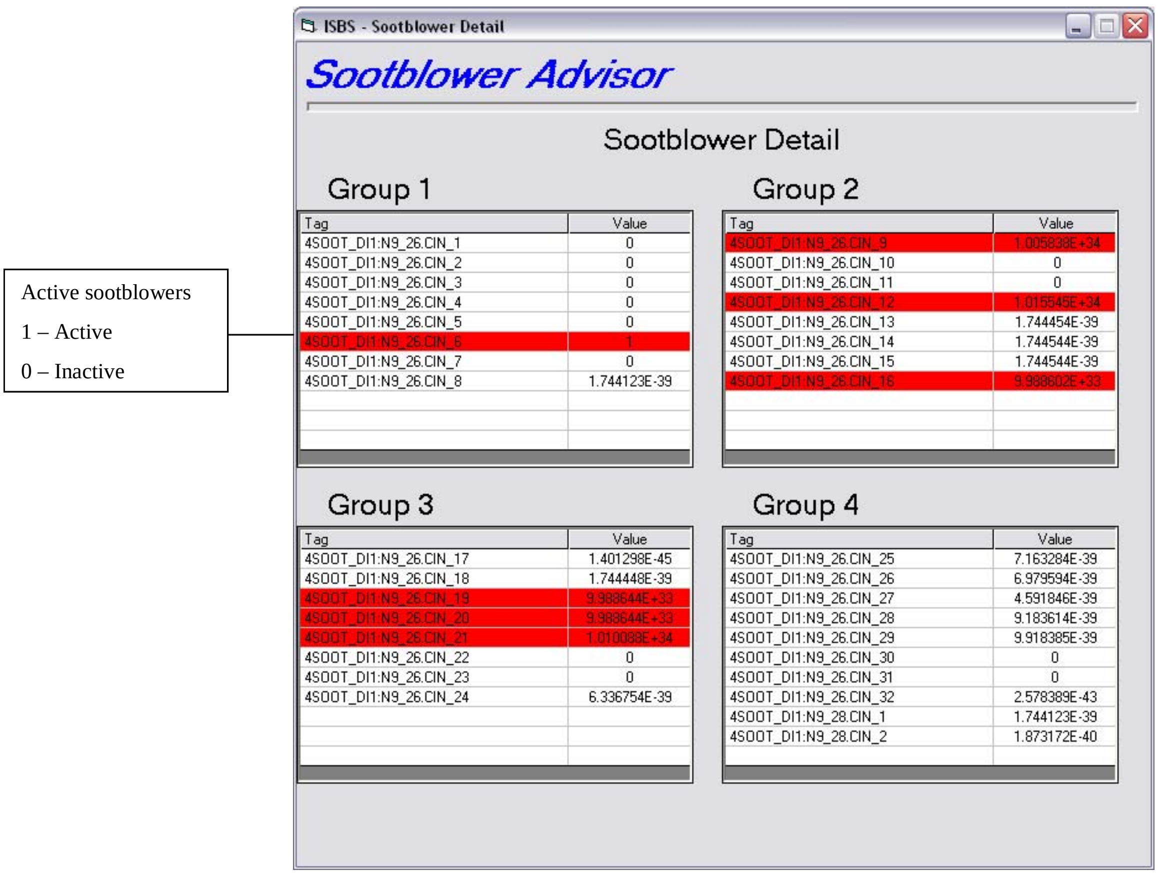


Figure 6-24 ISBS Master Client – Sootblower Group Detail Display

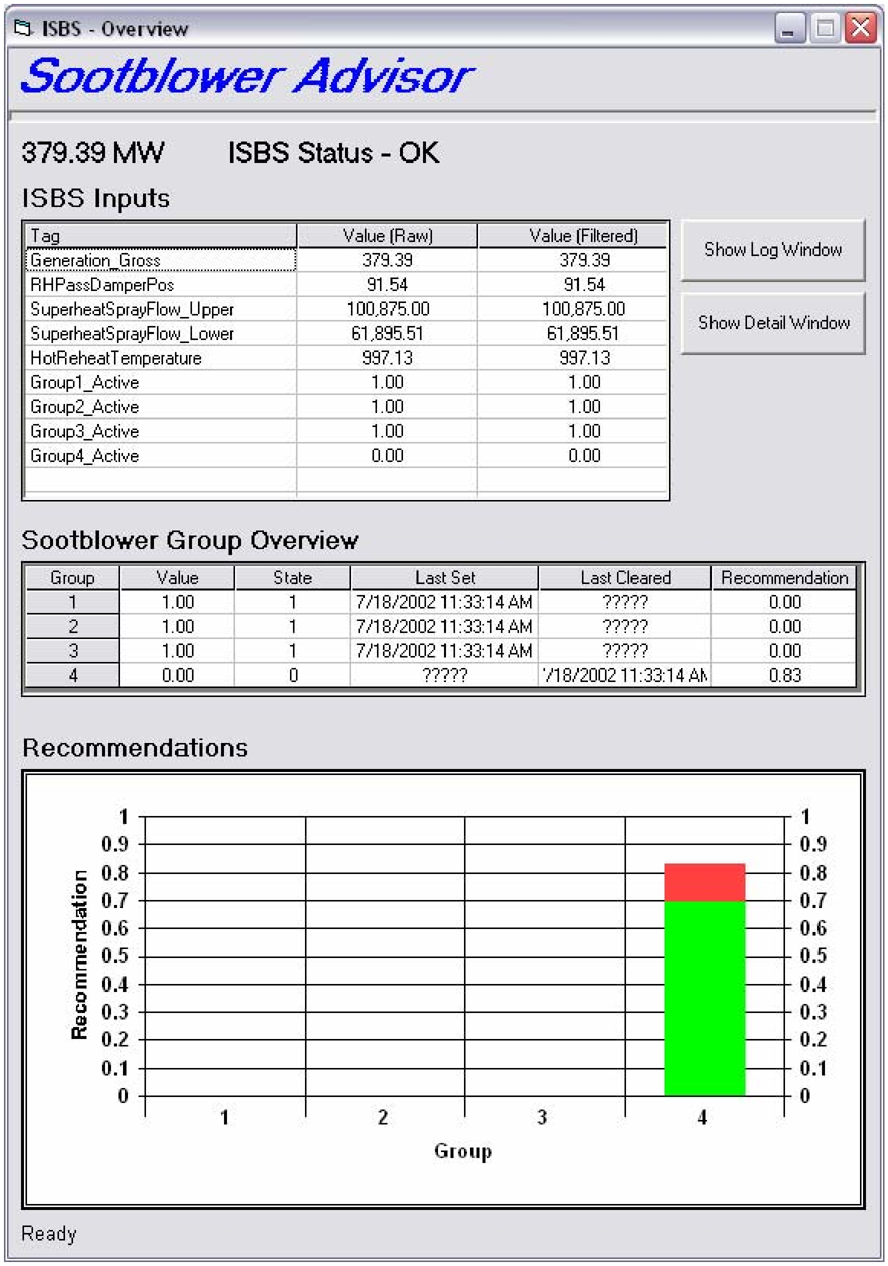


Figure 6-25 ISBS Operator Client – Main Display

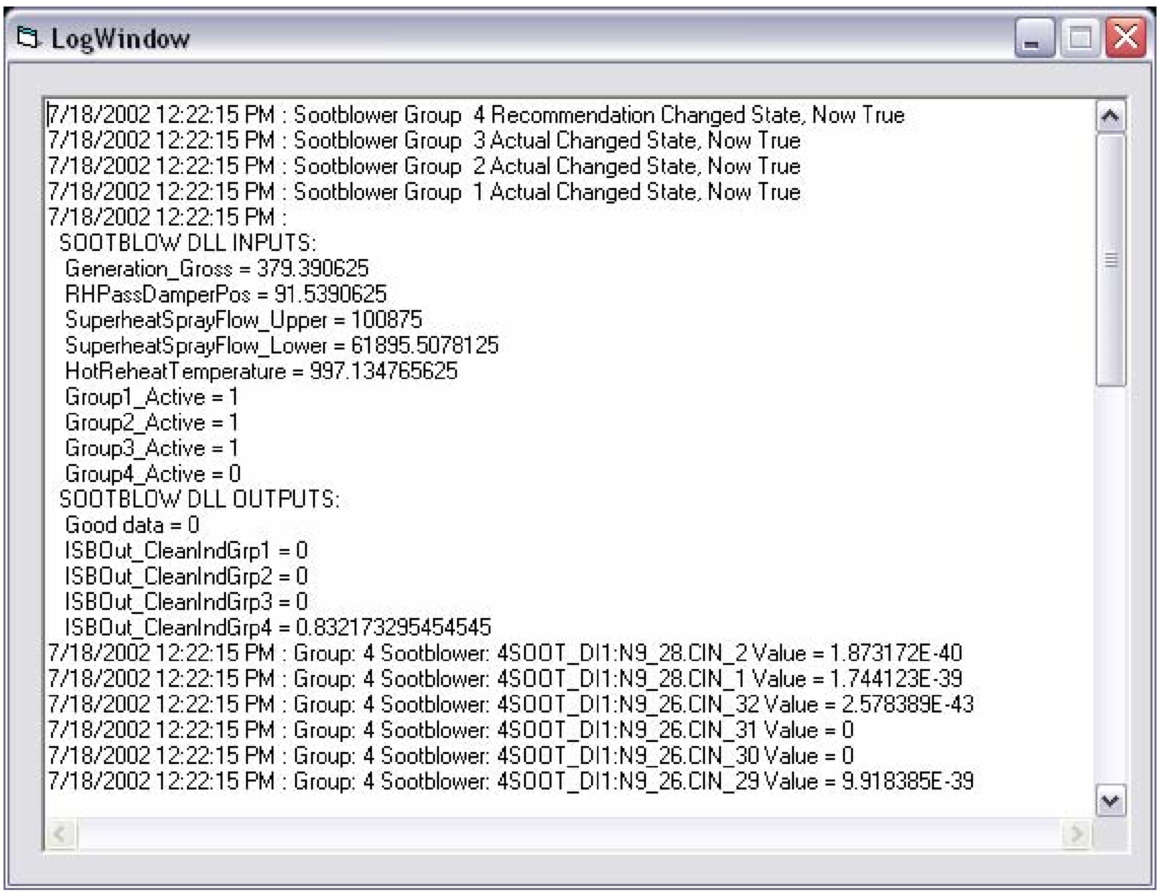


Figure 6-26 ISBS Client – Log Display

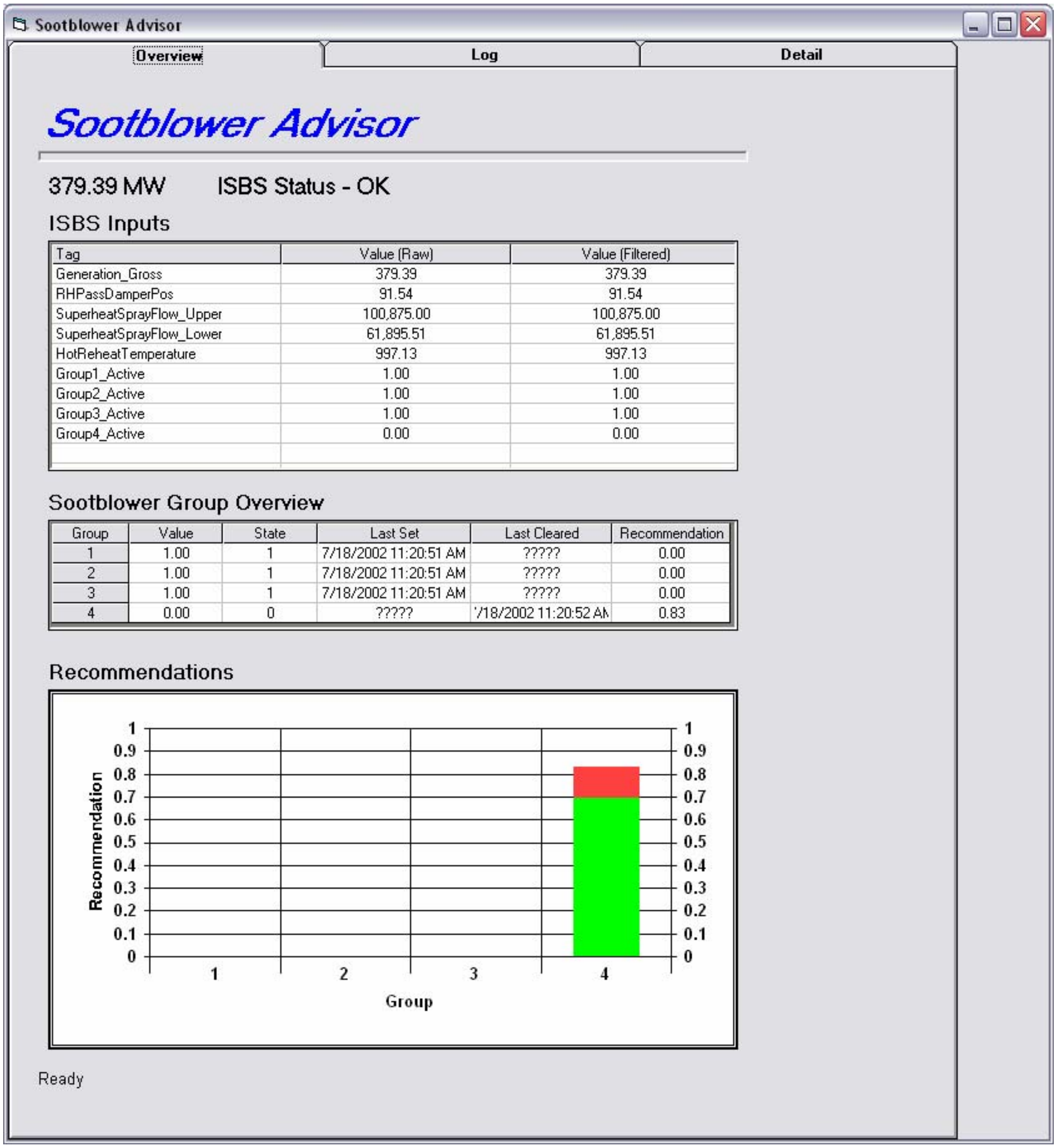


Figure 6-27 ISBS Operator Control-Based Client – Main Display

Performance

A discussion of the performance of the ISBS is provided in the following paragraphs. Only limited testing of this system has been conducted to date.

Testing Conducted January 15 through 17, 2002

Testing was conducted from January 15, 2002 to January 17, 2002 by David Turner (Powergen) and Jim Noblett (URS). The tests generally began with the shift starting at 7:00 am and continued through the afternoon. During this period, the unit was operating normally under economic dispatch. Load and sootblowing activity during this period is shown in Figure 6-28.

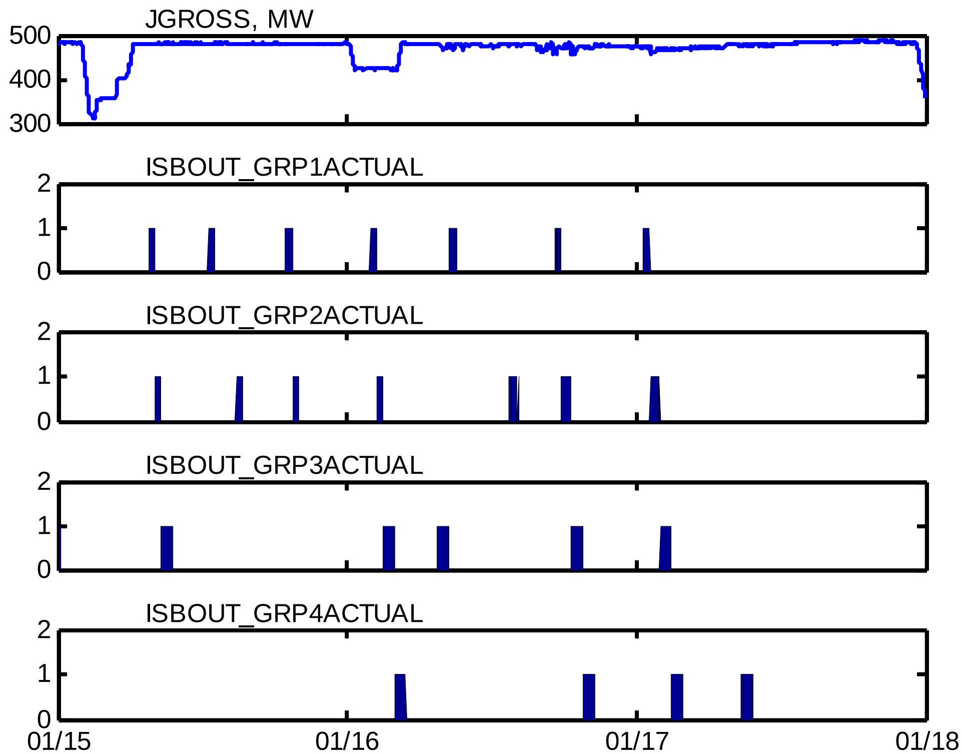


Figure 6-28 Load and Sootblowing Activity – Jan 15-17, 2002

When the ISBS model was started on the morning of January 15, 2002, it recommended blowing Groups 1 and 2 (Figure 6-29 and Figure 6-30). This was done and from the data it could be seen that use of the first group had a bigger effect upon plant parameters than the second group. This led to the idea that Groups 1 and 2 could be used alternately instead of immediately after each other. The model then recommended use of Group 3 and this was done. In all cases, after blowing, the advice returned to "don’t blow". The testing on the first day was excellent with the ISBS software behaving exactly as expected. Group 1 was then blown again in the afternoon. The operator said that currently there weren’t many active sootblowers in Groups 1 and 2 and this was showing up with the advice to blow Groups 1 and 2 before the end of the shift.

The next day, January 16, 2002, the ISBS advised using Group 3 in the morning (Figure 6-31 and Figure 6-32). This was done, though after use, the advice remained at blow Group 3, despite an initial drop in the screen bar showing a return to “don’t blow”. The advice for use of Group 3 needs a small adjustment in the light of these tests. At 8:15 am the recommendation was to blow Groups 1 and 2. Just Group 1 was blown and later at 13:17, just Group 2 blown. The operator had picked up a couple of extra sootblowers in Group 2 from the previous day and the effect was very noticeable with the advice for Groups 1 and 2 coming to zero just 40 minutes after the Group 2 sootblowers were started.

On the final day of testing, January 17, the initial advice was not to use any of the sootblowers (Figure 6-33 and Figure 6-34). At 8:32 am the advice was to use Group 4 and this was done. There wasn’t much apparent effect for the first half hour, though after one hour the damper position changed and the advice to use Group 4 dropped as expected. The testing was concluded after a further six hours when all recommendations for use of sootblowers were low. Based on this, it appears that there is scope to operate plant Hammond successfully with perhaps half the amount of sootblowing that is currently done.

When using Groups 1 and 2, the soot is blown downstream onto the reheater and convection superheater. While Groups 1 or 2 are being used there is a tendency for the advice for Groups 3 or 4 to increase slightly. The soot however doesn’t stop in the downstream heater surfaces since the advice to use either Group 3 or 4 falls back when the use of Groups 1 or 2 stops.

There was a different operator each day during the ISBS tests. They all thought the model would be very useful in enabling them to reduce the amount of sootblowing. This was particularly the case after they were shown that the model advice was based upon spray levels and damper position to assess the cleanliness of different parts of the boiler.

In summary, the testing was very successful and the operators seemed pleased with the display. The model has demonstrated the potential to reduce the amount of sootblowing at Hammond 4.

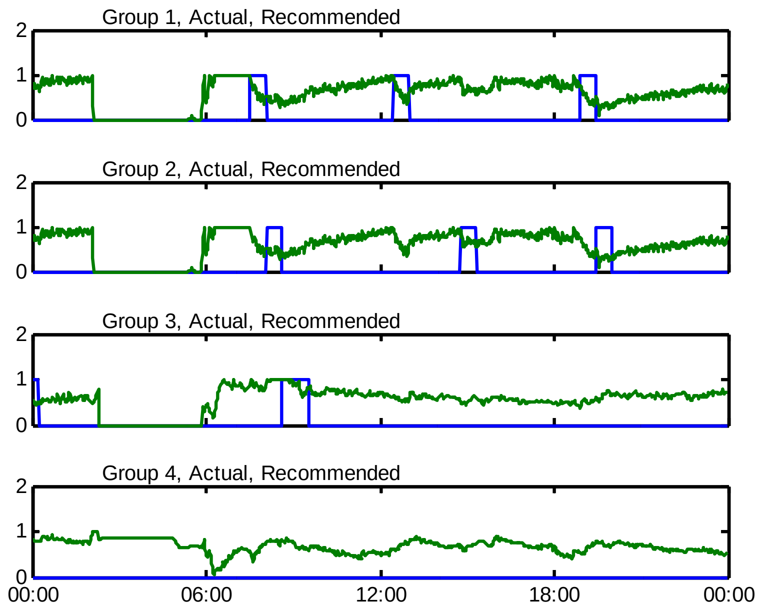


Figure 6-29 Sootblowing Activity and Recommendations– Jan 15, 2002

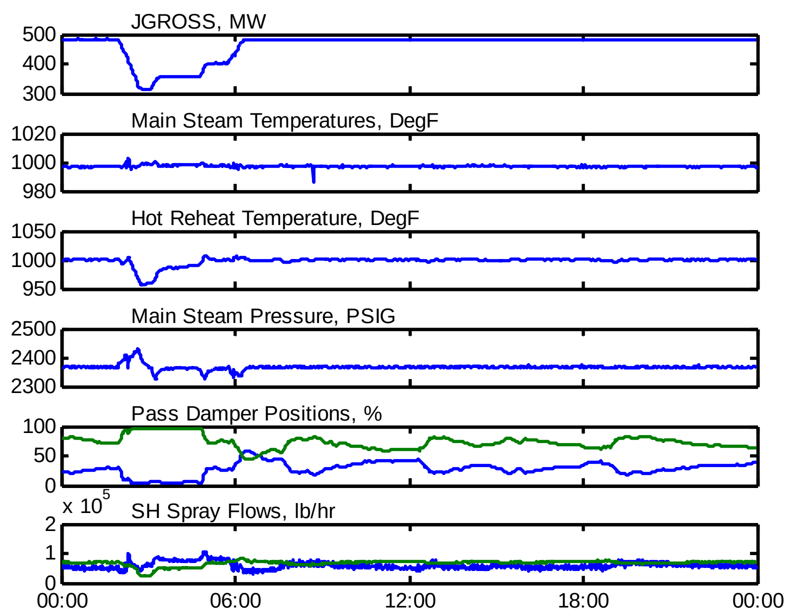


Figure 6-30 Process Data – Jan 15, 2002

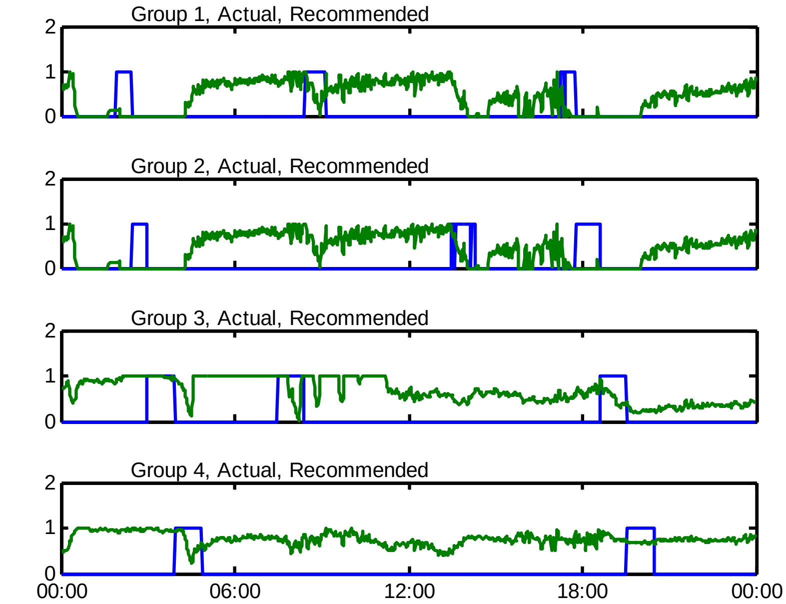


Figure 6-31 Sootblowing Activity and Recommendations – Jan 16, 2002

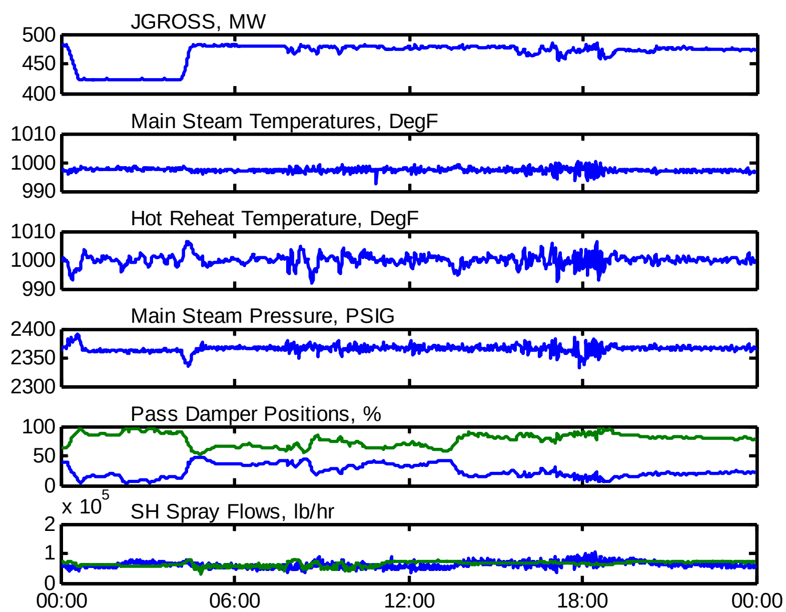


Figure 6-32 Process Data – Jan 16, 2002

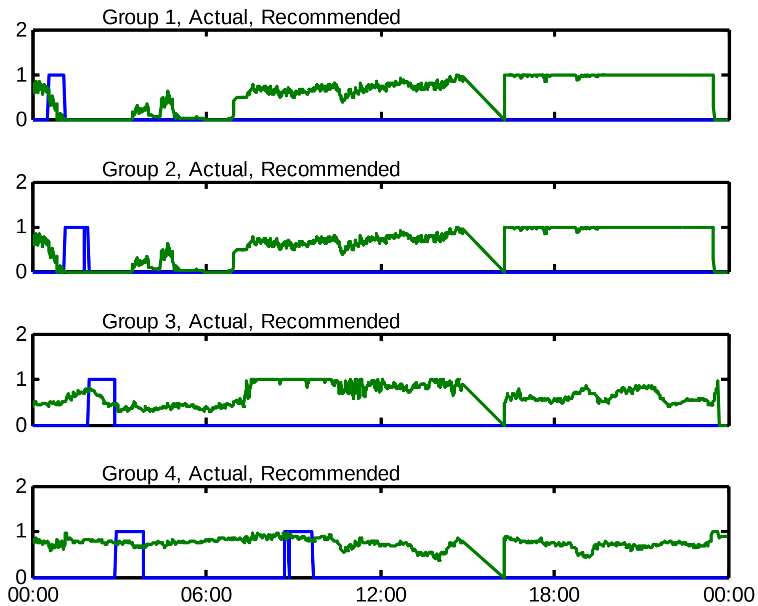


Figure 6-33 Sootblowing Activity and Recommendations – Jan 17, 2002

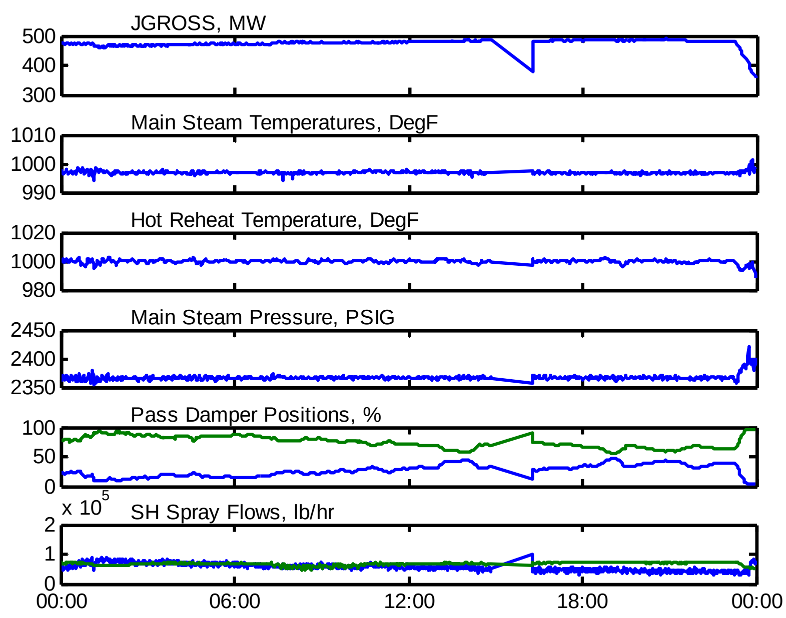


Figure 6-34 Process Data – Jan 17, 2002

Summary

SCS and Powergen developed a intelligent sootblowing system (ISBS) at Plant Hammond Unit 4. The test data collected to date suggests that the ISBS package demonstrated at Hammond 4 would be a useful tool to provide guidance to operators in sootblowing operation at relatively low cost as compared to other technologies now coming to market. The primary benefit demonstrated is the substantial reduction of sootblowing steam consumption when the ISBS advice is followed with no deleterious effects on other operating parameters. This reduction would have a direct beneficial impact on heat rate, condensate make-up requirements, and tube-life.

The fuzzy rule framework is well suited for this type process modeling for which direct modeling of the process (input-to-output) would be very difficult but there exist among the operators and other plant personnel extensive knowledge of process interaction and limitations. As such, the ISBS could be used to facilitate the transfer of best-of-practice among current and future operating personnel.

The ISBS software appears to be sufficiently robust to serve as an advisory package in a production environment. The client-server approach provides for flexible deployment and the ISBS displays are informative to the operator. With a few relatively minor modifications, the ISBS package could be incorporated into a closed-loop system initiating sootblower operations automatically without operator intervention. The required modifications include additional constraints on the system to insure that automated sootblowing occurs only under permissible operating conditions and transfer of recommendations to the DCS. For the former, the safeguards would be implemented in the DCS rather than in the ISBS software to provide independence and limit progression of software failures to actual implementation. The model software and rule-base are readily modified to reflect changes in the plant and preferred operating conditions.

7
REAL-TIME HEAT RATE PACKAGE

Overview

In support of the project, the Center of Electric Power at Tennessee Technological University was contracted to develop a real-time heat rate monitor for Hammond Unit 4. TTU was contracted to provide their software in December 1999 and an initial version was delivered to SCS during the summer of 2000. After that, the heat rate monitor underwent several revisions with the final version from TTU being transmitted to SCS in March 2001. As their practice at that time, TTU provided SCS with two sets of calculations in two dynamic load libraries (DLLs). The two are the “Direct” Method and the “Indirect” or “CEMS” method. It was SCS’ responsibility to write the interface between these DLLs and the RTDS including time averaging, redundant sensor averaging, error detection, and data substitution. TTU provided an option allowing the user to calculate the thermodynamic properties of the steam and water paths, but for the installation at Hammond, TTU provided these internal to their system using functions they have developed.

The program runs at intervals specified in the initialization file performing the following:

- Obtains process data from the RTDS
- Consolidates and averages this data
- Calls the CEP supplied libraries implementing the performance calculations
- Uploads the results to the RTDS

A description of the technology is provided in the following sections.

CEP Technology Description[1]

Overview

The output/loss method has proved to be the preferred method for power plant performance monitoring ever since EPRI launched a major effort toward power plant performance monitoring and heat rate improvement in the early 1980s. Several publications in the open literature report various aspects of the application of the output/loss method for heat rate monitoring [LMJ+84][LMB+86][GMTS89][LSC+87]. In the approach taken in all these works, the ultimate analysis of coal is needed to monitor the unit performance.

A schematic of the system modeled by the output/loss method is shown in Figure 7-1. This is a generic system used here to explain the method. Units differ from each other in many aspects. For example, what is shown in Figure 7-1 is a bisector air preheater. However, several units are equipped with trisector air preheaters. This is just one example and several other differences can exist between units. The various steps involved in applying the output/loss method are shown in Figure 7-2.

It is obvious from an examination of Figure 7-2 that the coal ultimate analysis is needed to start the calculations. However, it is well known that coal ultimate analysis is unavailable in real-time and as a result the well known output/loss method, when used in this form, cannot yield results in real-time. It was proposed to incorporate a novel technique developed earlier of determining coal composition from the gas composition into the output/loss method.

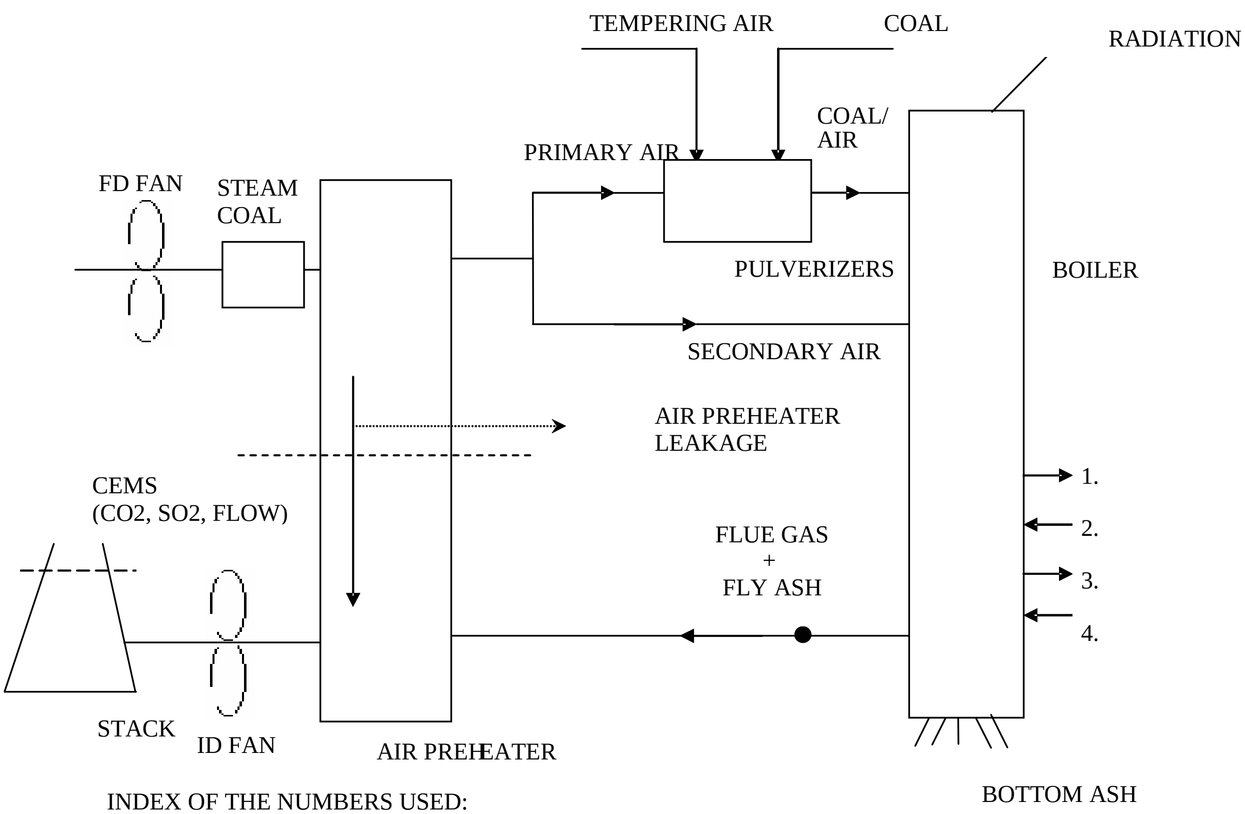
As a result of a feasibility study supported by a consortium of five utilities [MPM88], a new method for determining the elemental composition of coal based on the gas composition was developed. Subsequently, EPRI supported a study to conduct proof of concept experiments in the one million Btu/hr experimental combustor at Southern Research Institute in Birmingham, Alabama [MCO91]. In most of the power plants in this country, the flue gas data is available as CEMS data. Thus, the CEMS data were combined with other plant data, and the output/loss method was invoked to monitor plant performance [MK95][MCK95]. The steps involved in this procedure are shown in Figure 7-3. Since all data is available in real-time, the heat rate is calculated in real-time.

It is to be noted that sufficient information is not available to calculate the complete ultimate analysis. For example, fuel moisture and ash content in the coal cannot be calculated from available data. Similarly, the oxygen content and nitrogen content in coal cannot be calculated

[1] This section is an adaptation of the final report submitted to SCS by the CEP in fulfillment of their scope of work for this project.

from available data. However, if additional data (for example, the moisture content in the flue gas and nitrogen content in the flue gas) are available, then the fuel moisture and oxygen content in the coal can be calculated.

At this point in time, the data includes CO_2 and SO_2 measured by CEMS in the stack and O_2 measured at the economizer exit. In order for the present method to work, an approximate coal analysis is given as input data. It is, therefore, important to know the effect of each of the approximations on boiler efficiency and heat rate. It is also important to perform a sensitivity analysis for the direct method, as well as the real-time method.



INDEX OF THE NUMBERS USED:

1. HOT REHEAT
2. COLD REHEAT
3. MAIN STEAM
4. FEEDWATER

Figure 7-1 Schematic of the System Modeled By the Output Loss Method

Step 1: Starting from the ultimate analysis of coal, determine the chemical formula for coal containing one mole of carbon. Write down the combustion equation and, based on data on $\%O_2$ and CO in flue gas and unburned carbon in ash, complete the combustion equation.

Step 2: Determine amounts of total air and flue gas per mole fuel carbon. Divide total air into primary, secondary and leakage components. Use air preheater inlet and outlet data and data on coal mills. If total coal flow is needed, use a first approximation (90%) for boiler efficiency and determine approximate coal flow.

Step 3: Knowing all the flow streams in and out of the boiler on the fireside, perform energy analysis to determine the losses per unit mass of coal.

Step 4: Using temperature, pressure and flow rate data on feedwater, main steam and cold and hot reheat steams, determine the total heat transferred to steam Q_S. Alternately, if on-line steam cycle heat rate software is available on the unit, use that information.

Step 5: Using energy balance, calculate the coal flow rate. Use new value of coal flow rate and update calculations in Step 2 and iterate until desired convergence is achieved.

Step 6: Calculate boiler efficiency, coal flow, and heat rate.

Figure 7-2 Sequence of Calculations for Heat Rate Using Output/Loss Method

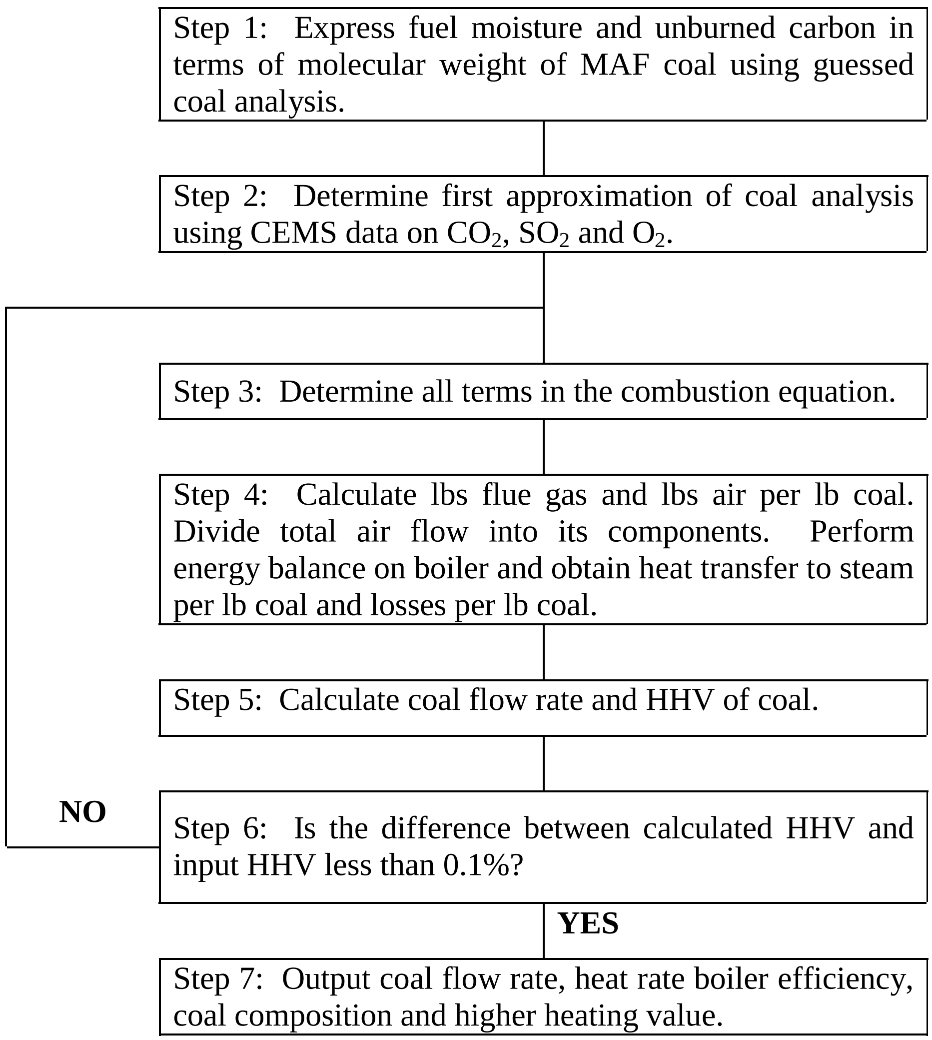


Figure 7-3 Sequence of Calculations for Heat Rate Measurement by Incorporating CEMs Data

Configuration

The TTU calculations are to a large degree boiler efficiency and combustion calculations. The turbine cycle components are only a minor part of their calculations and SCS provided guidance to TTU on what should be included. This consisted mainly of recommendations on methods to calculate steam flows in and out of the boiler control volume, notably cold reheat flow. In the development of the calculations, TTU was provided with unit drawings, available test data, historical plant process data, and typical coal properties.

The TTU library has a total of 59 inputs consisting of 38 inputs from the RTDS and 21 constants, read in from a constants file (Table 7-1). The 38 inputs are derived from 54 actual inputs from the RTDS. Several of these inputs were redundant signals, such as main steam temperature. Nineteen outputs are written back to the RTDS.

Table 7-1 Hammond – TTU Inputs and Outputs

Constants	Inputs	Outputs
Relative Humidity Percent	Fan inlet Temperature (Temp)	DIRECT:AIR_PREHEATER_LEAKAGE
LOI Percent	Prim Air from APH Temp	DIRECT:CYCLE_HEATRATE
CO PPM	Sec. Air from APH Temp	DIRECT:GROSS_HEATRATE
Fly ash (Percent of Total Ash)	Coal Air Temp	DIRECT:NET_HEATRATE
Air Heater Leakage Percent	Gas to APH Temp	DIRECT:OUTPUT_LOSS_EFFICIENCY
Boiler Leakage Percent	Gas from APH Temp	DIRECT:PTC_4_EFFICIENCY
Coal-Air (Air/Fuel Ratio)	Feed water Temp	DIRECT:COAL_FLOW
Maximum continuous rating of boiler	Feed Water Press	INDIRECT:CARBON_PERCENT
Percent Carbon from ultimate analysis	Feed Water Flow	INDIRECT:HYDROGEN_PERCENT
Percent Hydrogen from ultimate analysis	Main Steam Temp	INDIRECT:SULFUR_PERCENT
Percent Sulfur from ultimate analysis	Main Steam Press	INDIRECT:OXYGEN_PERCENT
Percent Oxygen from ultimate analysis	Main Steam Flow	INDIRECT:NITROGEN_PERCENT
Percent Nitrogen from ultimate analysis	Hot Reheat Temp	INDIRECT:MOISTURE_PERCENT
Fuel Moisture	Hot Reheat Press	INDIRECT:ASH_PERCENT
Percent Ash from ultimate analysis	Hot Reheat Flow	INDIRECT:HIGHER_HEATING_VALUE
HHV from ultimate analysis	Cold Reheat Temp	INDIRECT:COAL_FLOW
Blowdown, Percent of FW	Cold Reheat Press	INDIRECT:CYCLE_HEATRATE
Unknown Loss, Percent of HHV	Cold Reheat Flow	INDIRECT:NET_HEATRATE
CO_2 Correction Factor	Outside Air Temp	INDIRECT:BOILER_EFFICIENCY
SO_2 Correction Factor	Fan Room Air Temp	
HSTM Correction Factor	Barometric Press	
	Economizer Out O_2	
	Generator MW	
	Auxiliary MW	
	Stack Flow	
	Stack Temp	
	Coal Flow	
	Stack CO_2	
	Stack SO_2	

Standard for Comparison

The aim of the present work is to develop real-time heat rate monitoring software. In order to validate the software, a standard for comparison needs to be established. The standard for comparison is the direct output/loss method, which henceforth will be referred to as the direct method. In this method, the coal analysis is assumed known and based on this and other plant data, the heat rate is calculated, as shown in Figure 7-2.

The first step needed for validating the real-time software is to use the output from the direct method as input for the real-time software. A consistent set of data is used with the direct method and parameters such as CO_2 percent and of SO_2 PPM in the stack are calculated. These exact values are then used as input for the real-time method. In this case, both methods should give identical results if all the calculations are performed correctly in both methods. This verification was successfully done for Plant Hammond.

In order to validate the real-time method, the following procedure is suggested. Coal samples (preferably by isokinetic sampling in the coal pipes) and ash samples have to be collected, maintaining the unit load constant during that time. The data needed for real-time calculation, averaged over the sample collection time, needs to be identified. Using the respective data, the results from the direct method and real-time method can then be compared.

Sensitivity Analysis of Direct Method

In the direct method the key parameters needed for the calculations are the coal analysis, the percent O_2 at the economizer exit and the PPM of CO at the economizer exit. In order to perform a sensitivity analysis, one particular set of data is chosen as standard data. The "standard input data" is given in the appendix, however, the coal analysis and percent O_2 and PPM of CO are shown below:

Coal Analysis**:**

C	=	70.65
H	=	4.58
S	=	0.76
O	=	5.41
N	=	1.31
H_2O	=	7.88
ASH	=	9.41
		100.00

HHV = 12419 Btu/lb

O_2 at economizer exit = 4.195%

CO at economizer exit = 60 ppm

For this data set, the calculated gross heat rate and boiler efficiency are as follows:

Gross Heat Rate = 8501 Btu/kWh

Boiler efficiency = 90.15%

The percent carbon, hydrogen, oxygen, and fuel moisture are then systematically changed in the coal analysis and, for each case, the calculations are performed using the software. It is to be noted that whenever the percent of any element is changed, the error was compensated for by changing the ash composition such that the total always adds to 100. For example, if coal carbon is decreased from 70.65% to 64.65%, the ash is increased from 9.41% to 15.41%. The percent O_2 and PPM of CO at economizer exit were also changed and the calculations performed.

The results are given in Table 7-2. As shown, an error of 6% in carbon content leads to an error of nearly 50 Btu/kWh in heat rate. A 2% error in hydrogen content results in an error of nearly 135 Btu/kWh. A 4% error in oxygen content leads to an error of nearly 30 Btu/kWh in heat rate. A change in fuel moisture by 6% has almost no effect on heat rate. This observation of the effect of fuel moisture needs further explanation. First, it is to be noted that the sum of ash plus fuel moisture is assumed to be constant. Second, for this coal, the fuel moisture loss is only a very small component of the total losses and any change in fuel moisture has a very small effect on heat rate. Alternatively, for a coal with nearly 25% moisture content, any change in fuel moisture will have a noticeable effect on heat rate.

A change in O_2 at the economizer exit by ±10% from its original value of 4.195 has almost no effect on heat rate (Table 7-3). Similarly, a change in CO at economizer exit by ±100% from its original value of 60 has an insignificant effect on heat rate.

Table 7-2 Effect of Errors in Coal Analysis (Direct Method)

Carbon			
	% C	**HR**	**Δ HR**
	64.65	8554	+ 53
	76.65	8448	- 53
Hydrogen			
	% H	**HR**	**Δ HR**
	2.58	8366	- 135
	3.58	8433	- 68
	5.58	8570	+ 69
	6.58	8640	+ 139
Oxygen			
	% O	**HR**	**Δ HR**
	1.41	8533	+ 32
	9.41	8470	- 31
Moisture			
	% FM	**HR**	**Δ HR**
	1.88	8501	0
	13.88	8501	0

Table 7-3 Effect of Errors in Plant Data (Direct Method)

Excess Oxygen			
	% O_2	**HR**	**Δ HR**
	3.775	8502	+ 1
	4.6145	8500	- 1
CO			
	CO ppm	**HR**	**Δ HR**
	1	8498	- 3
	120	8504	+ 3

Error Analysis of Real-time Method

As explained previously, if sufficient measurements are made of the flue gas components the complete ash-free coal analysis can be calculated. However, since fewer measurements are available, a guessed coal analysis is needed as input to start the calculations. The first error analysis performed, therefore, was by changing the guessed coal analysis. Three sets of results are presented in Figure 7-4. Note that the maximum error in heat rate is 13 Btu/kWh. It is very encouraging to find that deviations in the guessed coal analysis result in insignificant errors in heat rate. It is to be noted that the predicted coal analysis closely follows the guessed coal analysis. Thus, by giving a wrong input for the guessed coal, the predicted coal analysis will be wrong but the heat rate value will be not be affected.

The effect of changing individual parameters is examined next, and the results are given in Table 7-4. The most significant effect is caused by changing the CO_2 in stack. A 10% decrease in CO_2 from 12.5% to 11.25% results in an increase in heat rate by nearly 200 Btu/kWh. The O_2 at economizer exit and air preheater leakage also has a considerable effect.

The importance of the air preheater leakage needs some explanation. Ideally, all the data pertaining to the flue gas composition (CO_2, O_2 and SO_2) should be available at the same location. However, O_2 is measured at the economizer exit, while CO_2 and SO_2 are measured in the stack. It is, therefore, necessary to correct the CO_2 and SO_2 in the stack to those at the economizer exit. The difference in the two sets of values is due to air preheater and other leakages downstream of the economizer. Equations relating the values of CO_2 or SO_2 in the stack to the respective values at the economizer exit as a function of air preheater leakage are built into the program; thus, it is important to perform an error analysis for this parameter.

	COAL COMPOSITION							
	C	**H**	**S**	**O**	**N**	**H_2O**	**ASH**	**HHV**
GUESSED	70.0	4.0	0.75	2.0	1.25	12.0	10.0	12419
PREDICTED	69.9	4.09	0.75	2.0	1.25	12.0	10.0	12419

CALCULATED HR = 8502 Δ HR = + 1

	COAL COMPOSITION							
	C	**H**	**S**	**O**	**N**	**H_2O**	**ASH**	**HHV**
GUESSED	55.0	7.0	1.0	1.0	1.0	15.0	20.0	12419
PREDICTED	59.05	3.17	0.63	1.07	1.07	15.0	20.0	12375

CALCULATED HR = 8514 Δ HR = + 13

	COAL COMPOSITION							
	C	**H**	**S**	**O**	**N**	**H_2O**	**ASH**	**HHV**
GUESSED	50.0	4.0	1.0	10.0	2.0	25.0	8.0	12419
PREDICTED	51.11	3.07	0.55	10.22	2.04	25.0	8.0	12369

CALCULATED HR = 8494 Δ HP = – 7

Figure 7-4 Effect of Change in Guessed Coal Analysis on Results for CEM Method

Table 7-4 Effects of Changing Individual Parameters on Real Time Heat Rate

Parameter	Input Value	Heat Rate (Btu/KWhr)	Boiler Efficiency (%)
Percentage of N in Coal	0.000 (-100%)	8513	90.02
	1.310	**8501**	**90.14**
	2.620 (+ 100%)	8489	90.28
Percentage of O in Coal	4.328 (- 20%)	8502	90.14
	5.410	**8501**	**90.14**
	6.492 (+ 20%)	8500	90.15
Fuel Moisture	6.304 (- 20%)	8503	90.13
	7.880	**8501**	**90.14**
	9.456 (+ 20%)	8499	90.17
Ash	7.528 (- 20%)	8499	90.17
	9.410	**8501**	**90.14**
	11.292 (+ 20%)	8503	90.12
CO ppm in Stack	1.000 (- 98.33%)	8499	90.17
	60.000	**8501**	**90.14**
	120.000 (+ 100%)	8503	90.13
SO2 ppm in Stack	0.000 (- 100%)	8512	90.03
	509.445	**8501**	**90.14**
	1018.890 (+ 100%)	8490	90.26
CO2 % in Stack	11.250 (- 10%)	8713	87.95
	12.500	**8501**	**90.14**
	13.750 (+ 10%)	8325	92.06
Boiler Average O2 %	3.775 (- 10%)	8548	89.65
	4.195	**8501**	**90.14**
	4.6145 (+ 10%)	8454	90.65
Airheater Leakage %	3.5	8541	89.72
	5.5	**8501**	**90.14**
	7.5	8461	90.57

Results from Plant Data

In this section, the results obtained from plant data, spread over a four-day period from October 4 through 10, 2000 will be presented. It is to be emphasized that the data obtained is "as is" data and not data produced as a result of planned tests. It is, therefore, understood that the comparison between the results from the direct method and the real-time method will not meet the rigorous standards set in the section *Standard for Comparison*. However, some useful conclusions can be drawn concerning the performance of the real-time method.

Figure 7-5 shows the CO_2 concentration vs. time. The figure shows that the CO_2 concentration varies roughly between 10% and 13%. Also, CEM calibration cycles can be seen occurring at approximately 8 hour intervals. In order to see if there is any trend in the CO_2 variation with load, the results shown in Figure 7-5 are reproduced in Figure 7-6. While one can see a trend of increasing CO_2 concentration with load, at any given load, there is a wide variation of CO_2 concentration. This CO_2 concentration is affected by the combustion air requirements, which vary with load and unit heat rate. The variation of $\%O_2$ at economizer exit as a function of load is shown in Figure 7-7. A clear trend of decreasing $\%O_2$ with load is evident from this figure. However, at any given load there is considerable variation in the $\%O_2$. This variation may be partly explained by non-steady state behavior of the plant. The variation of SO_2 concentration in the stack vs. load is shown in Figure 7-8.

The behavior of some of the other parameters is examined in the next four figures. The variation of economizer exit gas temperature with time is shown in Figure 7-9. It can be seen that periodically transient behavior takes place. In addition, this temperature varies continuously. This is one of the parameters used in the calculation of heat rate and boiler efficiency. The variation of average mill outlet temperature with time is shown in Figure 7-10. It is to be remembered that this is one of the controlled parameters in the unit. It can be seen from Figure 7-10 that, except for periodic spikes, the mill outlet temperature is fairly constant with time. The variation of measured coal flow rate as a function of load is shown in Figure 7-11. While a clear trend of increasing coal flow rate with load can be easily discerned, at any given load there is considerable scatter in the data. The variation of stack flow rate with load is shown in Figure 7-12. Once again, there is a noticeable trend of increasing stack flow with load and considerable scatter at a given load.

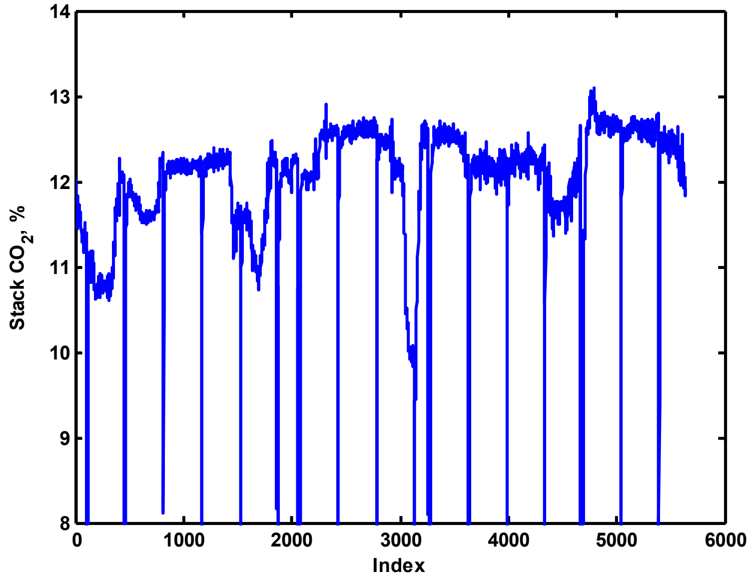


Figure 7-5 Stack CO_2 – October 4-7, 2000

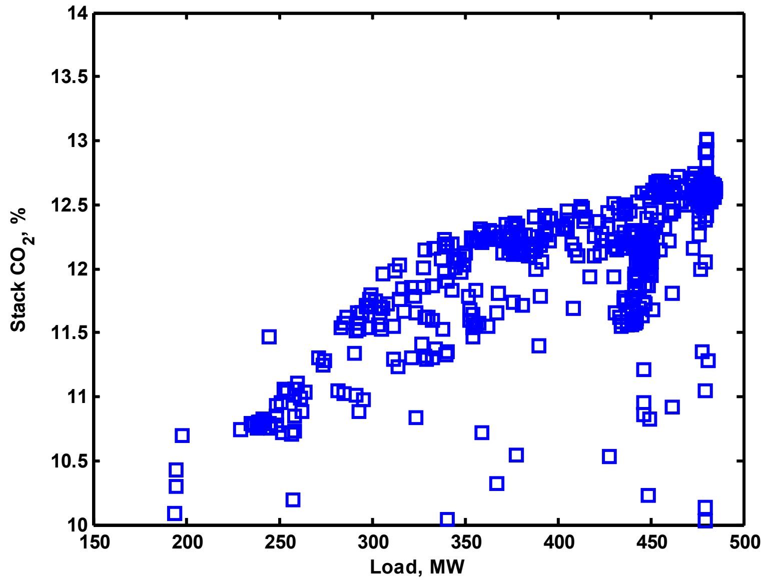


Figure 7-6 Stack CO_2 vs. Gross Load – October 4-7, 2000

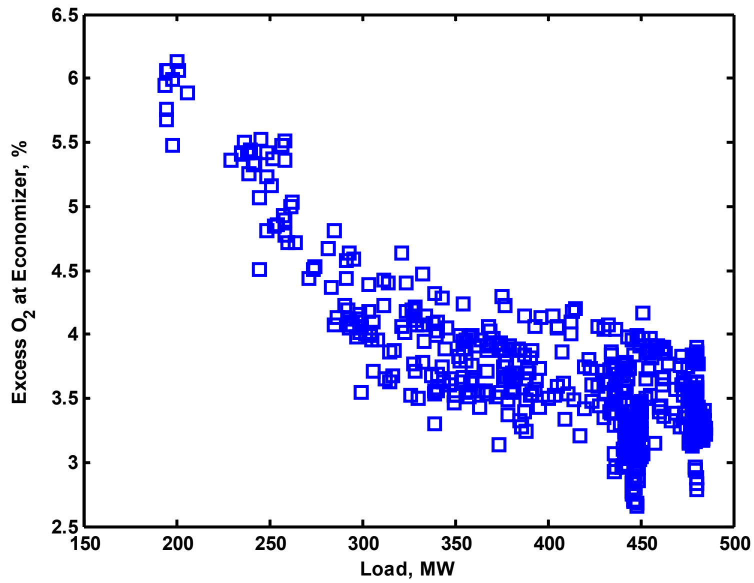


Figure 7-7 Excess O_2 vs. Gross Load – October 4-7, 2000

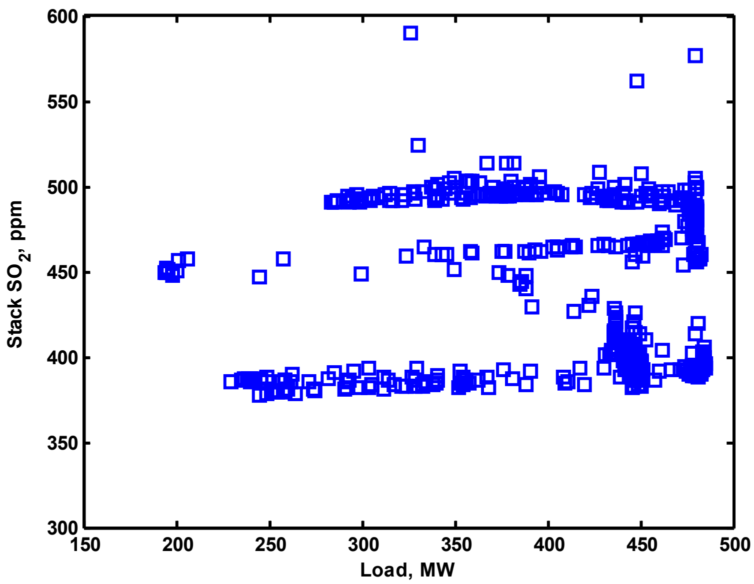


Figure 7-8 Stack SO_2 vs. Gross Load – October 4-7, 2000

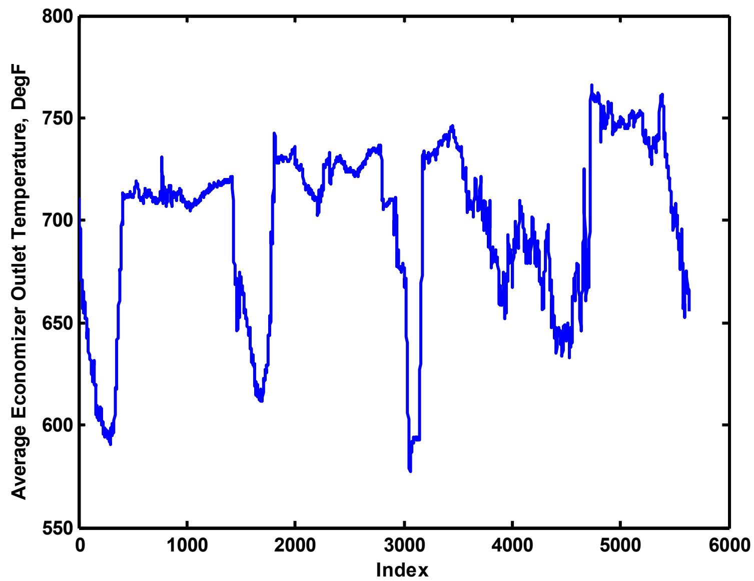


Figure 7-9 Economizer Outlet Temperature – October 4-7, 2000

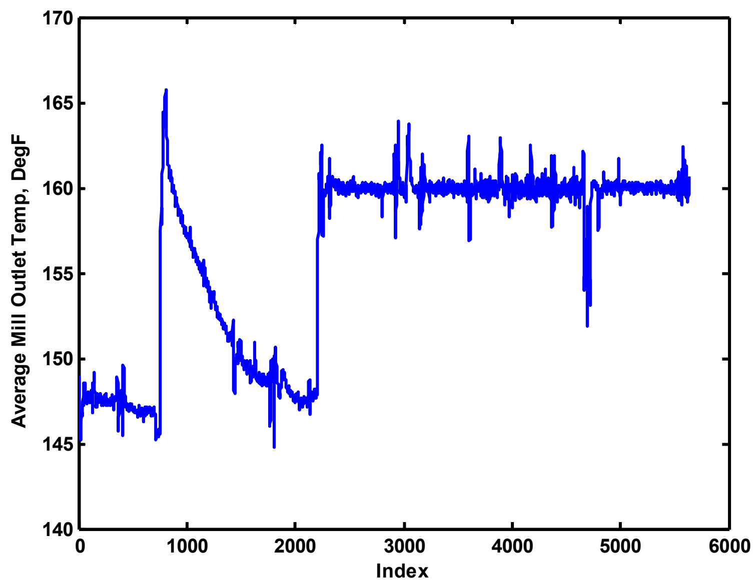


Figure 7-10 Mill Outlet Temperature – October 4-7, 2000

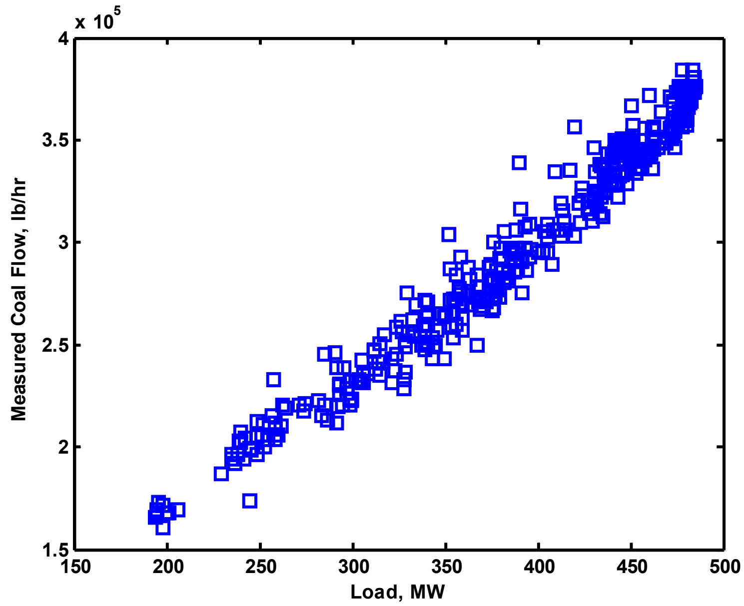


Figure 7-11 Measure Coal Flow vs. Gross Load – October 4-7, 2000

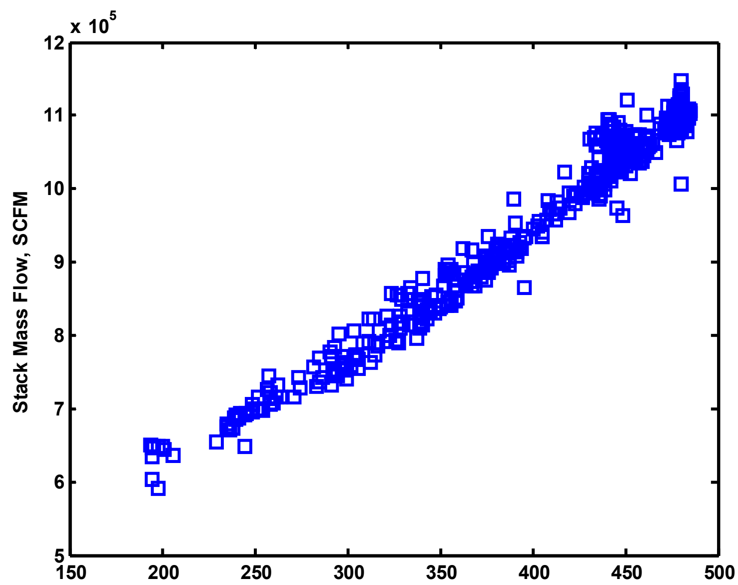


Figure 7-12 Stack Mass Flow vs. Gross Load – October 4-7, 2000

Air Preheater Leakage

The air preheater leakage is an essential parameter to the real-time heat rate method as a means of converting stack emissions measurements to those at the economizer exit. The selection of 5.5 for air preheater leakage in the early phases of this project was based on a hand calculation of a single data set. Now with a data set consisting of 5631 distinct data points over a three day period as shown Figure 7-13, we have the basis for a much more accurate estimate of the air preheater leakage. The direct method as implemented in the CEP code calculates air preheater leakage. The average air preheater leakage for the entire 5631 sets of data is 12.87 with a standard deviation of 3.8. The standard deviation of 3.8 in the air preheater leakage values is of some concern. As shown, the AH leakage determination exhibits spikes at intervals concurrent with the CEM calibration.

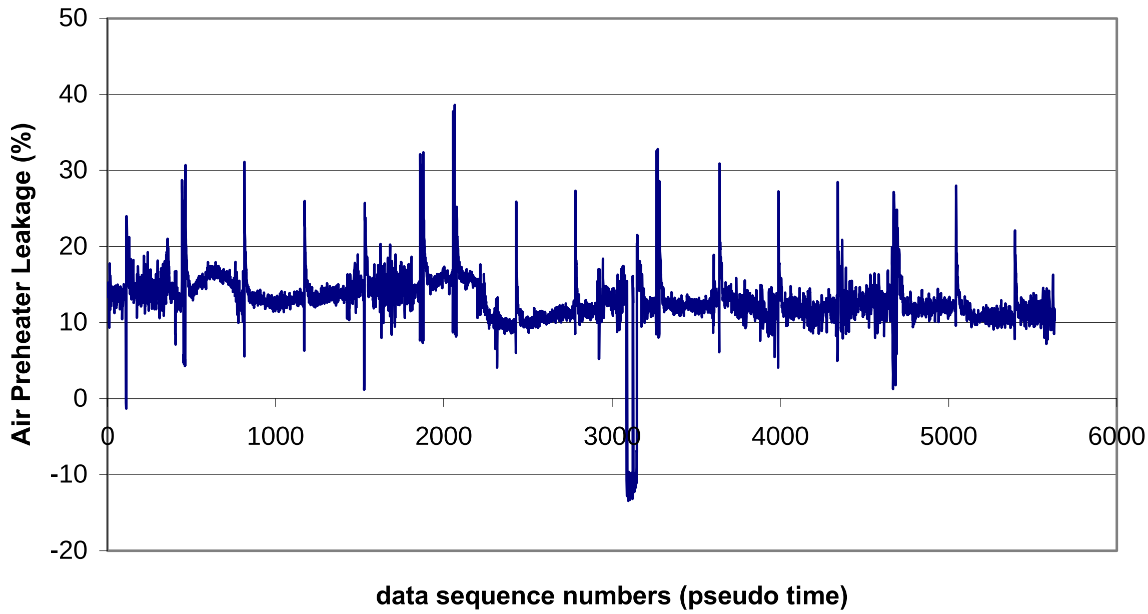


Figure 7-13 Air Preheater Leakage – October 4-7, 2000

Coal Analysis Comparison

A look at the real-time estimation of coal composition will provide insight into how well other computed values compare. This comparison was performed using the new estimate of air preheater leakage discussed above. Table 7-5 shows the average coal analysis computed by the real-time method for the entire 5631-sample data set. It is not exact but is much closer than the estimated coal analysis using the 5.5% air preheater leakage value. Since the coal employed over the three days of data was probably not homogeneous and only one sample was available for conventional analysis, a rigorous analysis of error is not possible. It is to be noted that no constituent varied by much more than a quarter of a percent between the traditional coal analysis and the average real-time predicted coal analysis. It would be interesting to know how the standard deviation of the real-time predicted coal analysis corresponded to the actual variation in coal composition over the 4-day period.

Table 7-5 Coal Analysis Comparison – October 4-7, 2000

Constituent	Hammond Provided Coal Analysis	Real-time Average Coal Analysis	Real-time Coal Analysis (Standard Deviation)	Difference between Provided and Real-time Coal Analysis
Carbon	70.65	70.91	0.72	0.26
Hydrogen	4.58	4.37	0.79	0.21
Sulfur	0.76	0.68	0.07	0.08
Oxygen	5.41	5.43	0.06	-0.02
Nitrogen	1.31	1.31	0.01	0.00
Fuel Moisture	7.88	7.88	0.00	0.00
Ash	9.41	9.41	0.00	0.00

Real-Time vs. Direct Comparison

Now the heat rate, boiler efficiency, and coal flow values between the real-time and direct methods will be compared. Once again the better estimate of air preheater leakage of 12.87% was employed. Table 7-6 shows the results in a tabular form. Some other comparisons are shown in the remaining figures. Figure 7-14 shows a comparison of the coal flow calculated by the real-time method with measured coal flow. The results appear to fall below the 45° line. This is possibly due to a bias error in the coal scales. The real-time boiler efficiency vs. load is shown in Figure 7-15. The real-time coal flow rate vs. load is shown in Figure 7-16. A linear trend of measured coal flow with load can be clearly seen from this figure. The real-time heat rate vs. load is shown in Figure 7-17. At any given load, there is considerable scatter in the data. Once again this is attributable to the scatter in the CO_2 data.

Table 7-6 Real-Time vs. Direct Comparison – October 4-7, 2003

Load (MW)	Direct Boiler Efficiency	Real-time Boiler Efficiency	Direct Coal Flow Rate	Real-time Coal Flow Rate	Direct Gross Heat Rate	Real-time Gross Heat Rate
193.9	92.1	92.0	140229.8	140366.3	8980.5	9022.6
235.5	91.9	92.3	170694.4	170087.7	9001.5	8989.5
253.5	91.7	92.3	173618.0	172722.0	8505.6	8476.2
296.3	91.5	91.1	200180.2	200897.4	8391.2	8435.4
307.8	91.5	91.8	190998.8	190481.7	7707.2	7689.5
312.0	91.6	91.8	211842.4	211431.9	8433.5	8414.3
338.5	91.4	90.9	237228.4	238387.6	8702.7	8754.6
364.4	91.2	91.9	238185.7	236649.6	8117.7	8060.2
364.4	91.2	88.8	245462.6	251505.0	8364.7	8590.3
369.1	91.1	91.7	233027.1	231782.9	7840.1	7788.5
376.2	91.1	91.5	251145.8	250078.2	8290.3	8246.9
379.3	91.1	91.7	259780.9	258262.7	8506.5	8450.6
390.4	91.3	91.6	280826.1	279861.8	8933.9	8898.9
409.1	91.1	91.6	290036.3	288626.6	8805.4	8757.2
420.3	90.8	91.0	296018.2	295168.4	8745.8	8711.1
425.1	90.9	89.9	298888.9	301864.1	8731.6	8825.1
436.8	90.7	90.4	315337.9	316454.4	8965.0	8997.3
442.9	90.7	90.3	322134.8	323554.4	9033.2	9077.2
444.0	90.8	90.3	311772.6	313371.8	8719.6	8762.0
445.3	90.9	91.3	323451.9	321901.7	9021.5	8965.2
446.1	90.7	90.4	327545.4	328411.3	9118.4	9143.1
446.3	90.9	90.9	316658.4	316377.9	8811.0	8794.2
446.4	90.5	90.0	317406.2	318935.9	8830.2	8869.1
447.5	90.6	90.6	317176.9	317101.2	8802.9	8777.5
447.7	90.7	90.7	321188.2	320964.1	8909.9	8884.7
448.0	90.5	90.9	319910.9	318533.4	8867.9	8817.6
448.7	90.7	90.9	314788.3	314007.5	8713.5	8691.2
448.9	90.8	90.9	315569.0	315085.0	8730.2	8703.8
452.3	90.7	90.9	321096.0	320552.4	8817.4	8800.2
459.3	90.6	90.6	319937.3	319896.6	8650.8	8647.8
459.8	90.8	91.3	319145.2	317790.9	8619.2	8580.2
471.5	90.3	90.6	338896.9	337797.4	8926.6	8882.7
478.0	90.6	90.9	346874.9	345588.1	9011.9	8965.5
478.9	90.4	90.7	338618.8	337610.2	8781.1	8740.5
479.2	90.7	91.5	345576.0	343042.5	8955.9	8878.5
479.7	90.6	91.0	347517.6	345876.4	8997.4	8942.3
480.0	90.8	91.6	347593.4	344823.5	8993.8	8910.3
480.6	90.3	90.9	342127.0	340183.5	8840.0	8774.6
480.7	90.3	90.7	342084.1	340457.7	8837.5	8776.0
481.9	90.7	91.1	349360.7	347775.4	9004.1	8951.0
482.4	90.7	90.8	348074.5	347445.2	8960.5	8932.2

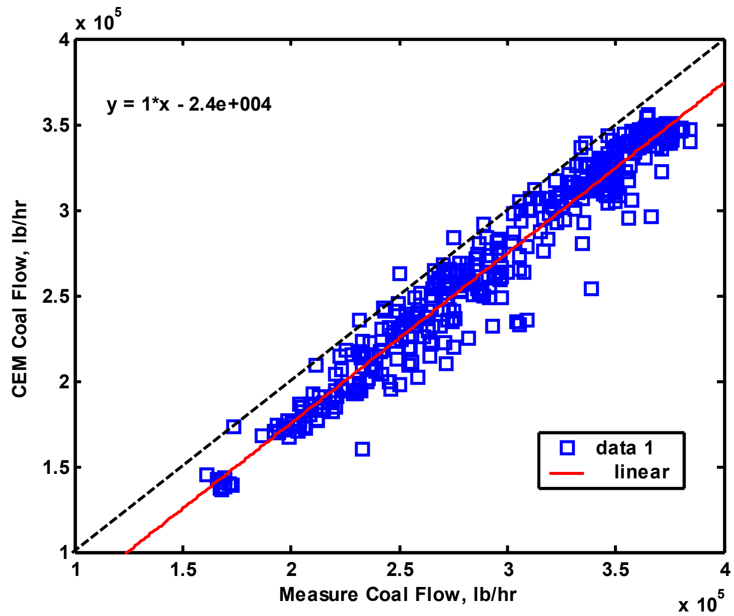


Figure 7-14 Real-Time vs. Measured Coal Flow – October 4-7, 2000

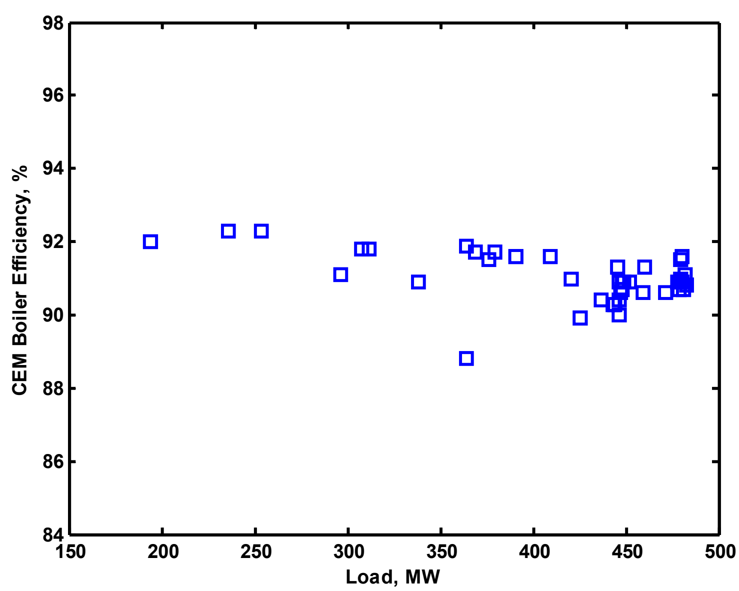


Figure 7-15 Boiler Efficiency vs. Load – October 4-7, 2000

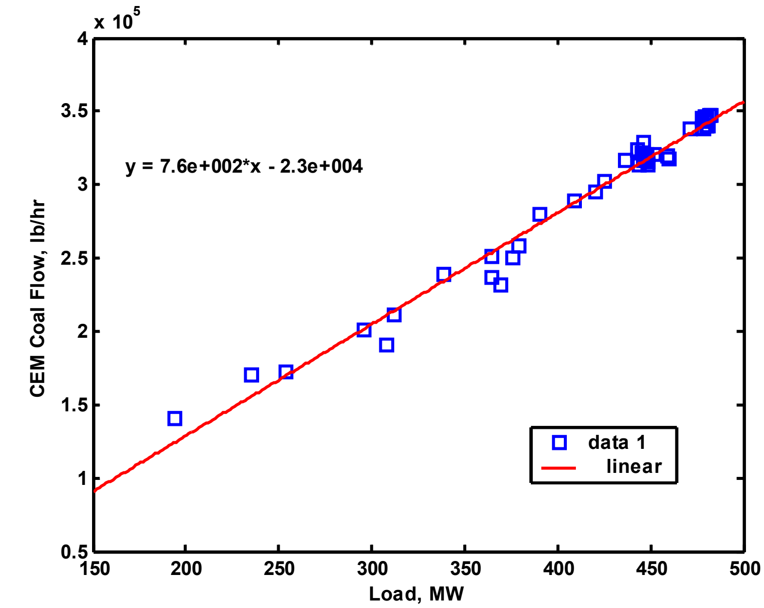


Figure 7-16 Real-Time Coal Flow vs. Load – October 4-7, 2000

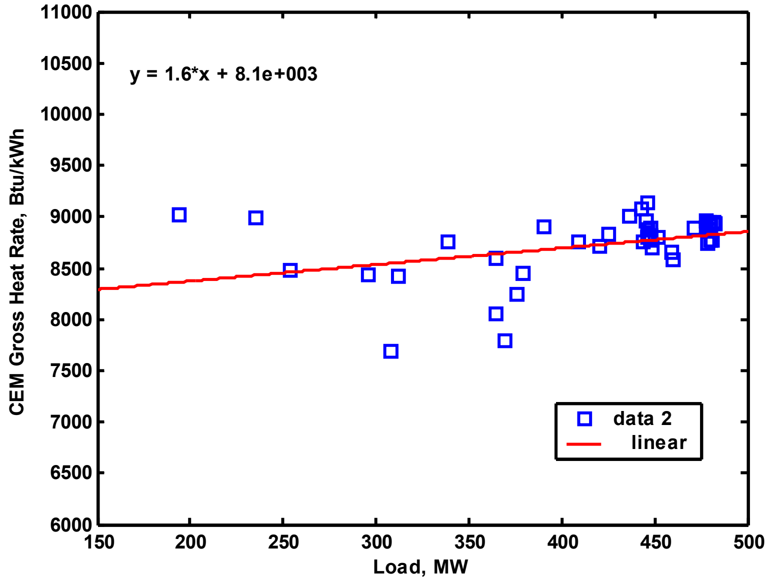


Figure 7-17 Heat Rate vs. Load – October 4-7, 2000

Discussion of Accuracy

The boiler efficiency provides a useful term for comparing the real-time method to the direct method. Figure 7-18 shows boiler efficiency computed for both methods over the 4-day period. Note there are periodic irregularities in the real-time calculation not present in the direct calculation, again these can be seen to correlate with the irregularities in the CO_2 measurement.

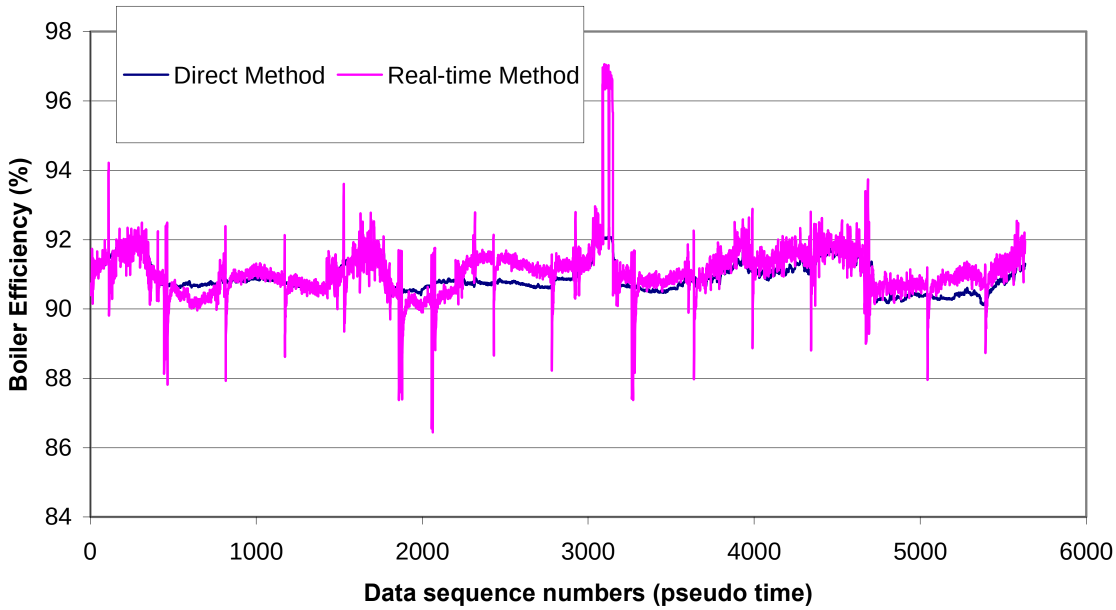


Figure 7-18 Boiler Efficiency Comparison between Direct and CEM Methods for the 4-Day Test

The difference for Figure 7-18 is shown as a histogram in Figure 7-19. The difference has been binned into 76 different error regions and the frequency of occurrence of differences over the 5631 data points for the 4-day test is tabulated and graphed. Note that the standard deviation is about 0.7 and is a good estimate of the error for the whole system, including error in all the sensors, even the periodic “calibration” variances in CO_2. Any improvement in the data provided to the real-time computation will only improve this already low difference.

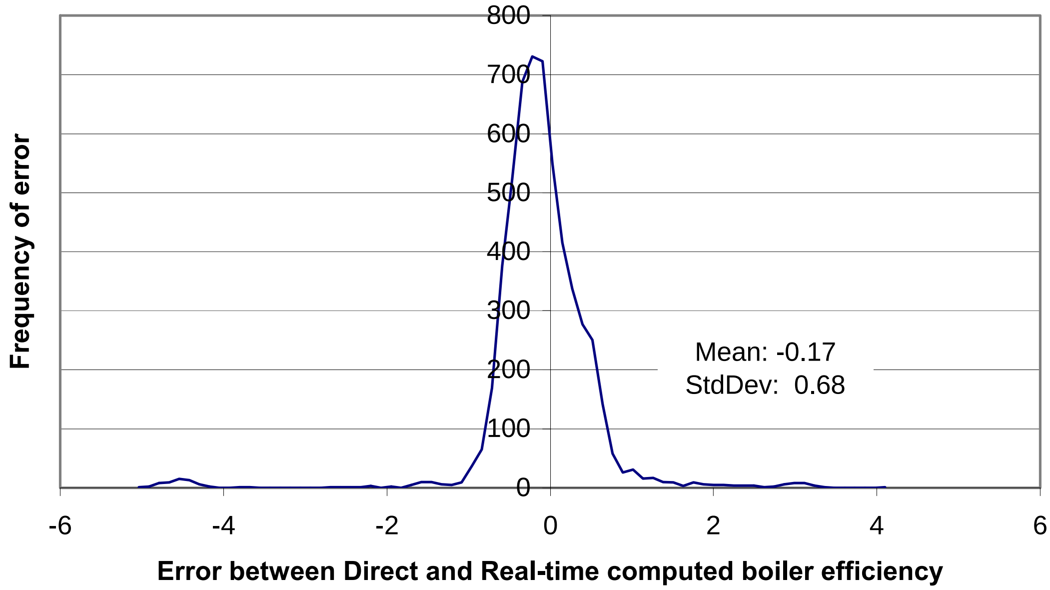


Figure 7-19 Histogram of Difference for Boiler Efficiency between Direct and CEM Methods

Similarly Figure 7-20 through Figure 7-23 show a detailed view into the error of the real-time calculation for gross heat rate and computed coal flow. Each of these shows a bias, but a generally low error when considering the magnitude of typical values for these quantities. In each of the histograms, the "Average" represents a bias which can be calibrated out if sufficient data points are provided. In this context, a data point consists of some plant data and a traditional coal analysis. In each of the histograms, "StdDev" is the standard deviation of the error and is used as a measure of the error of the system for that parameter with the provided data. This includes variations in all the constituent data parameters used in the calculation.

<u>Bias in the Gross Heat Rate and Coal Flow calculations</u>

There are several explanations for the bias in these two calculations. The first is that there is a bias error in the calculation which needs to be, and can easily be calibrated out. The second is that there is an error in the single traditional coal analysis. The direct calculation is based on a sample size of one, whereas 5631 estimates of the coal analysis based on the CEMS data indicate a consistent bias from the single traditional coal analysis. It would be inappropriate to assume all the error was in the method and none in the single coal analysis and a more thorough analysis would require multiple analyses.

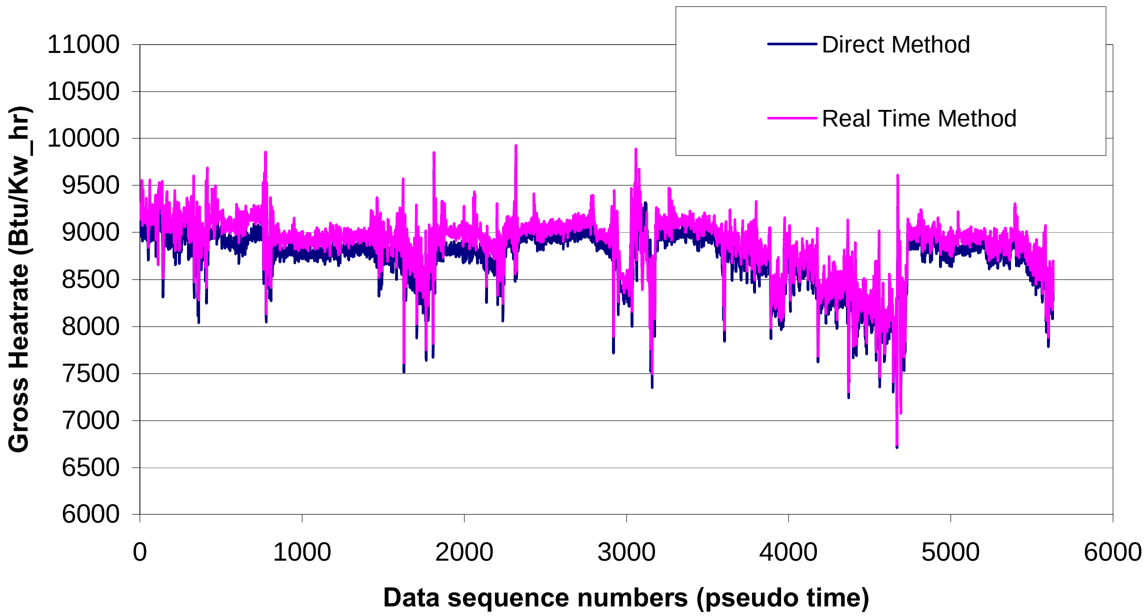


Figure 7-20 Direct and CEM Method Computed Gross Heat Rate

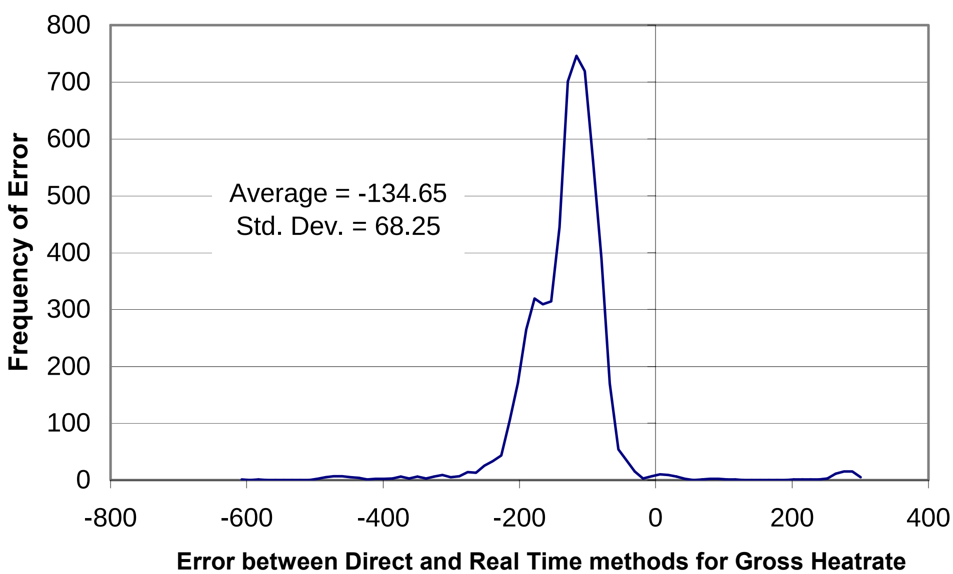


Figure 7-21 Histogram of Difference for Gross Heat Rate between Direct and CEM Methods

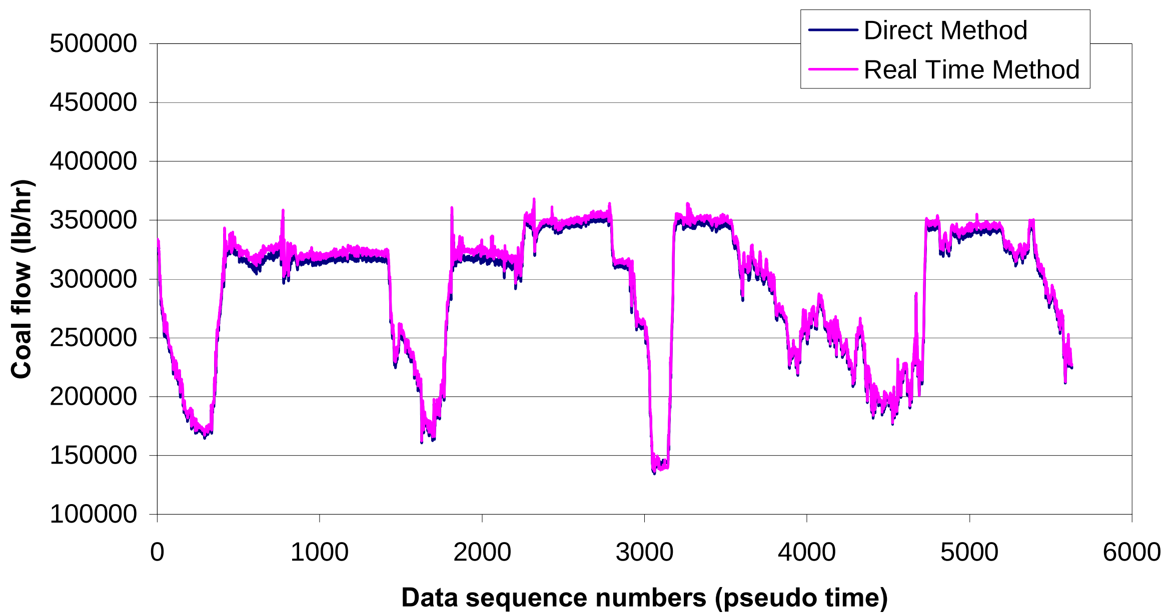


Figure 7-22 Direct and CEM Method Computed Coal Flow Rates

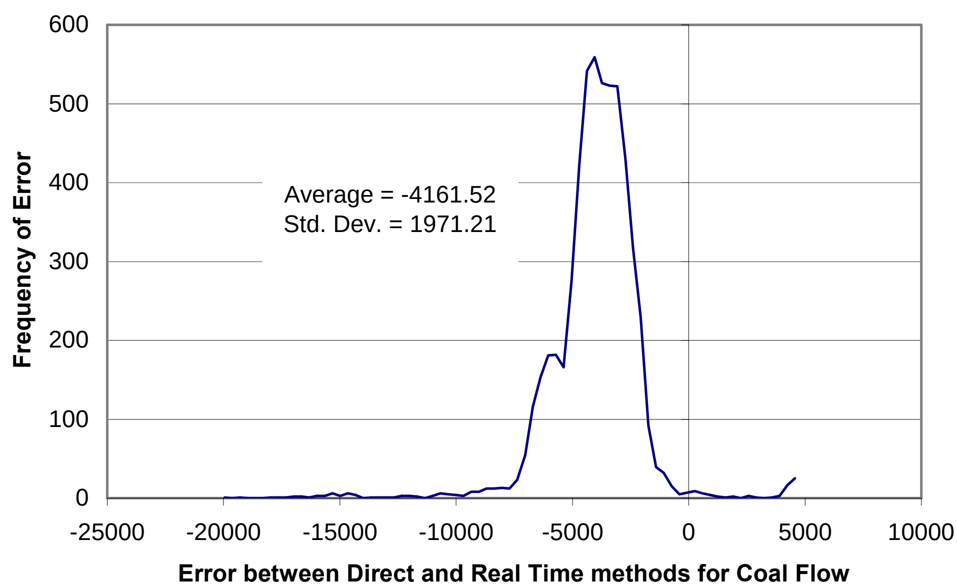


Figure 7-23 Histogram of Difference for Computed Coal Flow between Direct and CEM Methods

SCS Developed Interface

The purpose of the Real Time Heat Rate (RTHR) package is to analyze the heat-rate and efficiency of a boiler unit in real time. The program was designed to run continuously performing its services at given intervals specified in the initialization file. The program pulls information about the power plant from the Real Time Data Structure (RTDS). The information pulled from the RTDS is consolidated and averaged, and sent to the two Dynamic-Link Libraries (DLL) created by the CEP. One DLL analyzes the information using a direct method in which the coal properties are known. The second DLL uses the CEMS data to calculate the coal properties. The calculations are gathered from the DLL's and uploaded to the RTDS. An overview of the package and its relationship with the other components is shown in Figure 7-24 and Figure 7-25.

The interface package provides capability for:

- Initiation and parameter specification through an initialization file
- Error and status logging
- User interface for viewing inputs and outputs to the package and setting options (Figure 7-26)

Presently, the package is configured to run at one-minute intervals.

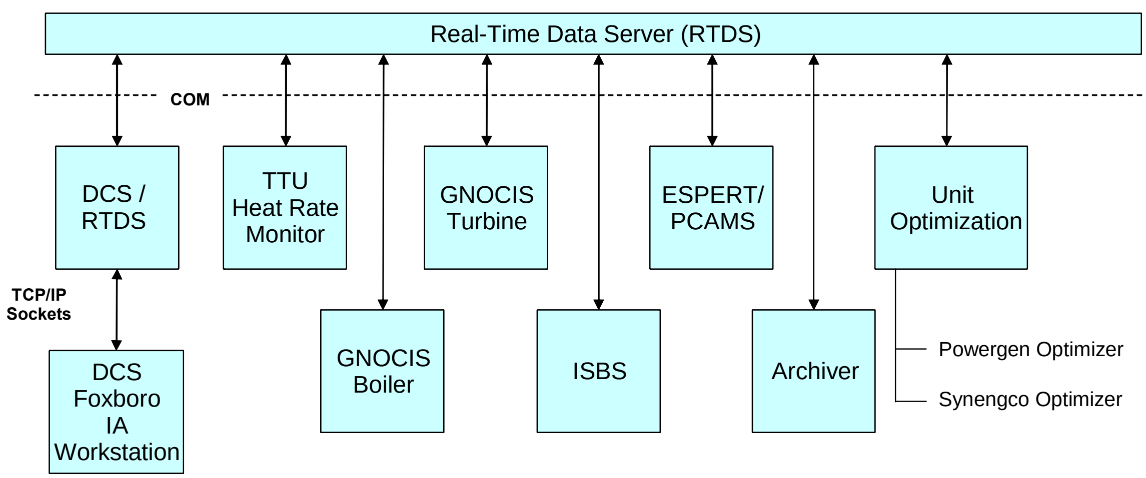


Figure 7-24 Hammond - Relationship of TTU Heat Rate Monitor to Other Software

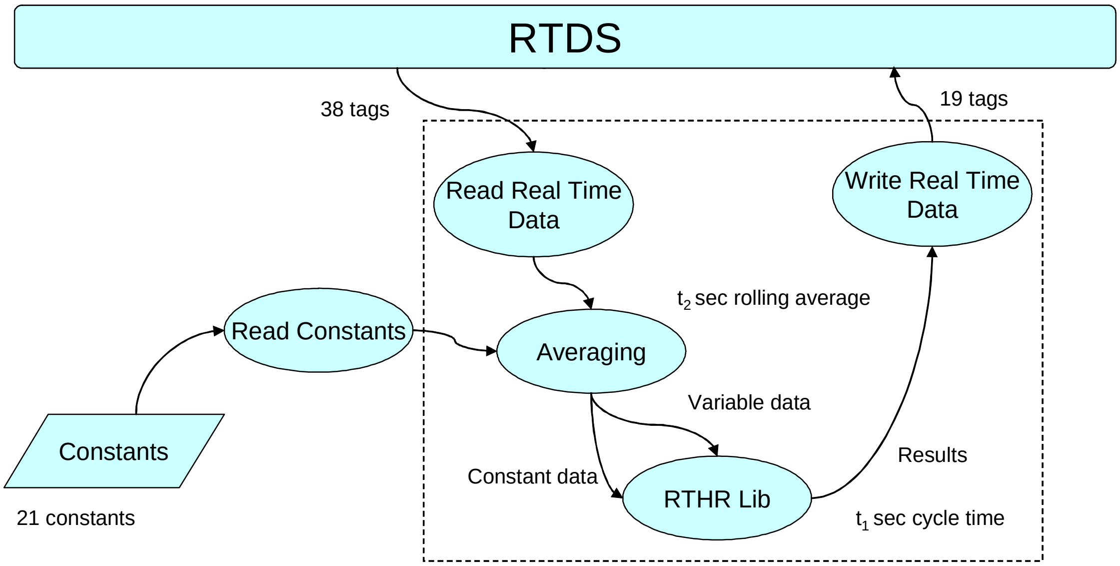


Figure 7-25 Hammond – TTU Heat Rate Monitor Software Overview

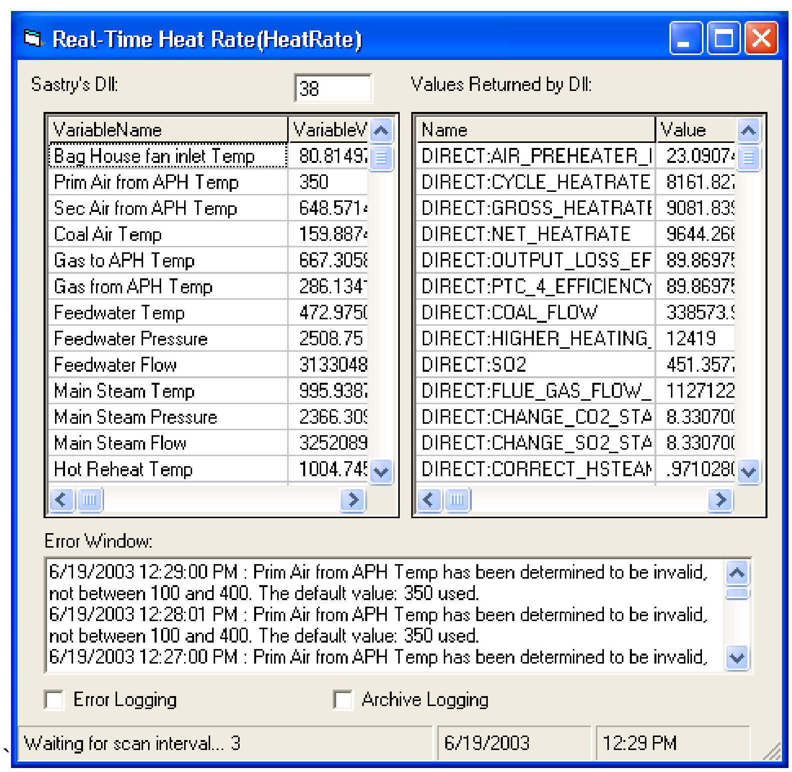


Figure 7-26 User Interface to the Heat Rate Package

Real-Time Heat Rate Monitor Performance

As mentioned, a goal of the installation of the CEP real-time heat rate package was to obtain a more timely and precise measurement of the Hammond 4 boiler efficiency and unit heat rate for use with the optimization packages being installed as part of this project. This section discusses the performance of the package.

Performance During 2002

During 2002, the RTHR package was operating approximately 138 days of the 274 days the unit was operating (Table 7-7). The criteria for determining that that the package was operating was that calculated heat rate was in the range of 5000 to 15000 Btu/kWh. Reasons for the differences between unit operating days and package operating days include software maintenance (RTHR package and other systems on which it depends, such as the RTDS) and failures. The following comparisons are based on this data set.

<u>Gross Unit Heat Rate</u>

The CEP package computes two gross unit heat rate values: the direct measurement in which coal properties are assumed and the indirect measurement where coal properties are estimated from CEM measurements. The latter is of most interest since on-line coal property measurement is difficult and costly. The following is a brief overview of the methods.

<u>Belt Scales</u> – This method uses the daily burn quantities as determined by the belt scales delivering coal to the coal bunkers, the higher heating value of a coal sample collected with the plant’s coal sampling system, and the daily gross generation. This method is generally considered the most robust and precise over extended periods (greater than several days). During shorter periods, the method suffers from problems arising from uncertainties in bunker level measurements and the time delay in the delivery of the coal delivered to the bunker compared to when it is delivered to the furnace. The calculation for heat rate using this input/output method is as follows:

Table 7-7 RTHR Package Operating Days During 2002

Unit Operating Days (Average Load > 200 MW)	274
Days Collected Using RTDS (Average Load > 200 MW)	150
Days RTHR Package Online (GUHR > 5000 & GUHR <15000)	138

$$HR = \frac{m \cdot H}{E}$$

where:

HR = unit heat rate
m = mass of coal delivered during the period
H = heating value of the coal during the period
E = generation during the period.

Feeders – Hammond 4 is equipped with Stock Gravimetric feeders. These provide real-time measurement of coal delivered to the furnace. According to Stock literature, these gravimetric feeders are capable of providing coal feed accuracy of ±0.5%. Applied on a real-time basis, assumptions must be made for the heating value of the fuel, which is dependent on the variability of the coal supply, and this affects the resultant heat rate measured over short periods of time. Using this value and the heating value of the coal delivered to the bunkers that day (using the plants belt sampling system), the unit heat rate may be determined by the input/output method (see above).

CEMS (F-Factor) – The Clean Air Act Amendments of 1990 required the installation of continuous emissions monitoring of stack emissions on most coal-fired units in the US. These amendments include a requirement to measure stack-gas flow and a diluent (CO_2 or O_2) so that heat input to the furnace can be determined. The so-called F-Factor method comes from the constant used in this determination. F-Factor method makes use of existing CEM outputs of volumetric gas flow rate (Q) and stack CO_2 (or alternatively, O_2) to compute the total heat input to the boiler (*W*):

$$W = \frac{Q \cdot (\%CO_2)}{Fc}$$

The EPA "F-Factor", Fc, as described in US CFR 40 Part 75 Appendix F, is defined as the standard volume of product gas per heating value of fuel under ideal stoichiometric combustion with dry air [MK95][EPA00]. The Fc factor used is dependent on the coal type ranging from 1,970 (scf CO_2 / mmBtu) for anthracite to 1,800 (scf CO_2 / mmBtu) for bituminous and sub-bituminous coals. If the heat input is known, the heat rate (*HR*) may be determined as follows:

$$HR = \frac{W}{P}$$

The attraction of this method is its simplicity and that it requires no inputs other than those originating from the CEMs and unit generation. The problem is that it is sensitive to the relatively uncertain measurements of stack gas flow and CO_2. The errors associated with these measurements have been studied extensively and have been reported to be on the order of 5%, leading to an error of approximately 500 Btu/kWh [EPR97]. As described in these references, there is generally a positive bias in these gas flow measurements leading to overestimation of emitted effluents, such as SO_2, and heat rate.

Comparisons of the heat rate values for 2002 are shown in Figure 7-27 to Figure 7-30. At least visually, the belt scale and feeder based heat rate values tracked well during the entire period. The day-to-day variations in the belt scale values may be explained at least in part by the loading patterns in the bunkers. Though not a conclusive indicator of error, the direct and indirect heat rate measurements exhibit a negative bias as compared to the other methods, particularly after third quarter. This bias can be seen more clearly in the distribution of the heat rate values for the various methods (Figure 7-28), the means of which are summarized below in Table 7-8.

Table 7-8 Gross Unit Heat Rate / 2002 Daily Data / All Loads > 200 MW (Btu/kWh)

	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Belt	6101	9305	9595	9605	9964	13500
Feeders	8988	9504	9639	9699	9834	10660
CEM	9229	9552	9885	9918	10230	11160
Direct	8073	8389	8713	8818	8987	14160
Indirect	8190	8558	8843	8946	9101	14270

For all methods, heat rate values below 5000 and above 15000 Btu/lb were considered outliers and are not represented in calculations and only common days were used. Heat rate as a function of load for 2002 is shown in Figure 7-29 (scatter plot) and Figure 7-30 (mean vs. load). As shown, the negative bias for the direct and indirect methods persists over the load range (as compared to the other methods). Similar plots for heat rate during a shorter period (one week) are shown in Figure 7-31 through Figure 7-34. As before, the bias for the direct and indirect methods is evident over the load range, but for this period it is much more pronounced at mid- to lower-load categories.

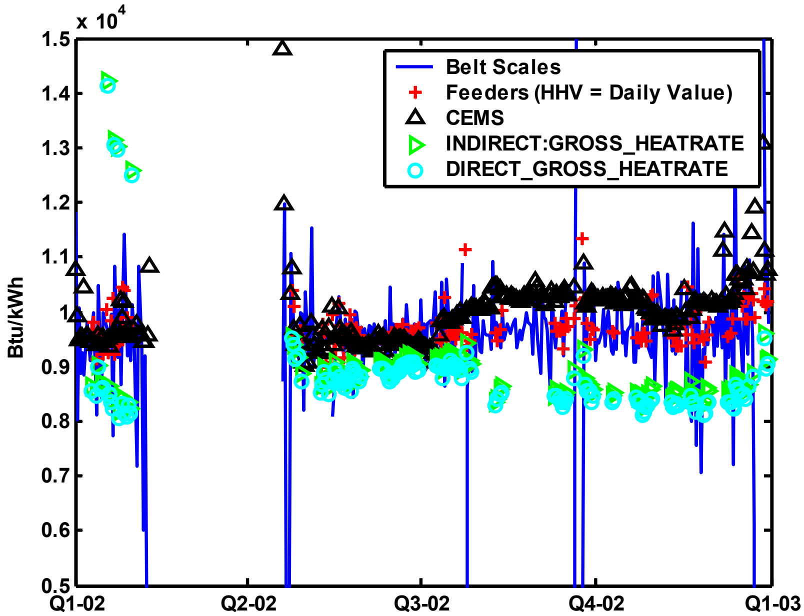


Figure 7-27 Daily Average Heat Rates for 2002

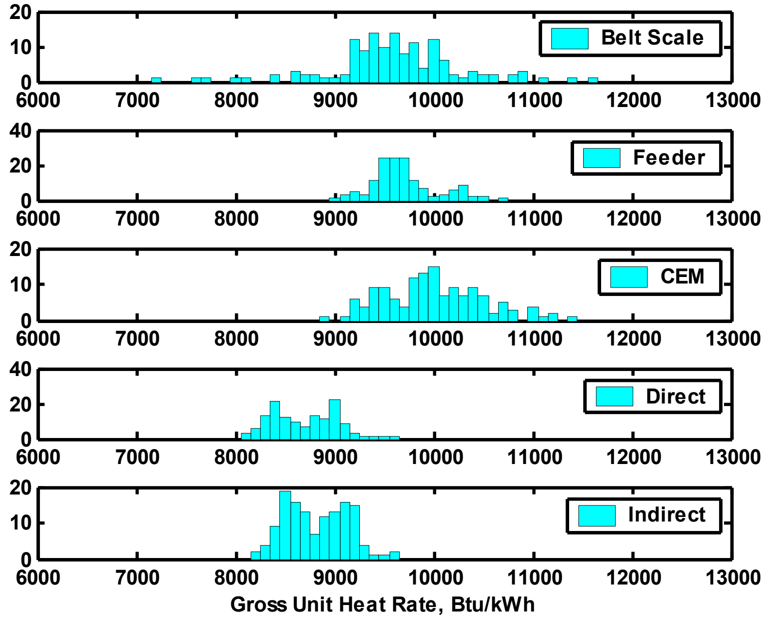


Figure 7-28 Daily Average Heat Rates Histogram for 2002

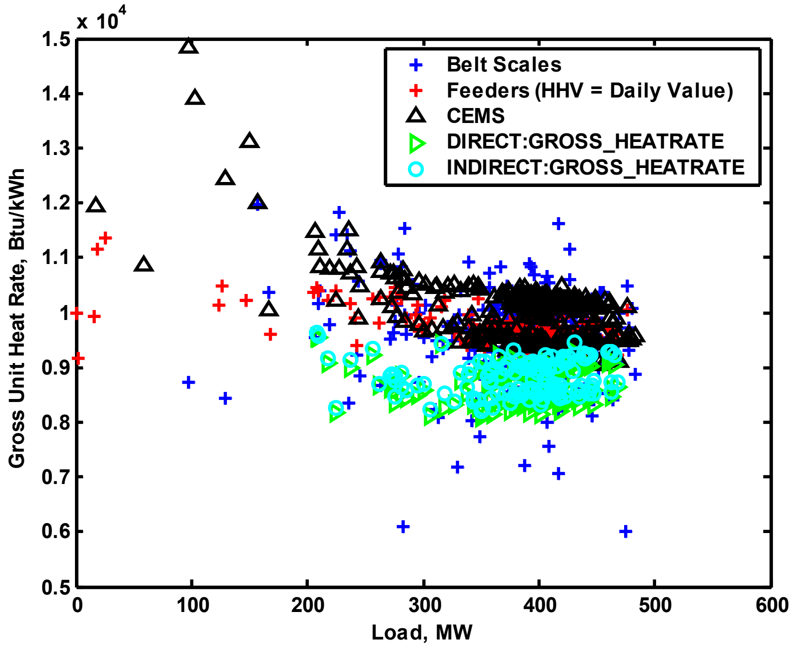


Figure 7-29 Daily Average Heat Rates vs. Average Load for 2002

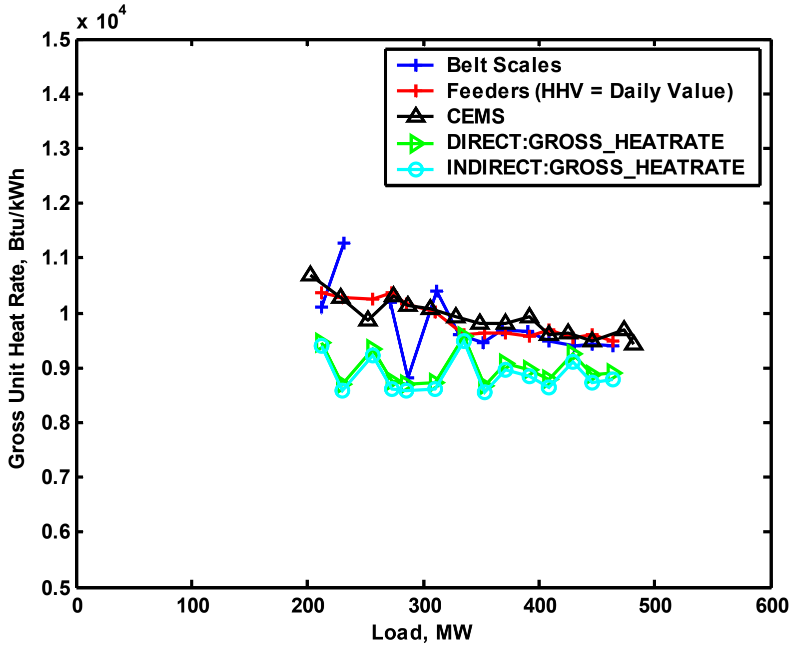


Figure 7-30 Mean Daily Average Heat Rates vs. Average Load for 2002

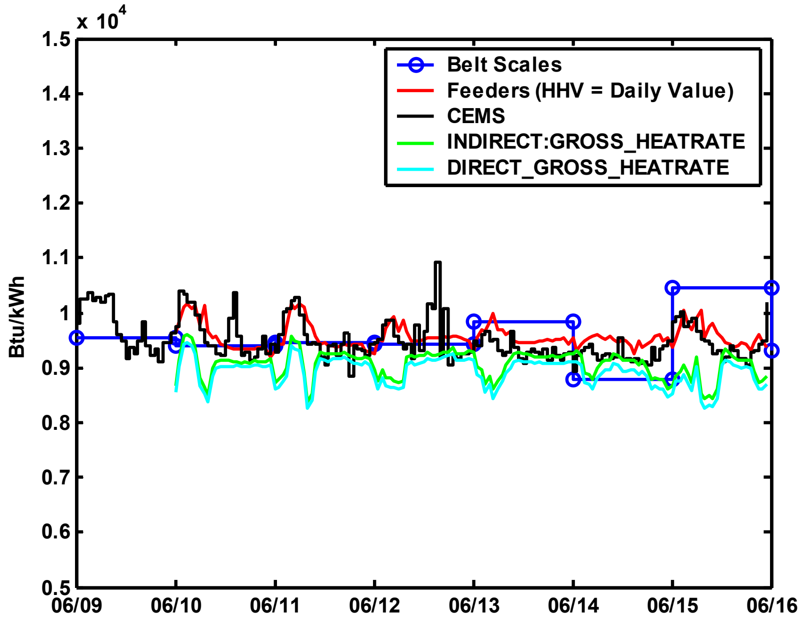


Figure 7-31 Hourly Average Heat Rates for the Week of June 9, 2002

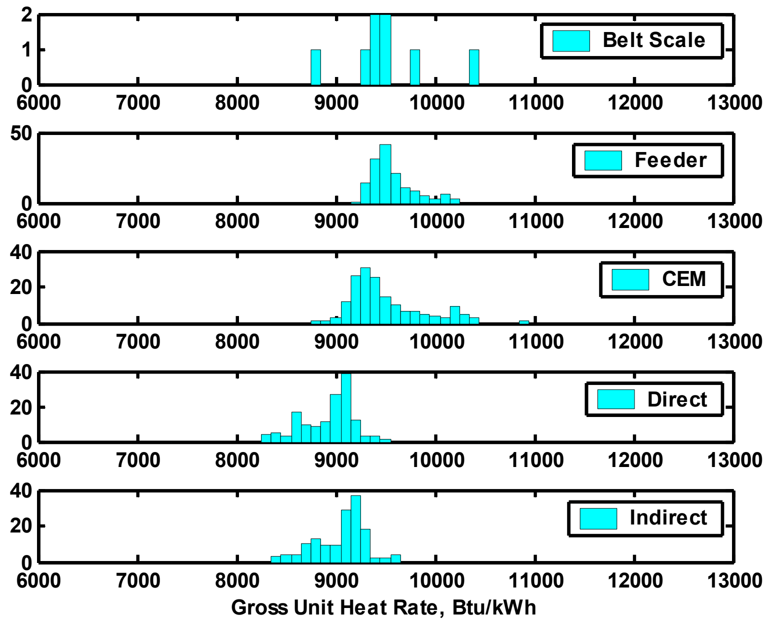


Figure 7-32 Hourly Average Heat Rates for the Week of June 9, 2002

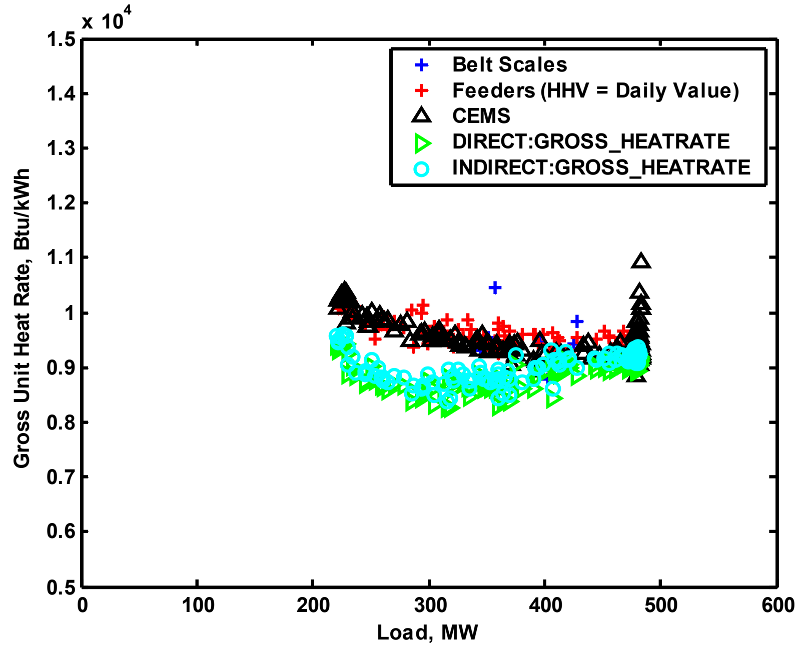


Figure 7-33 Hourly Average Heat Rates vs. Load for the Week of June 9, 2002

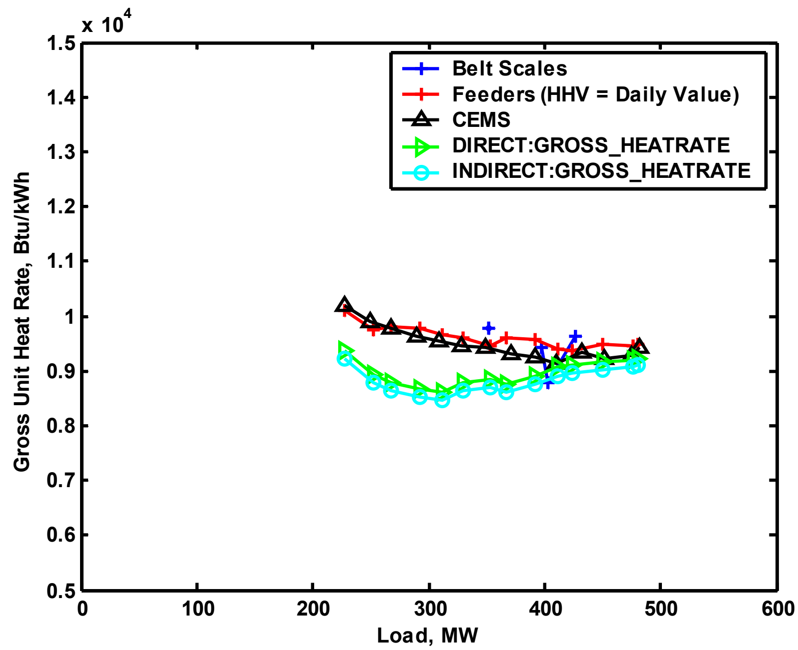


Figure 7-34 Mean Hourly Average Heat Rates vs. Load for the Week of June 9, 2002

Coal Flow

The CEP package calculates coal flow as part of the indirect method. Hammond 4 has a direct coal flow measurement (Stock gravimetric feeders) and this measurement, along with the belt scale measurement, may be compared to the CEM calculation to assess the robustness of the indirect method. A comparison of the daily coal burns is shown in Figure 7-35 though Figure 7-36. As shown, there was considerable day-to-day variation in coal burn resultant from changing generation demands on the unit. The belt scales and gravimetric feeders compared favorably while the indirect method exhibited a slight negative bias compared to the other measurements in Table 7-9.

Table 7-9 Coal Burn / 2002 Daily Data / All Loads > 200 MW (tons/day)

	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Belt	1611	3189	3549	3461	3813	4660
Feeders	2084	3266	3542	3482	3783	4291
Indirect	1748	2979	3317	3232	3556	5642

A comparison of the hourly coal flows for 2002 is shown in Table 7-10. For a shorter interval, hourly coal flow measurements are provided in Figure 7-38 through Figure 7-39 for the week of June 9, 2002. As shown, the indirect method tracked the feeders very closely, particularly at higher loads.

Table 7-10 Coal Flow / 2002 Hourly Data / All Loads > 200 MW (lb/hr)

	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Feeders	161600	235500	303300	290300	353300	389000
Indirect	127000	200000	274000	265100	330700	387100

Coal Higher Heating Value

Figures providing the coal higher heating value calculated by the indirect method are shown in Figure 7-40 and Figure 7-41. As shown, the calculated value exhibited little variation as compared to daily grab samples. The mean values were 12,662 Btu/lb and 12,638 Btu/lb for the grab samples and indirect method, respectively. In early versions of the CEP program, there was a positive correlation between calculated coal HHV and load. This apparently has been corrected (Figure 7-42).

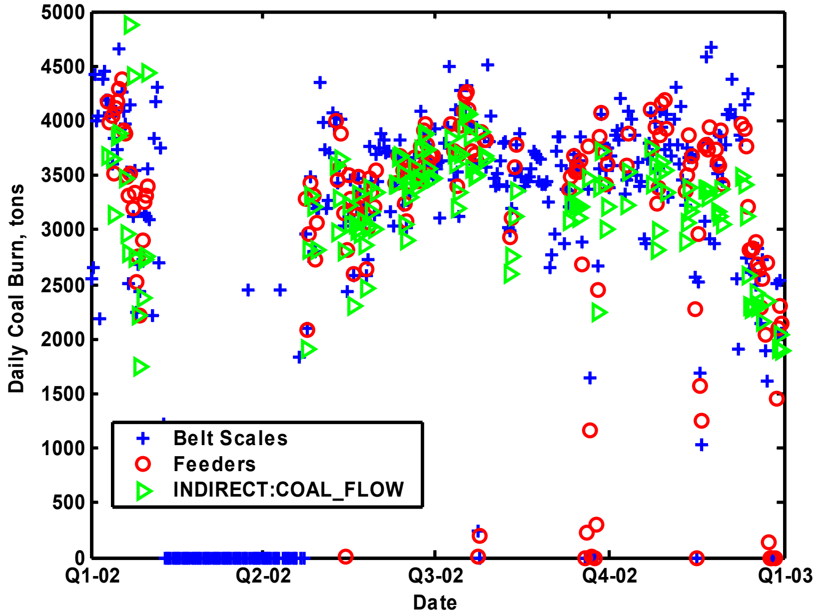


Figure 7-35 Daily Coal Burns by Day for 2002

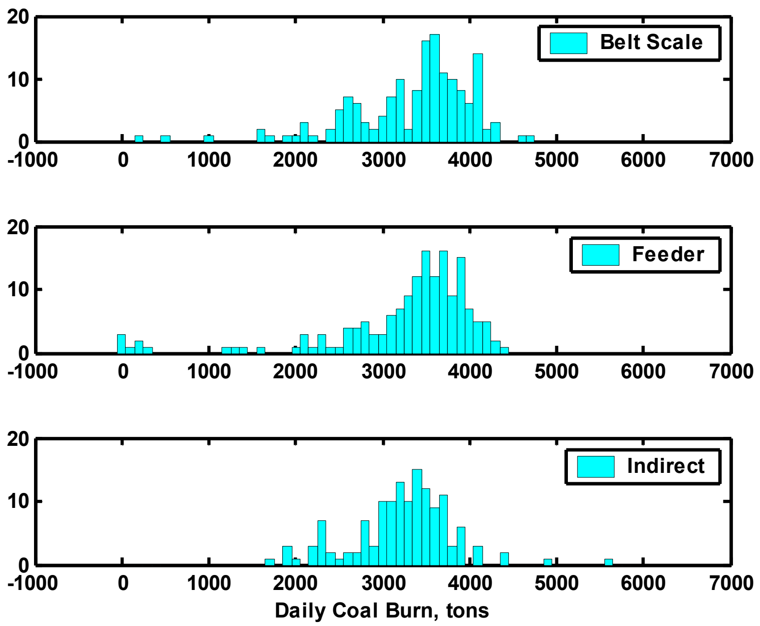


Figure 7-36 Daily Coal Burns for 2002

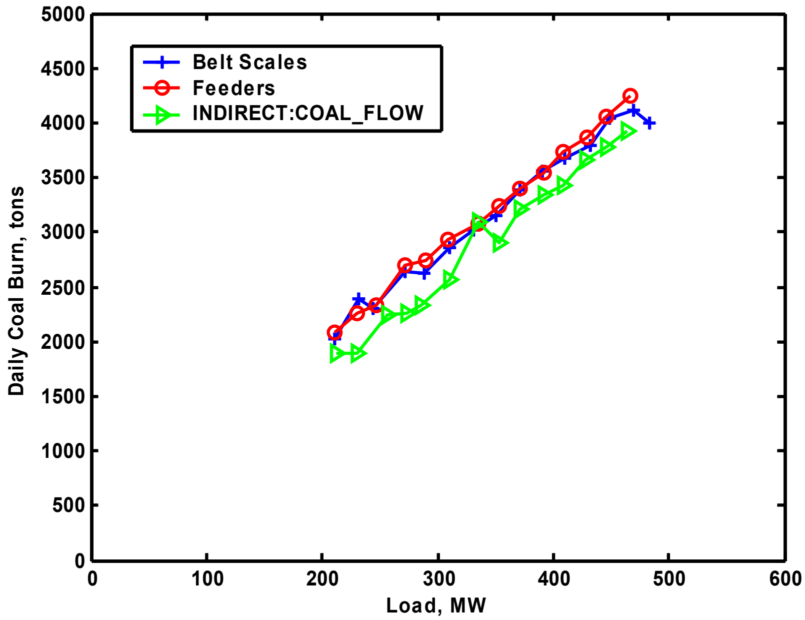


Figure 7-37 Daily Coal Burns by Load for 2002

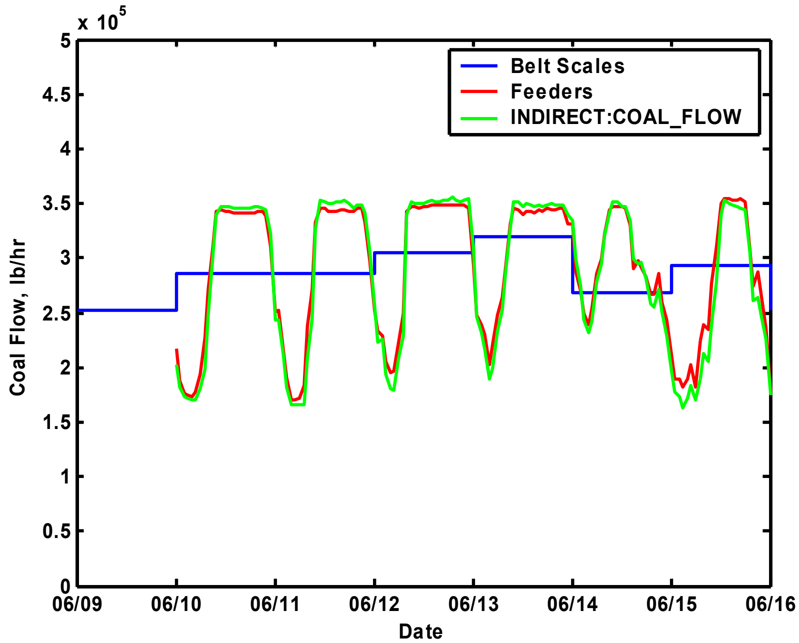


Figure 7-38 Hourly Average Coal Flow for the Week of June 9, 2002

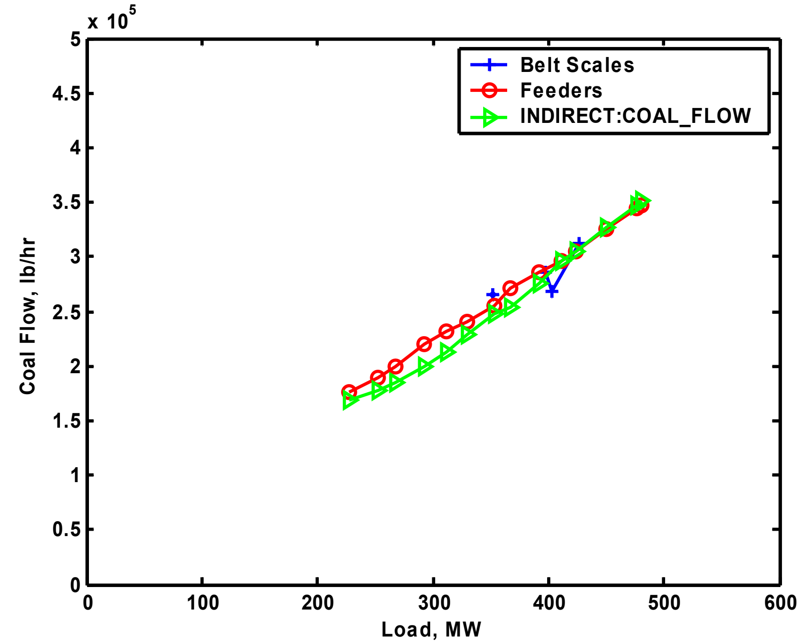


Figure 7-39 Hourly Average Coal Flow by Load for the Week of June 9, 2002

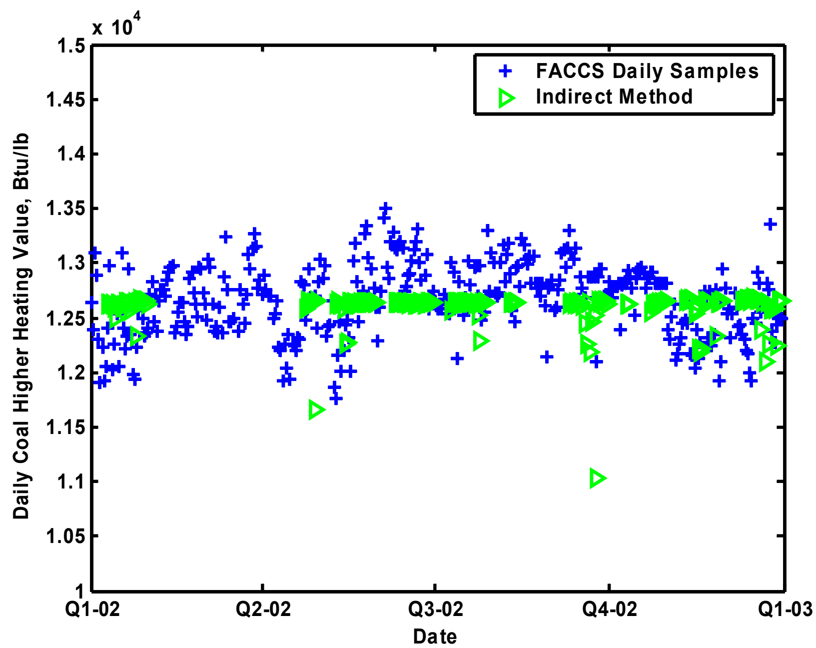


Figure 7-40 Daily Coal HHV by Day for 2002

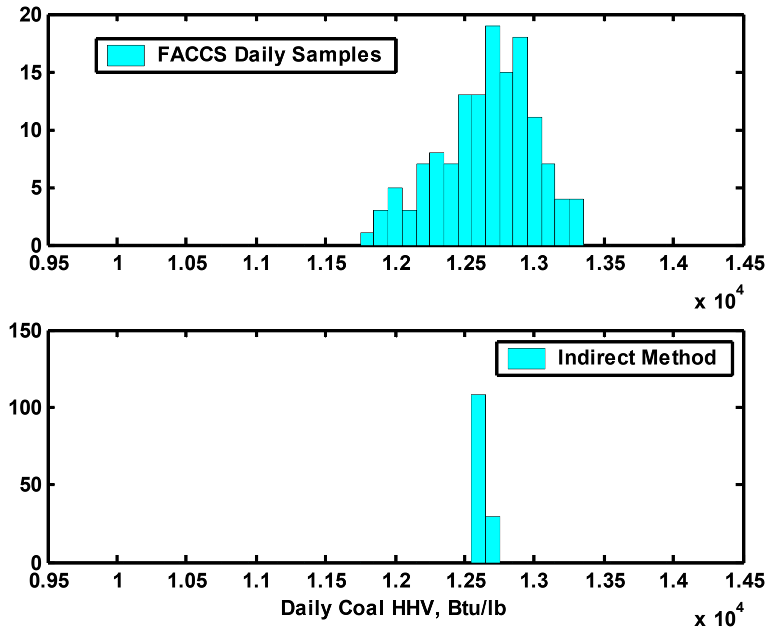


Figure 7-41 Daily Coal HHV Distribution for 2002

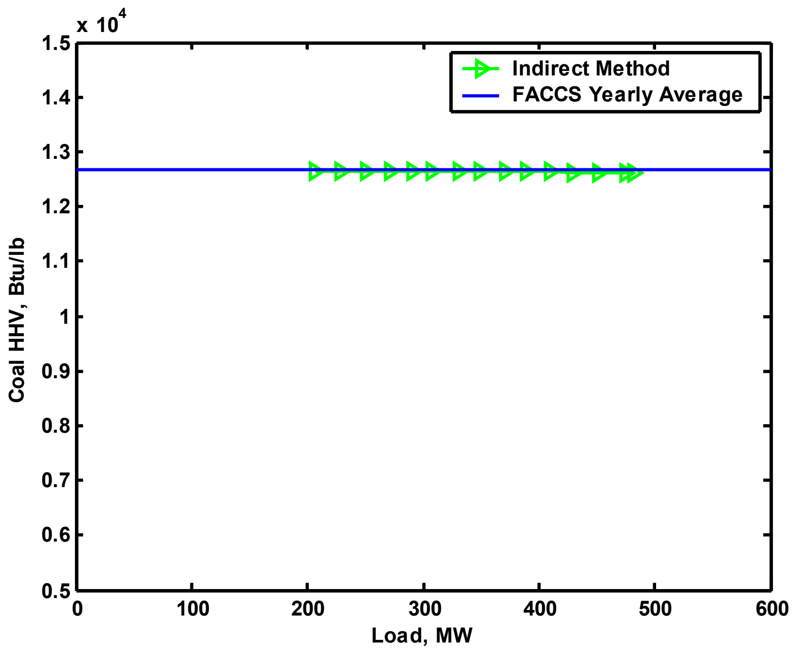


Figure 7-42 Hourly Coal HHV by Load for 2002

Summary

The CEP developed real-time heat rate package was installed as part of the unit optimization project at Hammond 4. The CEP provided a library containing the calculations and SCS wrote the interface to the RTDS. SCS contracted with CEP during December 1999 and a working version of the package was transmitted to SCS during July 2000. The software was subsequently revised with the final version being transmitted to SCS during March 2001. This package has performed reasonably well at the site with relatively little maintenance support by SCS and appears to be providing potentially valuable information. The following is based on an analysis of 2002 data.

Heat Rate – The package tracks unit performance but there is a substantial difference between the package results and other methods of determination. This difference is correlated with load and is greatest at mid- to lower-load categories. The difference is much greater starting in July 2002 than it was in earlier in the year. The reason for this shift is unknown.

Coal Flow – Considering the entire year, the package tracked daily coal burn reasonably well, although, as with heat rate, there was a bias between the calculated method and the direct measurements. This bias was not, however, correlated with load category as was the heat rate. Considering one week of operation (June 9 – June 15, 2002), the method tracked feeder coal flow with little deviation.

Coal Higher Heating Value – The indirect method of determining coal higher heating value showed very little variation and did not track daily changes as observed in the daily grab samples.

8
BOILER OPTIMIZATION PACKAGE

Overview

GNOCIS is a real-time, closed-loop system for performing boiler optimization. GNOCIS was first installed at Hammond 4 in 1996 as part of a prior phase of this project and was upgraded as part of this current project. A major improvement was the development and incorporation of on-line model error correction. Previous testing of GNOCIS at this site shows substantial benefits may be obtained by its application. The GNOCIS design, installation, and testing are discussed in this section.

GNOCIS

GNOCIS (Generic NOx Control Intelligent System) is an enhancement to digital control systems (DCS) targeted at improving utility boiler efficiency and reducing emissions [SCS97b]. GNOCIS is designed to operate on units burning gas, oil, or coal and is available for all combustion firing geometries. GNOCIS development was funded by a consortium consisting of the EPRI, Powergen, Southern Company, URS, U.K. Department of Trade and Industry, and U.S. Department of Energy.

GNOCIS utilizes a neural-network model of the combustion characteristics of the boiler that reflects both short-term and longer-term trends in boiler characteristics. A constrained-nonlinear optimizing procedure is applied to identify the best set points for the plant. These recommended set points can be implemented automatically without operator intervention (closed-loop), or, at the plant's discretion, conveyed to the plant operators for implementation (open-loop). The software is designed for continuous on-line use. The major elements of GNOCIS are shown in Figure 8-1.

The recommendations provided by GNOCIS, whether open- or closed-loop, are supervisory in nature and are ideally implemented via the DCS. As shown in Figure 8-2, GNOCIS utilizes process data collected from the DCS. Once determined, the recommendations are provided to the operator through the DCS or other displays. The operator can then make the final determination on whether these recommendations should be implemented. Alternatively, the recommendations are automatically implemented via the DCS.

Combustion optimization difficulty at Hammond has increased dramatically since the installation of the low NOx burners and advanced overfire air system. This added difficulty is a result of the increase in the number of adjustments and sensitivity of these burners to operating conditions (Table 8-1). Using this list as a starting point, GNOCIS was designed to make use of the variables shown in Table 8-2. The control variables in the first tier have been implemented and if successful, additional variables from the subsequent tiers will be considered if their inclusion improves the performance of the system significantly. Software hooks were designed into the DCS to facilitate the incorporation of these signals into the control logic.

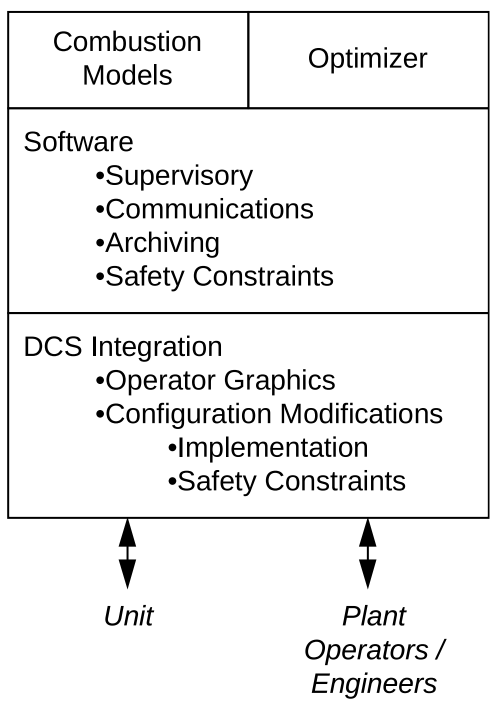


Figure 8-1 Major Elements of GNOCIS

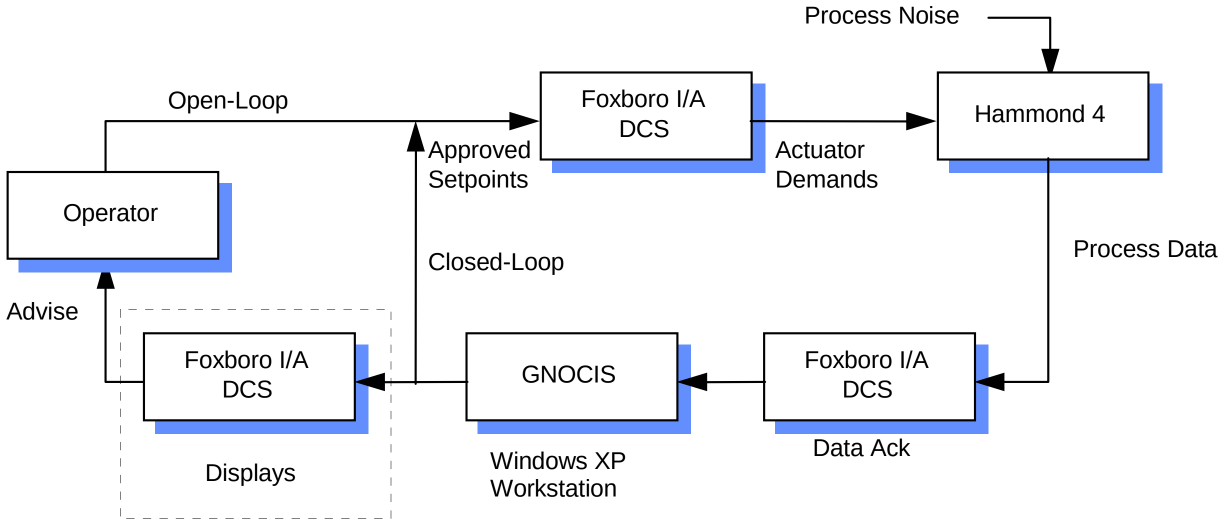


Figure 8-2 Hammond 4 GNOCIS Implementation

Sample operator graphics for GNOCIS are shown in Figure 8-3. Typically, the DCS operator displays are the principal interface to GNOCIS. These displays must (1) clearly convey to the operator the recommendations and predicted benefits and (2) allow the operator flexibility in setting constraints. As shown, the operator is presented with the current operating conditions and two sets of recommendations and predictions. One set corresponds to the current mills-in-service operating condition. If accepted, the operator can either implement the recommendations by individually setting the manipulated parameters to the targets or have the DCS automatically implement the recommendations (*Implement Recommendations*). When clamped, the operating parameter is assumed clamped to the current operating condition, and the optimization is performed with the remaining parameters. The operator can also remove or add parameters from the optimization by using this screen (*Clamped / Free).*

Since in many instances the mill selection can affect performance and emissions, it is important to provide recommendations concerning the mills in service. However, due to many externalities not measurable by the DCS or best judged by the operator, the mill configuration cannot be achieved or is not desirable. As a compromise, another set of recommendations are provided as to the optimum mills-in-service and the performance/emissions benefits. Given the predicted improvement and the current state of the plant, the operator can decide whether it is of overall advantage to change the mills in service. Closed-loop mode, if implemented, is obtained by selecting the *Close Loop* button from this screen.

Table 8-1 Combustion Tuning Control Points at Hammond 4

Pre-LNB+AOFA Retrofit	Post-LNB+AOFA Retrofit
Burners	Burners
Sleeve registers (24)	Sleeve registers (24)
Secondary air	Tip Positions (24)
Windbox balancing dampers	Inner registers (24)
Mill Biasing	Outer registers (24)
	Advanced overfire air
	Can-in-can dampers (8)
	Flow control dampers (4)
	Secondary air
	Windbox balancing dampers
	Boundary air
	Mill Biasing

Table 8-2 GNOCIS Control Points

Parameter of Interest	Controlled Parameter	Advisory Mode Open-Loop	Supervisory Mode Close-Loop
First Tier			
Overall Furnace Air / Fuel Ratio	Excess O_2 Bias	Y	Y
Overall Furnace Staging	AOFA Flow (4)	Y	Y
AOFA Distribution	AOFA Flow (4)	Y	Y
Mill Biasing	Mill Coal Flow (6)	Y	Y
Mills-in-Service	Mill Coal Flow (6)	Y	Advise
Second Tier			
AOFA Distribution	AOFA Can Dampers (8)	Y	Y
Furnace Secondary Air Distribution	Burner Dampers by Banks (8)	Y	Y
Third Tier			
Furnace Secondary Air Distribution	Burner Dampers (24)	Y	Y

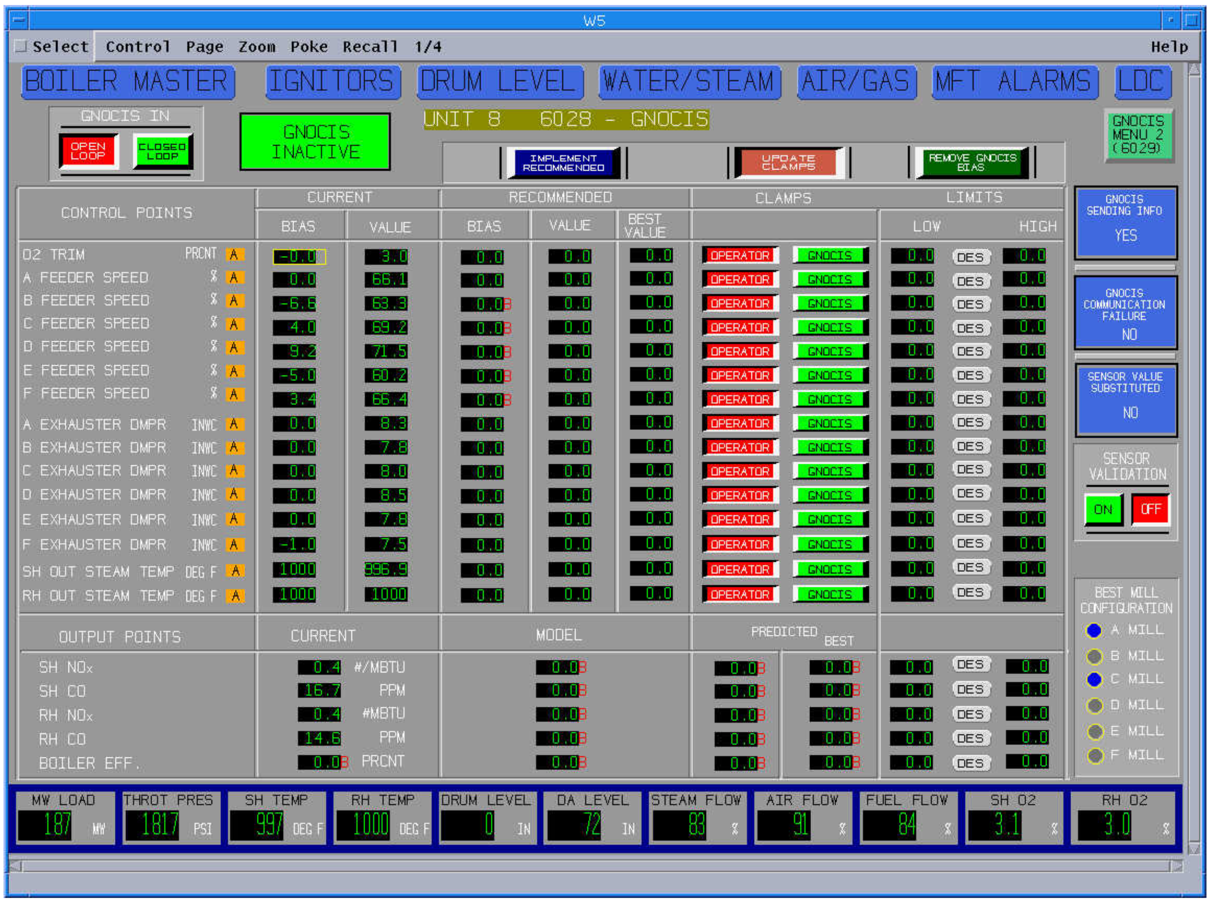


Figure 8-3 GNOCIS Recommendation Screen

Modifications to GNOCIS

Several modifications were made to GNOCIS as part of this project. These modifications consisted of:

- Interface to the RTDS rather than a direct connection to DCS
- Boiler model upgrades
- On-line error correction

These modifications are discussed in the following sections.

On-Line Error Correction

Prior to 2002, for all GNOCIS installations, the neural network models were built using historical data only.[1] Starting with raw, unfiltered process data collected through the distributed control system, the model developer visually examines some subset of the collected data to purge questionable data. This filtered data is then used to develop the neural network model(s). This static model is then placed into service and, dependent on observed performance of the model, periodic retraining of the model is performed using previously archived data.

Although this manual, offline retraining has been sufficient for many GNOCIS installations, in some cases problems have occurred in which, after some period of time, the GNOCIS models no longer sufficiently reflects the actual process. This degradation in model performance may be due to:

- Inadequacies in the original model due to such factors as:
  - Training data does not provide sufficient coverage of the unit's potential operating envelope
  - Model design (inputs, outputs, etc) not robust
- Underling plant changes due to such factors as:
  - Fuel changes
  - Equipment degradation or modification
  - Weather

An example of the error that can occur is shown in Figure 8-4. As can be seen from this chart,

[1] *GNOCIS Plus*, introduced in 2002, incorporates on-line retraining of models. *GNOCIS* and *GNOCIS Plus* are both being offered and the product installed depends on application and end-user preference.

the NOx model prediction accuracy deteriorated rapidly beginning near May 10. A histogram of the NOx prediction error is shown in Figure 8-5. As shown, the distribution is bimodal with the cluster around zero largely representing the data before May 10 and the cluster around 0.06 lb/Mbtu reflecting data collected before that date. Similar data for October 2001 is shown in Figure 8-6 and Figure 8-7. The spikes in the data in Figure 8-6 are from the daily calibration cycle of the CEM. Data from one day, October 15, from this period is shown in Figure 8-8 and Figure 8-9. The model prediction error is usually highly correlated with recent past values of the error (Figure 8-10).

The purpose of the on-line error correction package was to develop a software plug-in to GNOCIS to implement on-line, continuous model adaptation for instances where it may be beneficial, particularly for existing GNOCIS installations, with only minor modifications to the GNOCIS software and minor disruptions to operations.

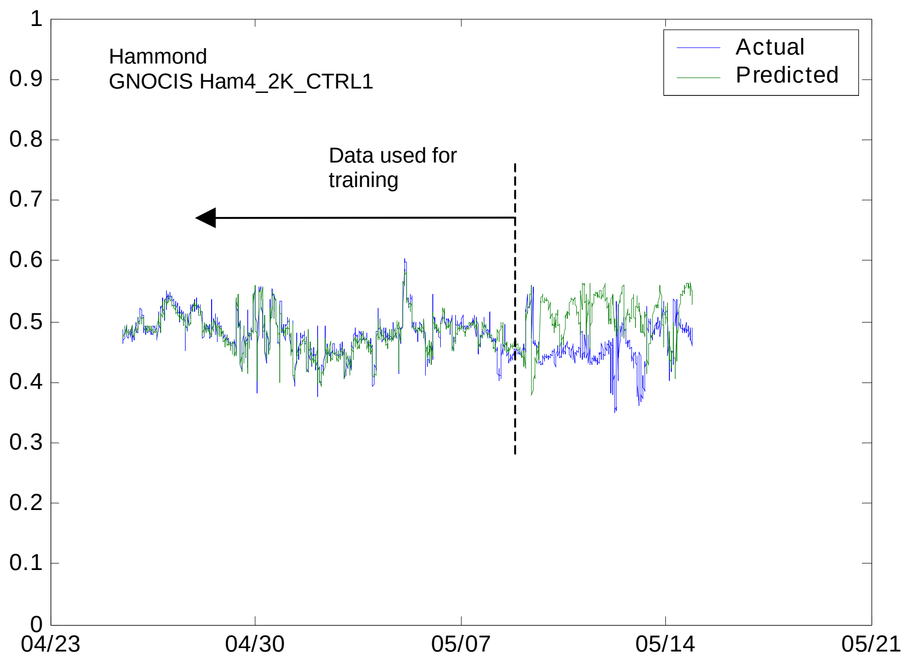


Figure 8-4 NOx Predicted vs. Actual – April – May 2000

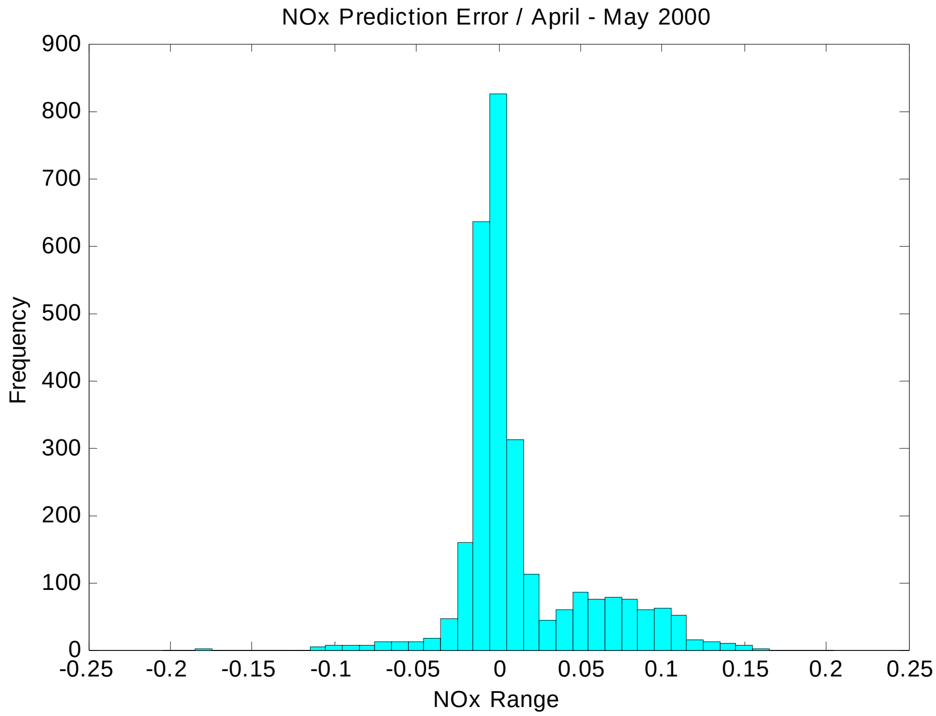


Figure 8-5 NOx Predicted vs. Actual – April – May 2000 (Histogram)

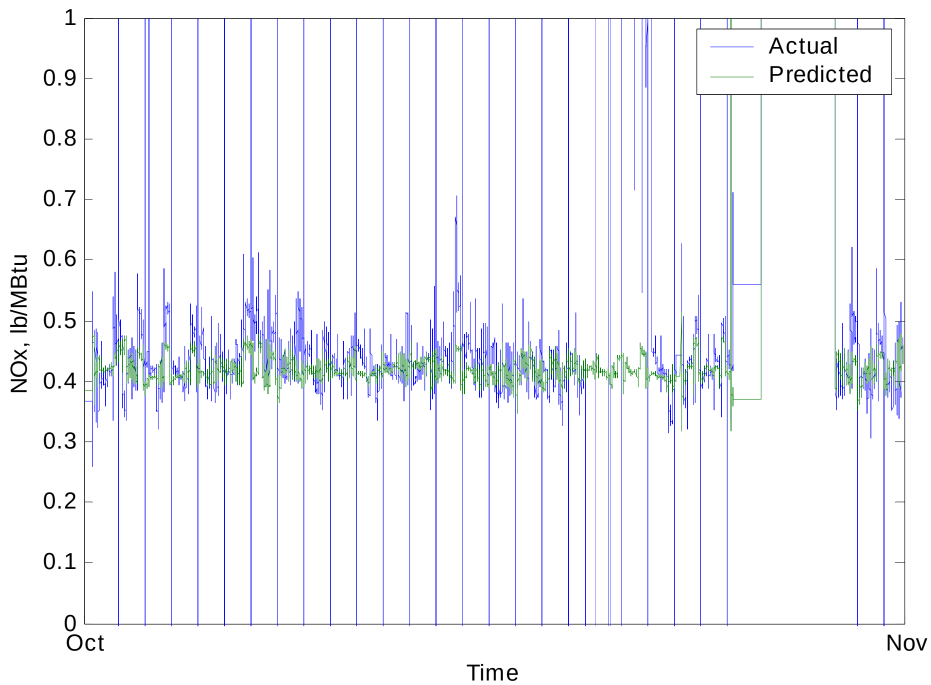


Figure 8-6 NOx Predicted vs. Actual – October 2001

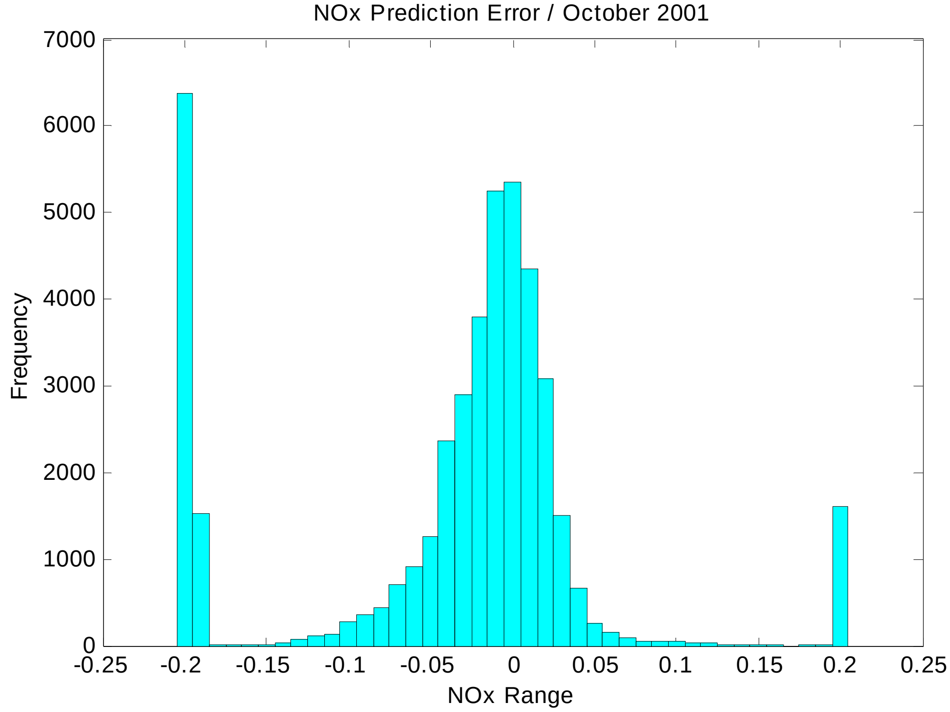


Figure 8-7 NOx Predicted vs. Actual – October 2001 (Histogram)

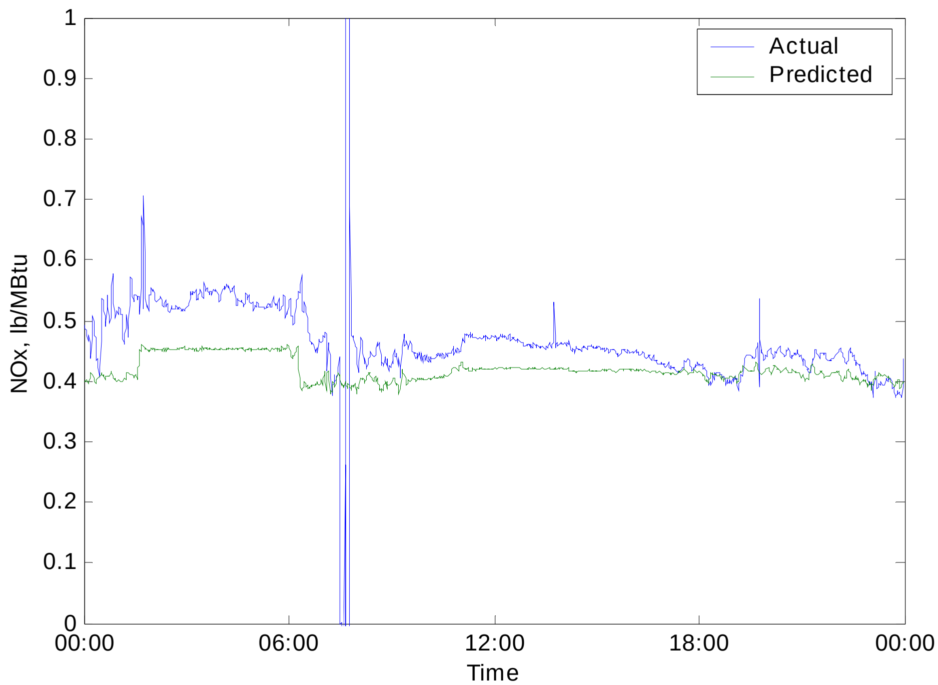


Figure 8-8 NOx Predicted vs. Actual – October 15, 2001

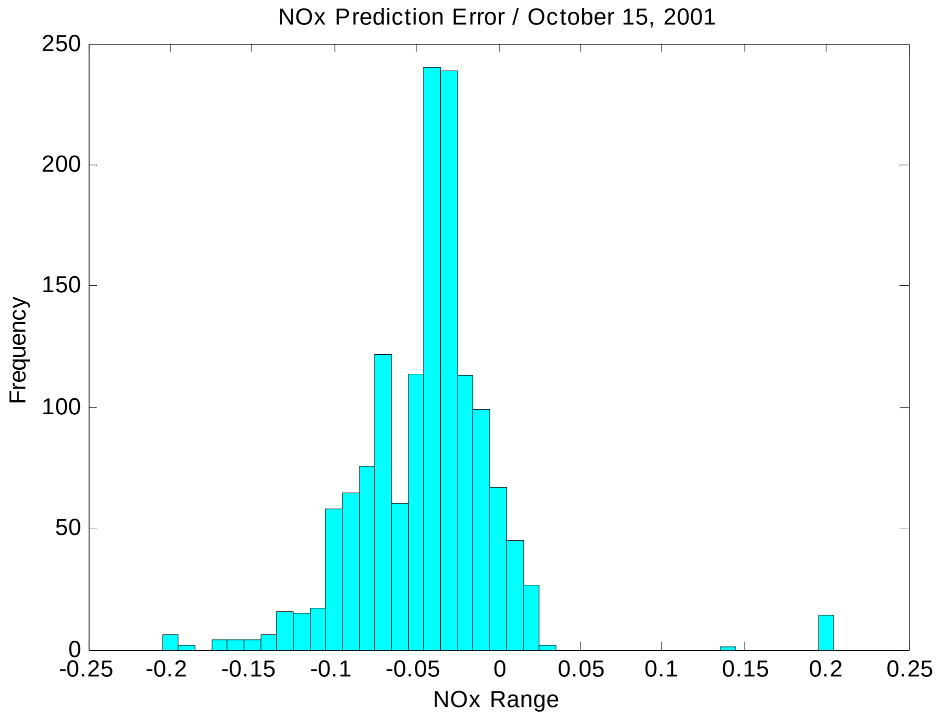


Figure 8-9 NOx Predicted vs. Actual – October 15, 2001 (Histogram)

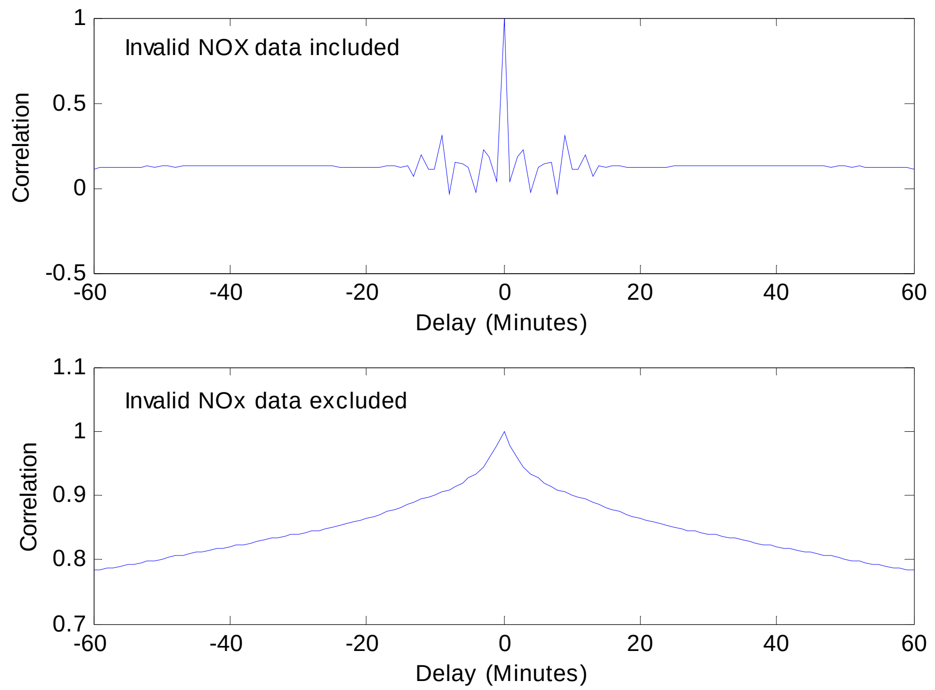


Figure 8-10 Auto-Correlation of NOx Prediction Error

Implementation Overview

An overview of the GNOCIS component GNCTL is shown in Figure 8-11. GNCTL is the GNOCIS software component responsible for the online prediction and optimization of the process. During the initialization phase, the software gathers data from the data server (typically the DCS) and performs sensor validation on the inputs (including limit checking and more sophisticated methods). The data is also transformed to model variables actually used by the neural network models. Examples of transformed variables include average excess oxygen, total fuel flow, and boiler efficiency. The transformed variables are passed to the routine *Run_Model*. *Run_Model* can operate in either of two modes: straight prediction or control/optimization (Figure 8-12 and Figure 8-13).

In the prediction mode, the transformed variables are passed to the neural network engine to calculate predicted outputs. In the control/optimization mode, the transformed data along with limits and goals are passed to *Run_Model* which iterates on the neural network model, and determines the model inputs that will achieve these goals and limits optimally. For both the prediction and control/optimization modes, software hooks have been provided by Pavilion to set constant (for the given run model) biases to the model outputs prior to the calling of *Run_Model*.

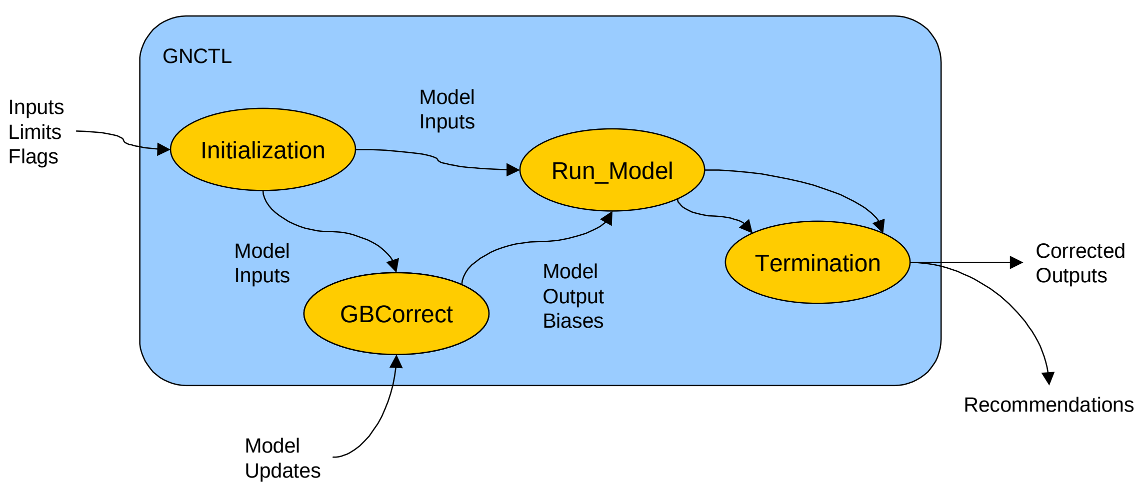


Figure 8-11 GNOCIS Overview

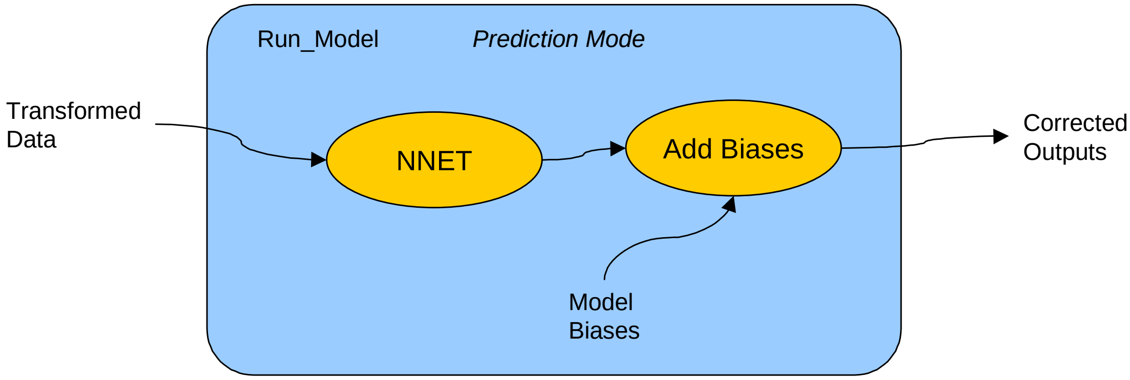


Figure 8-12 GNOCIS Prediction Mode

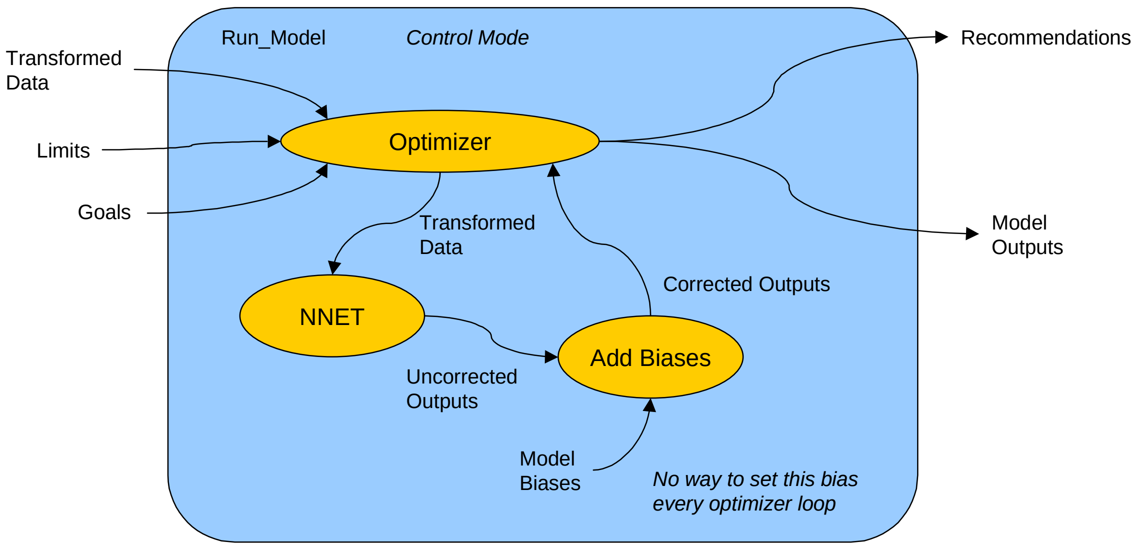


Figure 8-13 GNOCIS Control (Optimization) Mode

For the prediction mode, the use of a constant bias that is set before calling *Run_Model* is satisfactory — the bias can be updated and a function of the inputs to *Run_Model;* therefore the neural network can be corrected over its entire surface. For the control/optimization mode, the bias cannot be adjusted before each calling of the neural network model but only before the call to *Run_Model* and, therefore, the corrections are constant biases, shifting the entire surface up or down by the same amount. An example of how of this impacts a made-up function is shown in Figure 8-14. The impact of this limitation depends on several factors including:

- Variability of process and neural network model
- Control range allowed in the control/optimization mode

Unless hooks are found or are developed that allow inter-optimization modification of the biases, there will be limitations in the ability to correct the static combustion models.

The impact of this limitation for a specific NOx model and time period (October 2001) is shown in Figure 8-15. For this case, as the constraint limits are expanded from 0 to 100%, the standard deviation of the error of assuming a constant model bias increases to a maximum of approximately 0.012 lb/MBtu.

Schematics of how the model correction interfaces with GNCTL is shown in Figure 8-16 and Figure 8-17. GBCorrect, the error correction module, is called from within GNCTL to calculate the current biases. GBCorrect can use data passed to it from GNCTL and also data from the RTDS and other sources to develop the error models. A function schematic for GBCorrect is shown in Figure 8-18. GBCorrect can be called directly as a DLL or as a COM object. Multiple error correction models are supported and the configuration of which model types are used and parameters used by these models are configurable by initialization files (Figure 8-19 and Figure 8-20).

Within GNCTL, *Run_Model* is first called in the predict mode to determine the predicted model outputs without the addition of the biases. The model inputs, actual outputs, and predicted outputs are then passed to GBCorrect. GBCorrect then updates the error correction models using this data dependent on the model type.

A list of the model types developed is shown in Table 8-3. Model selection and configuration is generally performed by initialization files. Other models can be developed and added to the model library directory.

On-line, the user interfaces to the error correction through two panels (Figure 8-21 and Figure 8-22). Through these panels, the user may enable and disable error correction and monitor its operation.

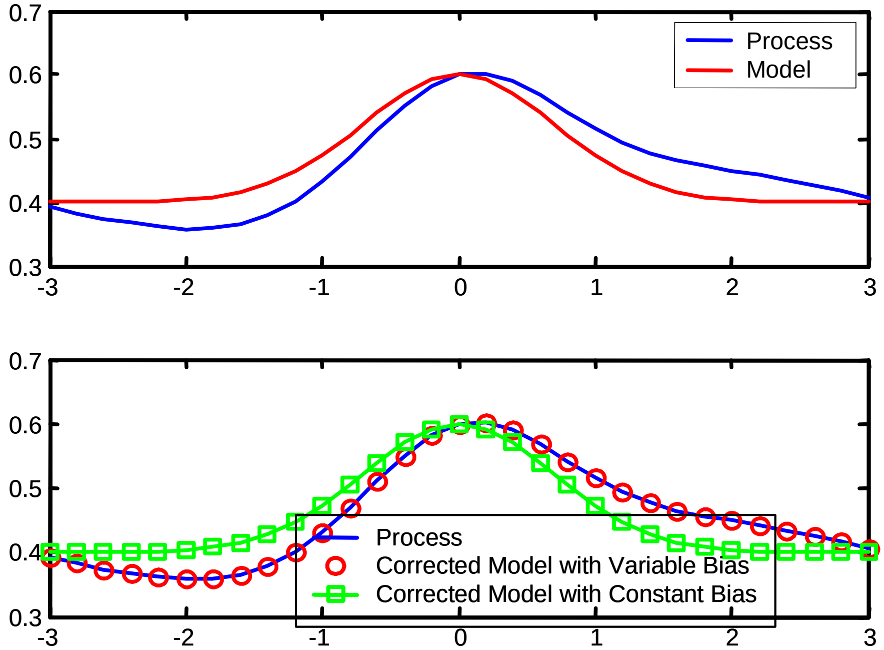


Figure 8-14 Constant vs. Variable Bias

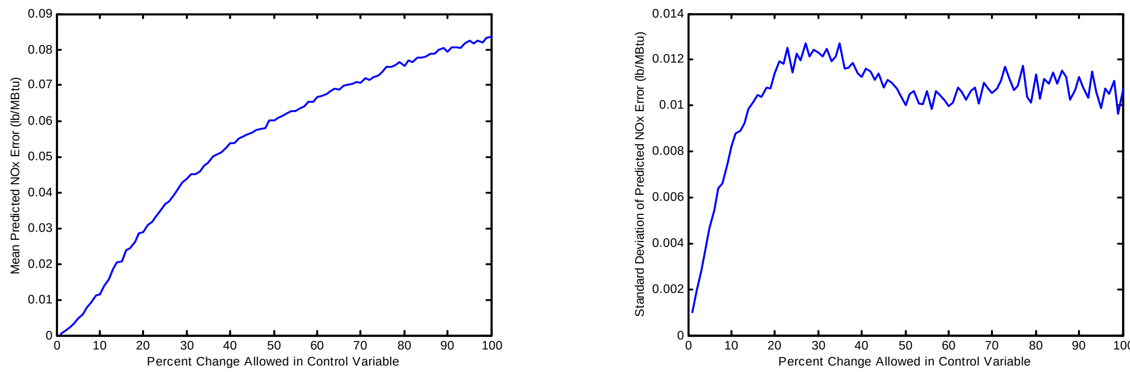


Figure 8-15 Predicted NOx Error

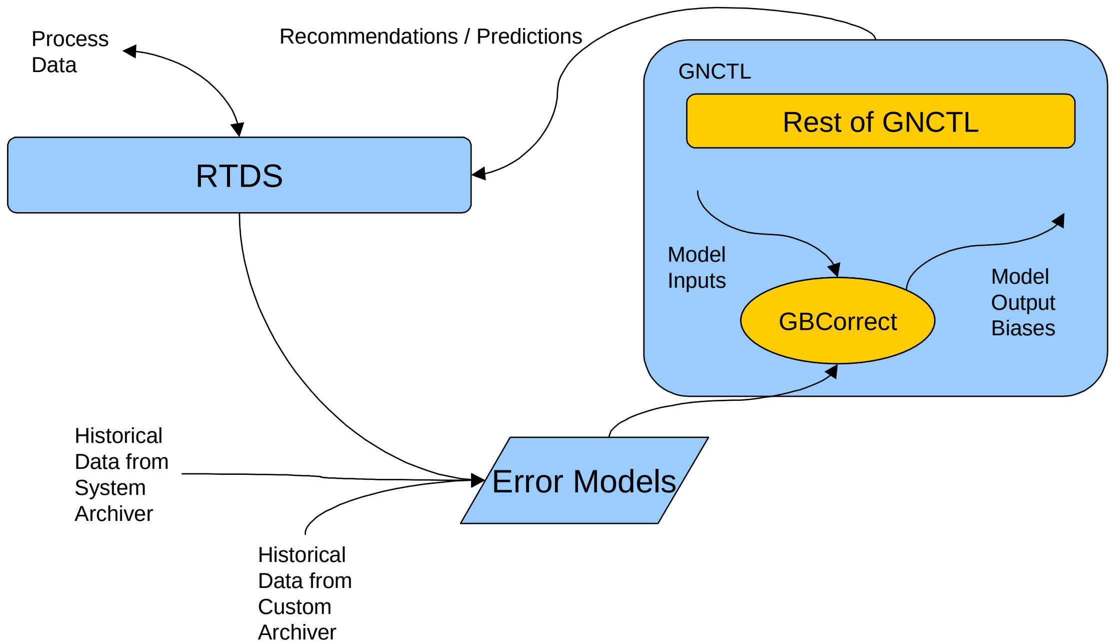


Figure 8-16 Overview of GBCorrect Error Correction Module

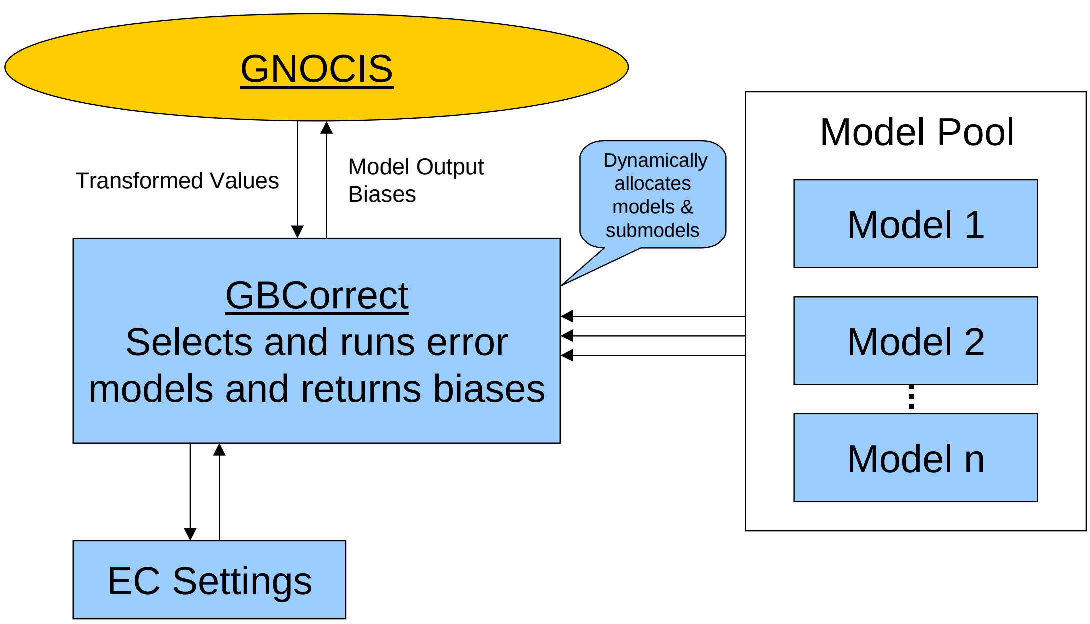


Figure 8-17 Overview of GBCorrect Error Correction Module

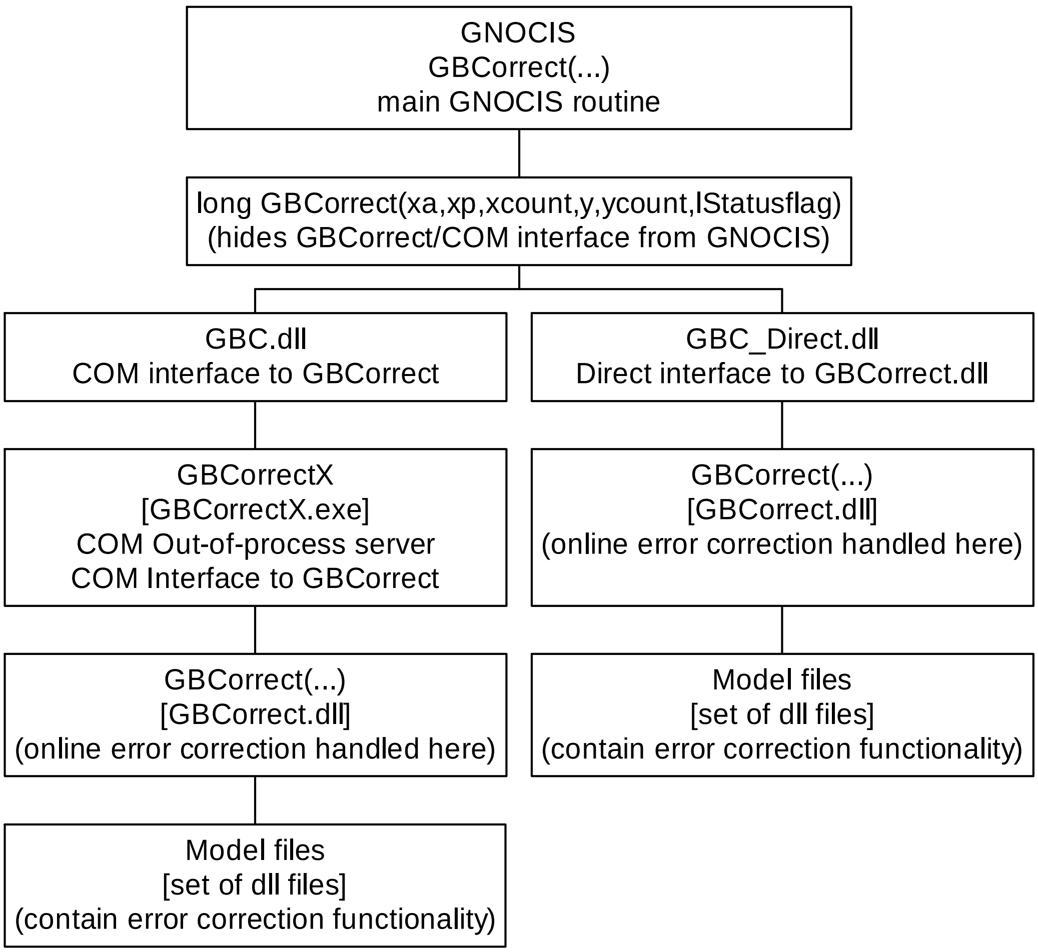


Figure 8-18 Functional GBCorrect Schematic

```
[NOx]                                                  --- Model Section Name ---
OutputName=NOX_LBMMBTU                                 --- Name of output in GNOCIS Model File ---
Enabled=1                                              --- Enable / Disable Bias ---
UseModelBias=1                                         --- Enable / Disable Model Bias ---
ManualBias=0.0                                         --- Manual Bias Level ---
LowBiasLimit=-0.1                                      --- Low Bias Clamp ---
HighBiasLimit=0.1                                      --- High Bias Clamp ---
ModelPath="ModelLibraryPath\tlm1"                      --- Model Type (dll file) ---
ModelData="%OLEC_DATA_DIR%/ModelTest.ini|NOXC2"        --- Model Data (usually, INI file|section) ---
```

Figure 8-19 Example GBCorrect INI File

```
[NOXC2]                                                --- Model Section Name ---
; top level model calling drbf2 with checking of the inputs
ModelType = "%OLEC_LIB_DIR%/tlm1"                      --- Model Type (dll file) ---
DefaultTagList = "%OLEC_DATA_DIR%/DefaultTagList.txt"  --- Default tag list if not already loaded ---
SubModel = "%OLEC_DATA_DIR%/ModelTest.ini|DRBF2"       --- Sub Model (usually, INI file|section) ---

Inputs = WMILLAC, ….., WMILLEC, WMILLFC, O2            --- List of inputs, corresponds to sections ---
                                                       ---   Inputs must be in this file ---
References = NOX_Actual                                --- List of references, corresponds to sections ---
                                                       ---   References must be in this file ---
                                                       ---   Must be valid for model to update ---

[DRBF2]                                                --- Model Section Name ---
ModelType = "%OLEC_LIB_DIR%/DRBF2"                     --- Model Type (dll file) ---
Sigma0 = 8.0                                           --- Model specific parameters ---
epsmax = 2.0
epsmin = 0.02
emin = 0.05
gf = 0.2
UpdateMode = 3
MaxCenters = 50
.
; The following are some of the input blocks
[O2]                                                   --- Input block ---
Source = AVG_O2                                        --- Source tag name ---
ValidLowerBound = 2.0                                  --- Valid lower bound ---
ValidUpperBound = 6.0                                  --- Valid upper bound ---
ScaleUpper = 0.0                                       --- Scale input between these two values ---
ScaleLower = 10.0                                      --- Scale input between these two values ---

[NOX_Actual]
Source = NOX_LBMMBTU
ValidLowerBound = 0.2
ValidUpperBound = 0.6
ScaleUpper = 0.0
ScaleLower = 1.0
```

Figure 8-20 Example Model INI File

Table 8-3 Current Model Types

Model Type	Description
Batcher1	During model updates, collects the inputs over a specified period to perform batch training. Can serve as the front end of the other models, though it is relevant on some more than others.
BiasAdjust	Returns the last error as the current error. Since the error tends to be highly auto-correlated for small delays, this may be useful, but the other filter models are probably better even for this case.
ConstantModel	Model that returns a constant, primarily used for testing.
DRBF1	Model based on an adaptive radial basis function neural network.
DRBF2	Serves as a front end to DRBF1 allowing setting of configuration settings within the initialization file.
Filter0	First order filter of the form: $bias = k*bias+(1-k)*bias_{new}$
Filter1	Filter type model using both past biases and inputs.
GenericModel	Model that serves as a front end for other models. Used for configuring models that don't read INI files.
MLP1	Adaptive Multiple-Layer Perceptron neural network.
NullModel	Model that returns zeros, primarily used for testing.
RunAvg	During model updates, collects the inputs (X and Y) over a specified period passing the average to a submodel.
RunAvgY	Average of the past N samples, where N may be set.
TLM1	Serves as the front end to other models, allowing setting of inputs by name and implementing error checking on these inputs.

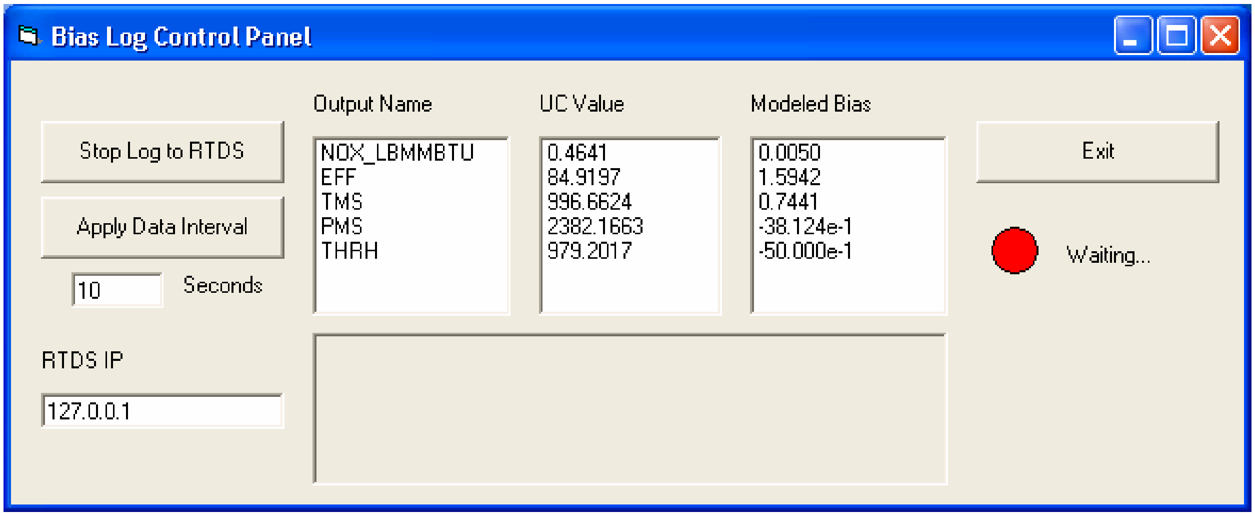


Figure 8-21 Bias Log Control Panel

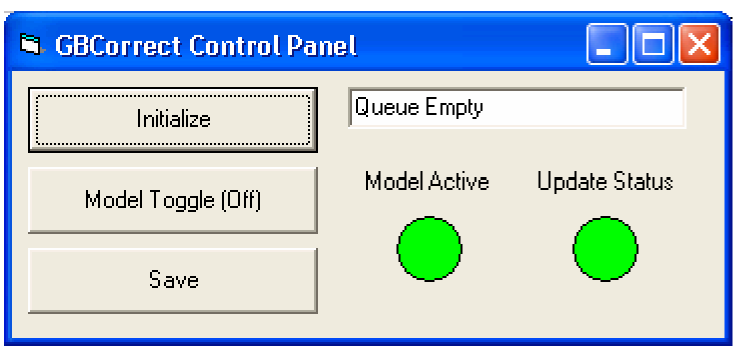


Figure 8-22 GBCorrect Control Panel

Performance of Corrected Models

The effectiveness of the on-line bias correction depends on the modeled process, the uncorrected model, the bias model, and the data set used to determine the effectiveness. Examples of the effectiveness are provided in the following paragraphs.

<u>Example 1</u>

In this example, the correction model was based on a radial-basis function neural network with a single input (excess oxygen). This structure may be applicable if there were errors in the unit's excess oxygen instrumentation or furnace backpass leakage varied over load or with time. The RBF network was updated every call (~1 minute intervals) to the error correction routines with no batching or averaging of the inputs. The results are shown in Figure 8-23 and Figure 8-24. Error correction was enabled during the entire period. In the abscissa range of 0 to ~2000, predicted (uncorrected) NOx tracked actual NOx but after that, the two diverged. With the error correction, the predicted tracked much more closely.

<u>Example 2</u>

In this example, the correction model was based on a radial-basis function neural network with individual feeder flows as inputs. The RBF network was updated every call (~1 minute intervals) to the error correction routines with no batching or averaging of the inputs. The results are shown in Figure 8-25 and Figure 8-26. Error correction was enabled during the entire period. As with Example 2, in the range of 0 to ~2000, predicted (uncorrected) NOx tracked actual NOx but after that, the two diverged. With the error correction, the predicted tracked much more closely.

<u>Example 3</u>

In this example, the correction model was based on an adaptive multi-layer percepton neural network with excess oxygen as the input. The network was updated every call (~1 minute intervals) to the error correction routines with no batching or averaging of the inputs. The results are shown in Figure 8-27 and Figure 8-28. Performance was similar to that shown in the earlier examples.

Table 8-4 On-Line Error Correction – Example 1

Model Type:	DRBF1 (NOXC1)
Inputs:	O2
Batch Period:	None
Averaging Period:	None
Period:	April – May 2000 (ModelTest_02)

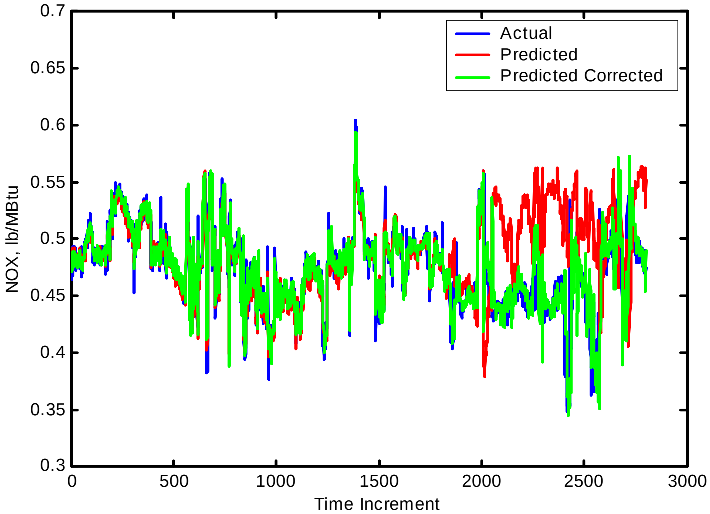


Figure 8-23 Effectiveness of Error Correction (Example 1)

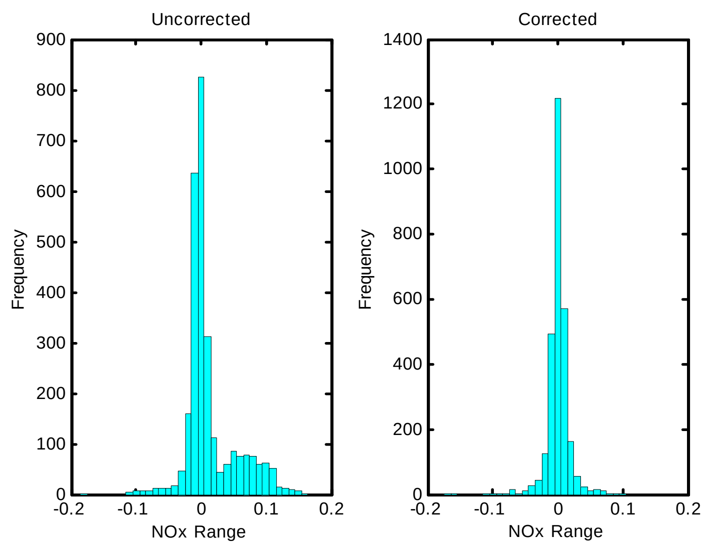


Figure 8-24 Effectiveness of Error Correction – Error Histogram (Example 1)

Table 8-5 On-Line Error Correction – Example 2

Model Type:	DRBF1 (NOXC2)
Inputs:	WMILLAC, WMILLBC, WMILLCC, WMILLDC, WMILLEC, WMILLFC, O2
Batch Period:	None
Averaging Period:	None
Period:	April – May 2000 (ModelTest_02)

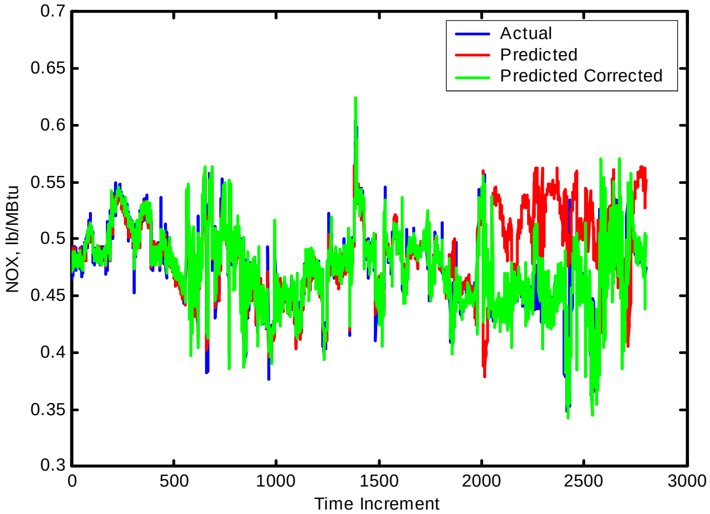


Figure 8-25 Effectiveness of Error Correction (Example 2)

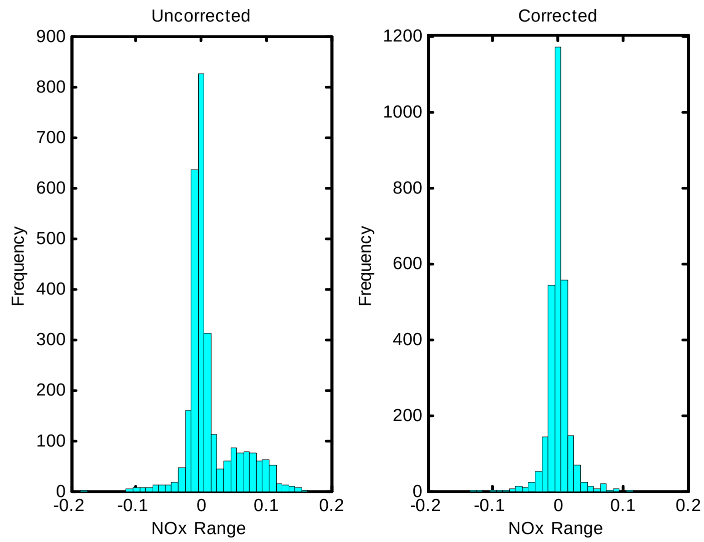


Figure 8-26 Effectiveness of Error Correction – Error Histogram (Example 2)

Table 8-6 On-Line Error Correction – Example 3

Model Type:	MLP1 (ZZNOXC1)
Inputs:	O2
Batch Period:	None
Averaging Period:	None
Period:	April – May 2000 (ModelTest_02)

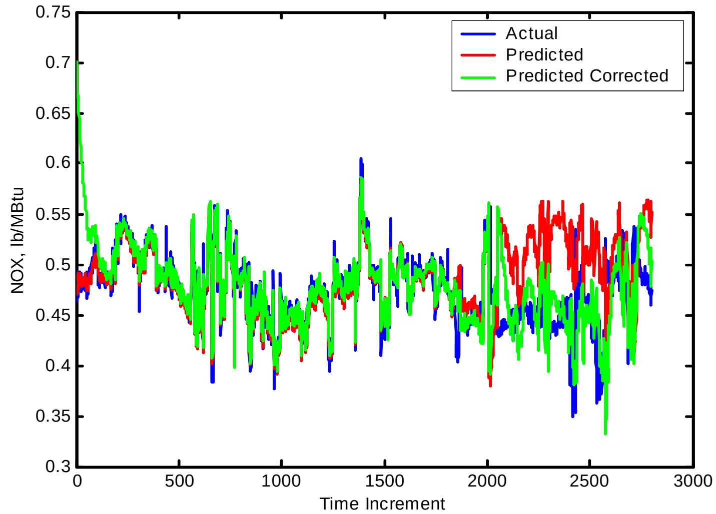


Figure 8-27 Effectiveness of Error Correction (Example 3)

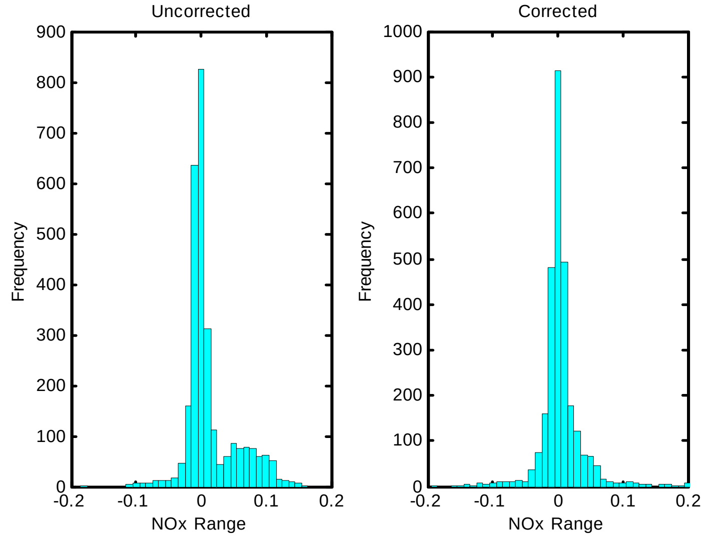


Figure 8-28 Effectiveness of Error Correction – Error Histogram (Example 3)

GNOCIS Interface Modifications

At the start of the project, it was anticipated that many processes would require access to real-time process data and the ability to share data. As installed in 1996, GNOCIS interfaced directly with the Foxboro DCS over a local area network (Figure 8-29). At 20 second intervals (adjustable by the user), *gnread* collects data from the DCS, and transmits this data, via sockets, to *gnctl*, the GNOCIS control application, and *gnarch*, the GNOCIS archiver. The data is then time averaged in each application to obtain one-minute averages (user definable) which is then used in the optimization and archived. The recommendations, from *gnctl*, are calculated and then transmitted (using sockets) to *gnwrite* for possible implementation in the DCS.

Although this configuration performed satisfactorily for the initial installation, there would be difficulties in applying this communication method successfully to the multiple, semi-independent processes envisioned for this project. To address this issue, a shared memory architecture was adopted (Figure 8-30). The RTDS (Real-Time Data Server) serves as the common repository for current process data and calculated results for GNOCIS and the other applications. This DCOM component may be accessed on the local or remote PC. The processes *dcs_wr*, *dcs_rd*, *gn_rd*, and *gn_wr* marshal the data between the two communication protocols. Using this method, you can revert to original configuration rapidly without code modifications.[1]

[1] During 2nd quarter 2003, the plant installed a plant operations information system (Aspen Info21Plus) that provides a commercial, robust, and flexible interface to process data collected from the DCS beyond that available through the RTDS. Migration from the RTDS to the Aspen system is being considered.

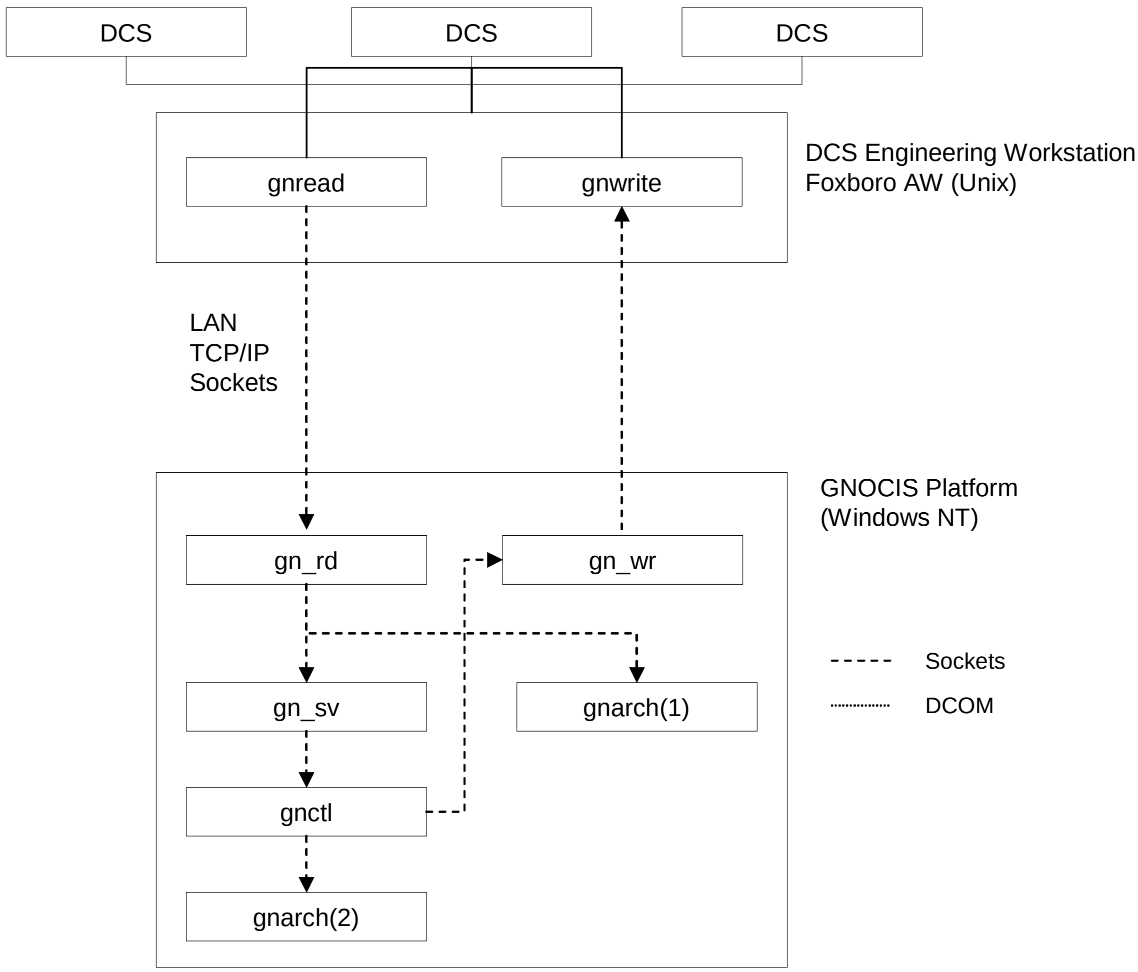


Figure 8-29 Original GNOCIS Interface to DCS

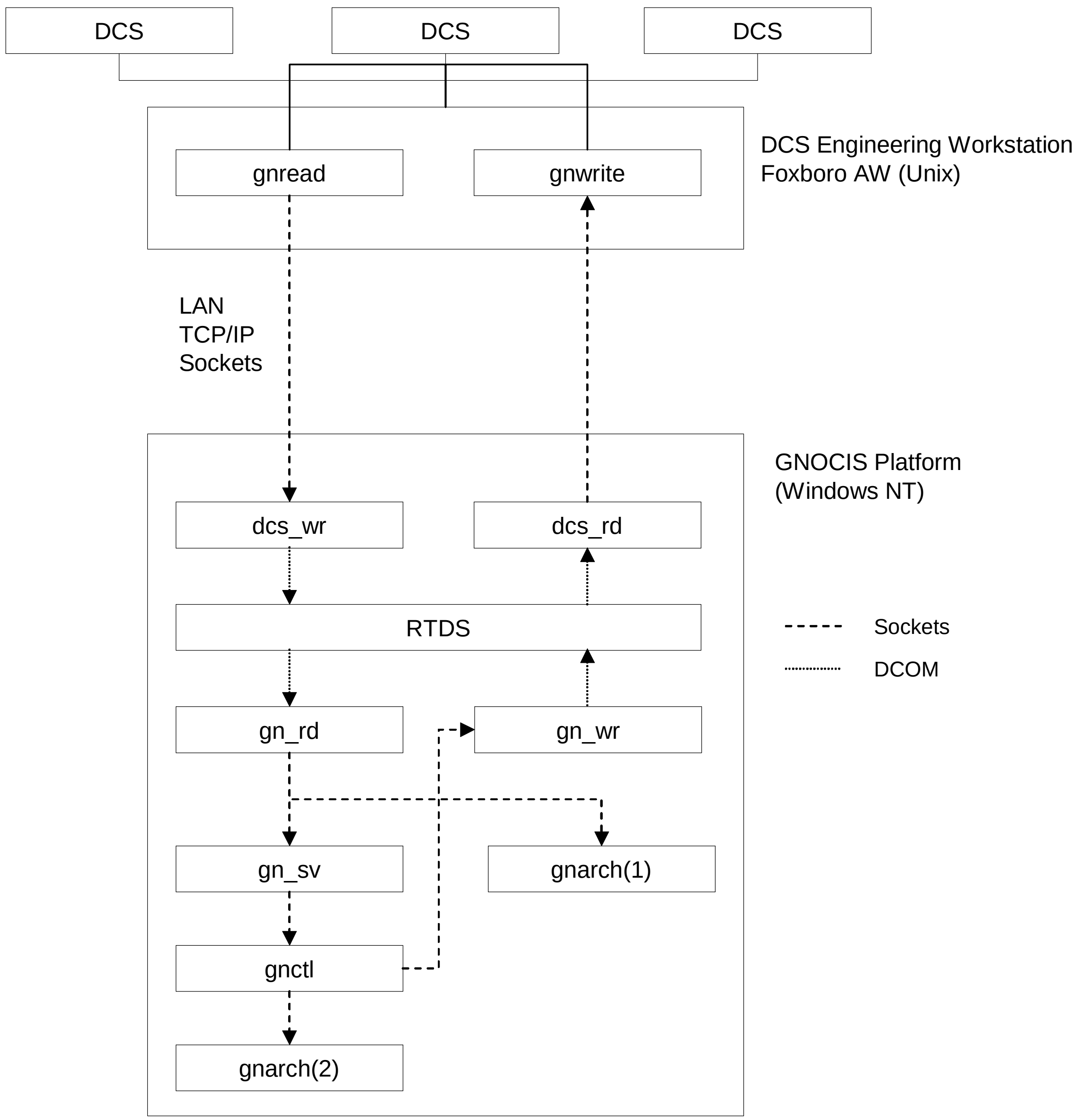


Figure 8-30 Revised GNOCIS Interface to DCS

GNOCIS Boiler Model Modifications

As part of this scope addition, the GNOCIS model was modified substantially to reflect changing plant conditions and integration with the unit optimization package. The configuration of the boiler model in 1996 is shown in Table 8-7 and the most recent revised model is shown in Table 8-8. The major modifications include the addition of inputs and outputs to be able to predict the impact of boiler operating decisions on steam conditions (main and reheat temperature, main steam pressure) and its primary control variables (spray flows and reheat damper position). The revised model performs reasonably well, as shown in Figure 8-31 through Figure 8-36. In these figures, the predicted value is from the base model and not the error corrected model.

Table 8-7 GNOCIS Boiler Variables Original (Model Ham31H)

Variable	Description	Type
Model Inputs		
WMILLAC	Mill A Coal Flow, lb/hr	M
WMILLBC	Mill B Coal Flow, lb/hr	M
WMILLCC	Mill C Coal Flow, lb/hr	M
WMILLDC	Mill D Coal Flow, lb/hr	M
WMILLEC	Mill E Coal Flow, lb/hr	M
WMILLFC	Mill F Coal Flow, lb/hr	M
YAOFAF1	Overfire Air Control Damper Front 1	M
YAOFAR1	Overfire Air Control Damper Rear 1	M
YAOFAF2	Overfire Air Control Damper Front 2	M
YAOFAR2	Overfire Air Control Damper Rear 2	M
AVG_O2	Average Excess Oxygen, %	M
Model State		
O2LH	Excess Oxygen, Left Hand (West), %	S
O2RH	Excess Oxygen, Right Hand (East), %	S
WTOTSA	Secondary Air Flow, lb/hr	S
Model Outputs		
NOX_LBMMBTU	NOx Emissions, lb/MBtu	O
CIA	Fly ash Carbon-in-Ash, %	O
EFF	Boiler Efficiency (expanded), %	O

Table 8-8 GNOCIS Boiler Variables Revised (Model HamGO8)

Variable	Description	Type
Model Inputs		
WMILLAC	Mill A Coal Flow, lb/hr	M
WMILLBC	Mill B Coal Flow, lb/hr	M
WMILLCC	Mill C Coal Flow, lb/hr	M
WMILLDC	Mill D Coal Flow, lb/hr	M
WMILLEC	Mill E Coal Flow, lb/hr	M
WMILLFC	Mill F Coal Flow, lb/hr	M
YAOFAF1	Overfire Air Control Damper Front 1	M
YAOFAR1	Overfire Air Control Damper Rear 1	M
YAOFAF2	Overfire Air Control Damper Front 2	M
YAOFAR2	Overfire Air Control Damper Rear 2	M
AVG_O2	Average Excess Oxygen, %	M
AVG_TSAAI	Average Temperature Secondary Air Heater Air Inlet, °F	I
AVG_TPAAI	Average Temperature Primary Air Heater Air Inlet, °F	I
SCR_ON	SCR On = 1/ Off = 0	I
TMS_Setpoint	Main Steam Temperature Setpoint, °F	M
THRH_Setpoint	Hot Reheat Temperature Setpoint, °F	M
PMS_Setpoint	Main Steam Pressure Setpoint, psig	M
Model State		
AVG_TSAGO	Average Temperature Secondary Air Heater Gas Outlet, °F	S
AVG_TPAGO	Average Temperature Primary Air Heater Gas Outlet, °F	S
AVG_DIV_WALL_INLET_T	Average Division Wall Inlet Temperature, °F	S
HOT_REHEAT_PRESS	Hot Reheat Pressure, psig	S
SUPERHEAT_INLET_TEMP	Superheat Inlet Temperature, °F	S
Model Outputs		
NOX_LBMMBTU	NOx Emissions, lb/MBtu	O
CIA	Fly ash Carbon-in-Ash, %	O
THRH	Hot Reheat Temperature, °F	O
TMS	Main Steam Temperature, °F	O
PMS	Main Steam Pressure, psig	O
SH_SPRAY_FLOW_UPPER	Superheat Spray Flow Upper, lb/hr	O
SH_SPRAY_FLOW_LOWER	Superheat Spray Flow Lower, lb/hr	O
EFF	Boiler Efficiency (expanded), %	O
RH_DAMPER_POS	Reheat Pass Damper Position, %	O

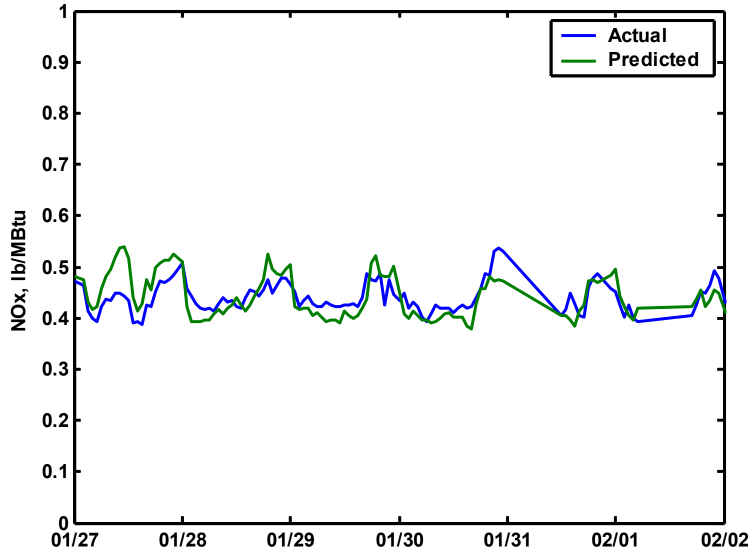


Figure 8-31 Predicted vs. Actual NOx Jan 27 – Feb 2, 2002

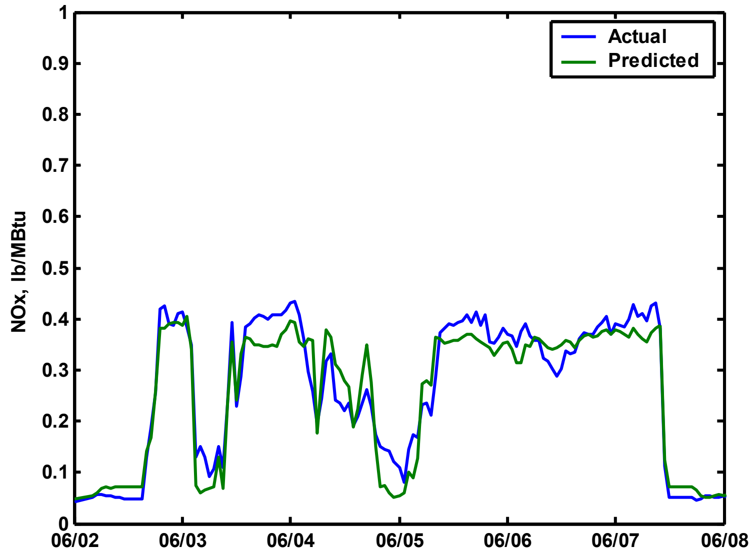


Figure 8-32 Predicted vs. Actual NOx June 2 – June 8, 2002

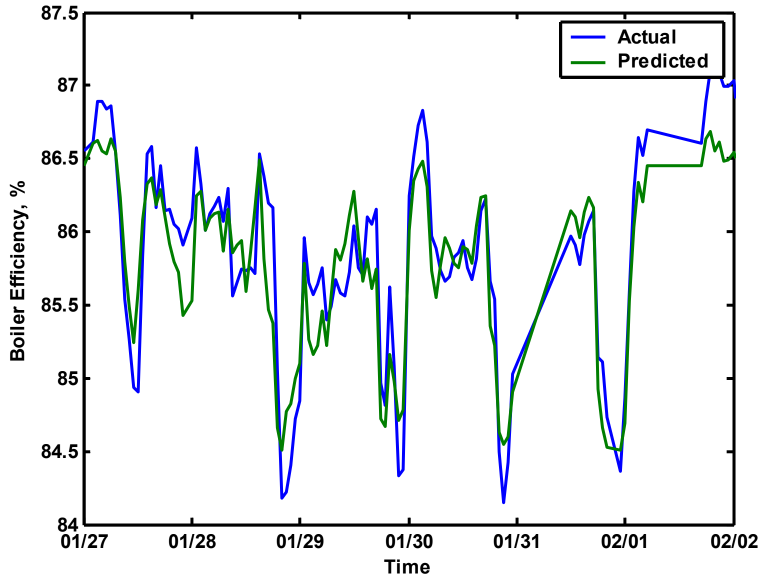


Figure 8-33 Predicted vs. Actual Efficiency Jan 15 – Feb 2, 2002

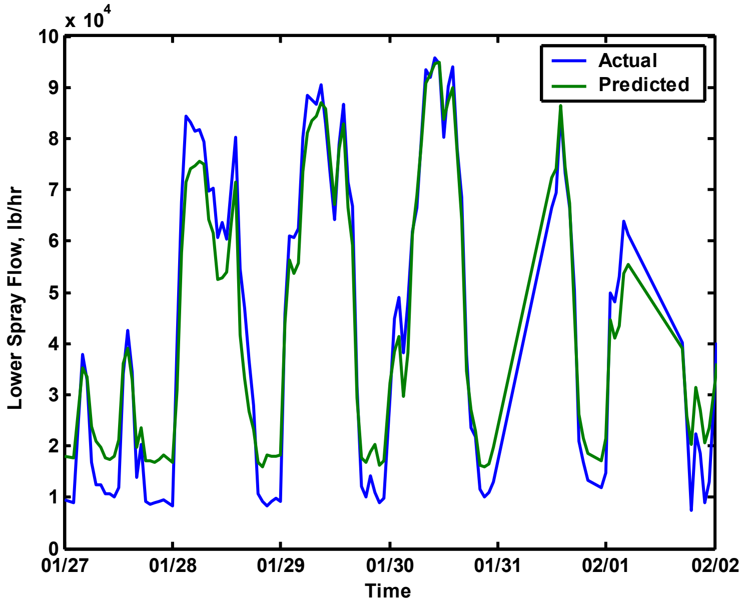


Figure 8-34 Predicted vs. Actual SH Spray Flow Upper Jan 17 – Feb 2, 2002

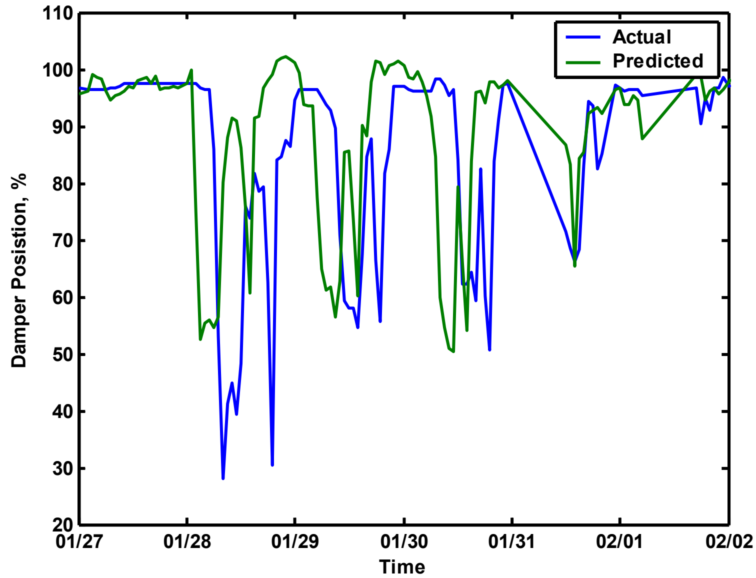


Figure 8-35 Predicted vs. Actual RH Pass Damper Position Jan 17 – Feb 2, 2002

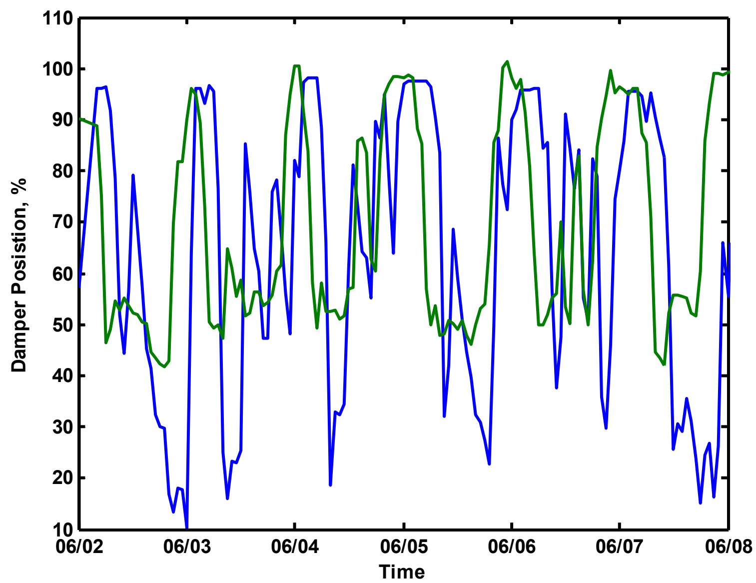


Figure 8-36 Predicted vs. Actual RH Pass Damper Position June 2 – June 8, 2002

Testing

Insufficient testing was performed to quantify the performance of the GNOCIS models at the site. Testing with an interim model was conducted during January 2002, but the results of this testing were inclusive in part due to the unit being under economic dispatch during the testing and resultant load changes.

Summary

GNOCIS is a real-time, closed-loop system for performing boiler optimization. GNOCIS was first installed at Hammond 4 in 1996 and was upgraded as part of this current project. A major improvement was the development and incorporation of on-line model error correction. This error correction greatly improves the accuracy and robustness of the neural-network combustion models. An operator interface exists on the DCS for this component and this system is capable of both open- or closed-loop operation. The current configuration makes recommendations on excess O_2, feeder coal flows, and overfire air for optimizing NOx emissions, boiler efficiency, and fly ash unburned carbon. Although previous testing of GNOCIS at this site showed substantial benefits may be obtained by its application, the testing conducted during this most recent phase is inconclusive. Potential areas for further work on this package include:

- Testing to confirm performance
- Interfacing with the plant's recently installed plant information network
- Confirmation of performance with the unit optimization package

9
TURBINE CYCLE OPTIMIZATION

Overview

The overall goal of this project was to demonstrate online, optimization techniques to several power plant processes and to the unit as a whole. Processes included the boiler, intelligent sootblowing, precipitator, and the unit as a whole. As part of the project, it was proposed that another process be identified and included as part of the optimization mix. Based on a study conducted by ENTEC, it was found that for Hammond 4, steam conditions had the highest cost impact of any parameter under consideration. The design of this system is discussed in this section.

Current Practice

Throttle and Reheat Temperature Control

By far, the most common practice in the utility industry is to have a fixed setpoint on steam conditions (throttle temperature and reheat temperature). The setpoints are generally design values set by the boiler and/or turbine manufacturer. In some instances, the design setpoints cannot be attained due to current combustion conditions or design or fuel deficiencies.

For Hammond 4, superheat temperature is controlled at two different locations in the boiler. First, the division wall inlet superheat temperature is controlled by the use of the left and right hand lower attemperating sprays. The setpoint for these control loops is 20ºF above the minimum of drum saturation temperature and 700ºF. The final superheat temperature is controlled by the use of the left and right hand upper attemperating sprays. The setpoint for this loop is normally 1000ºF and can be set by the operator.

Reheat temperature at Hammond 4 is controlled through modulation of the bypass dampers in the furnace backpass. The setpoint for this loop is normally 1000ºF and can be set by the operator. Although configured in the DCS, reheat attemperating spray as a reheat temperature control method is not currently used at Hammond 4.

Pressure Control

Main steam flow and hence generated load is effectively controlled by setting the steam pressure at the inlet to the first stage nozzles of the high-pressure turbines. This can be accomplished by either (1) throttling the steam flow by modulating the governor (or throttle) valves of the turbine while maintaining constant upstream conditions; (2) varying the steam pressure ahead of the turbine; or (3) some combination of the above. For the latter two, the pressure setpoint is adjusted so that the throttle valves operate at valve points, i.e. where no valve is partially open.

Constant Pressure Operation

At Hammond, the Unit Master Station (UMS), when in automatic mode, will always try to control turbine throttle pressure to setpoint (at Hammond 4, normally set to 2400 PSIG). The UMS will normally control throttle pressure by adjusting the fuel firing rate (boiler follow mode) or, in unusual circumstances, by modulating the turbine governor valves (turbine follow mode).

Sliding Pressure

In sliding pressure operation, the throttle pressure setpoint (and therefore throttle pressure) is varied to achieve load demand while the turbine throttle valves are controlled to wide-open position (VWO - Valves Wide Open). The primary advantages of this mode of operation are:

- Reduced throttling losses from the governor valves increases turbine efficiency.
- Boiler feedpump power consumption is reduced at lower loads.
- Higher superheat temperatures at reduced loads improves turbine cycle efficiency.

A disadvantage of sliding pressure operation is slower response time for the unit.

At Hammond 4, the DCS has been configured to allow sliding pressure operation but testing has not been conducted to develop the necessary setpoint curves and it is not utilized.

Recommendation Based on ENTEC Study

Based on the Total Plant Optimization study, ENTEC recommended that the steam turbine inlet conditions be considered in the process optimization mix[1]. Specific parameters recommended include main steam temperature, main steam pressure and reheat pressure. Total combined impact on income for these control variables was projected to be $457,000 (Table 9-1). This study assumed that the indicated changes in steam conditions are achievable. The impact of these variables on turbine cycle heat rate is shown in Figure 9-1 through Figure 9-3. These

[1] Further information is provided in the section *Application of EPRI's TPCO Guidelines*.

curves, although not specific to Hammond, are typical curves for a single reheat unit and are representative of Hammond 4 [Ame85]. Using these charts, the estimated impact of increased operating setpoints on heat rate and operating costs is shown in Table 9-2.

As an example, the 1999 load characteristics for these variables are shown in Figure 9-4 through Figure 9-6. These controllable parameters operated on average, below the design setpoints, particularly at mid-to-lower load categories. The cumulative financial impact of these deviations are affected by the load profile during the period (Figure 9-7). Using the heat rate deviation curves, load characteristics, and load profile, the load weighted average heat rate impact from these controllable parameters being off-design is approximately 44 Btu/kWh or $250,000/year for this unit.

Table 9-1 Prioritization of Control Variables

	D. PRIORITIZATION OF CONTROL VARIABLES		**Ranking**	**Impact of Each Variable on Operating Income**			
	Ranking of Parameter Impacts	**Units**	**of Variable**	$ x 1,000	% of Overall	**Range (Low to High)**	
1	Coal Quality (Name & % in Blend, A/B/C)	Name & %	11	$0	0.0%	100% KY-HammB to 100% KY-HammB	
2	Excess Air Downstream of Economizer	%	10	$1	0.2%	27 to 19	
3	Air Heater Leakage	%	11	$0	0.0%	11 to 13	
4	Overfire Air Damper Setting	% Open	9	$3	0.4%	0% to 100%	
5	Superheater Spray Flow	lb/hr	8	$23	2.8%	0 to 75415	
6	Reheater Spray Flow	lb/hr	5	$111	13.4%	0 to 37707	
7	Main Steam Throttle Pressure	psig	2	$158	18.9%	2300 to 2420	
8	Main Steam Temperature	° F	1	$166	20.0%	990 to 1010	
9	Reheat Steam Temperature	° F	4	$135	16.2%	990 to 1010	
10	Cycle Makeup Water	% of MS Flow	3	$149	17.9%	0% to 1%	
11	Condenser Cleanliness	Fraction	6	$60	7.2%	0.55 to 0.70	
12	Velocity of Water in Condenser Tubes	ft/sec	7	$25	3.0%	6.0 to 7.0	
13	Limestone Stoichiometry	moles CaCO3/mole SO2	11	$0	0.0%	#N/A	
14	Ammonia-to-NOx Ratio	moles NH3/mole NOx	11	$0	0.0%	#N/A	
	OVERALL IMPACT OF CONTROL VARIABLES (TOTAL IMPACT ON INCOME)			$833	**100%**		
	TABLE D NOTES:						
	1. All costs are absolute values						
	2. Rank is based on absolute value of costs						

Table 9-2 Potential Benefits as a Result of Heat Rate Improvement

Parameter	Nominal	Change	Delta[3] TCHR	Delta[4] GUHR	Annual[2] Benefit (k$)	APW[1] (k$)
Throttle Temperature	1000ºF	+10ºF	-13.8	-15.3	69	966
Reheat Temperature	1000ºF	+10ºF	-11.7	-13.0	59	821
Throttle Pressure	2400 psig	+100 psig	-26.6	-29.6	133	1869
Total	-----------	------------	(55.1)	(61.9)	259	3655

[1]APW - Accumulated Present Worth through 2020
[2] Cumulative benefit over all load ranges.
[3]TCHR - Turbine cycle heat rate; assumed to be 8500 BTU/KWHR.
[4]GUHR - Gross Unit Heat Rate; calculated from TCHR assuming boiler efficiency of 90%.

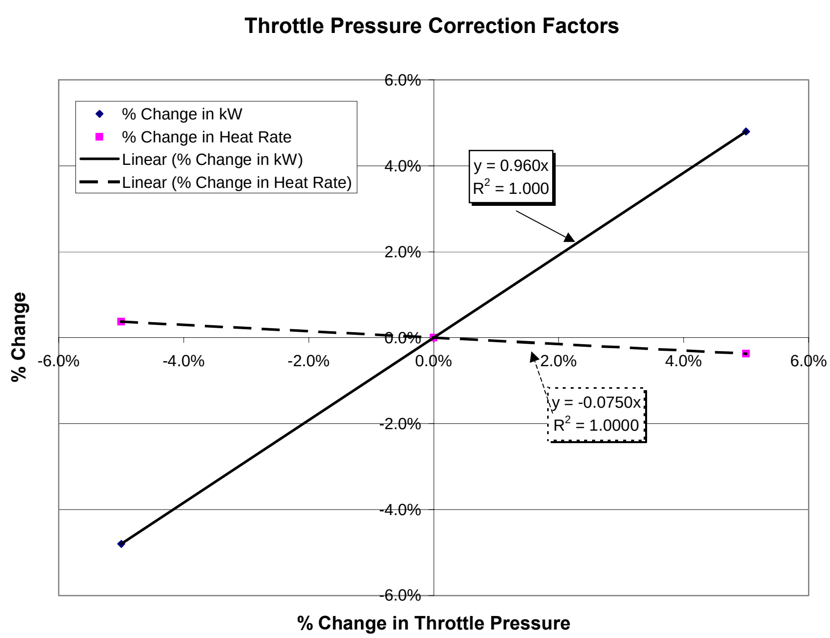


Figure 9-1 Throttle Pressure Correction Factors for Load and Heat Rate

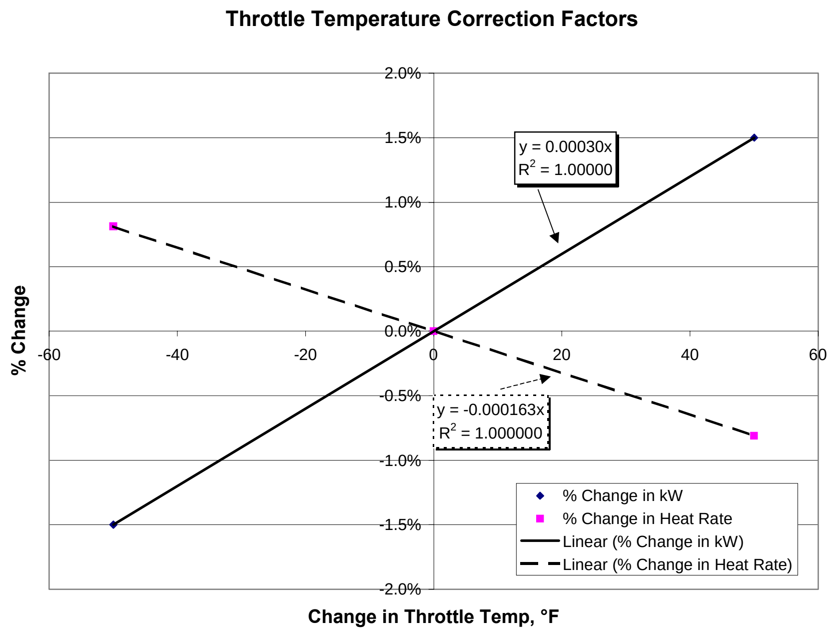


Figure 9-2 Throttle Temperature Correction Factors for Load and Heat Rate

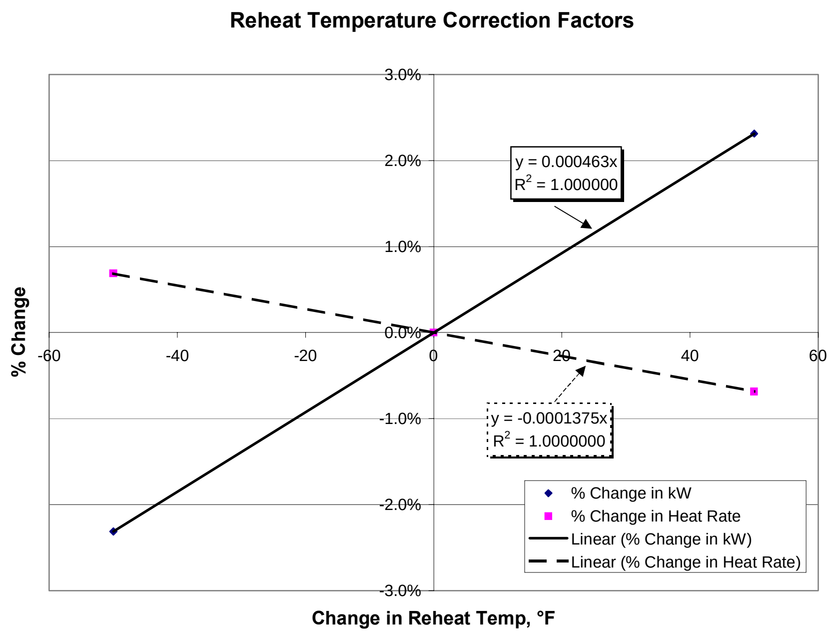


Figure 9-3 Reheat Temperature Correction Factors for Load and Heat Rate

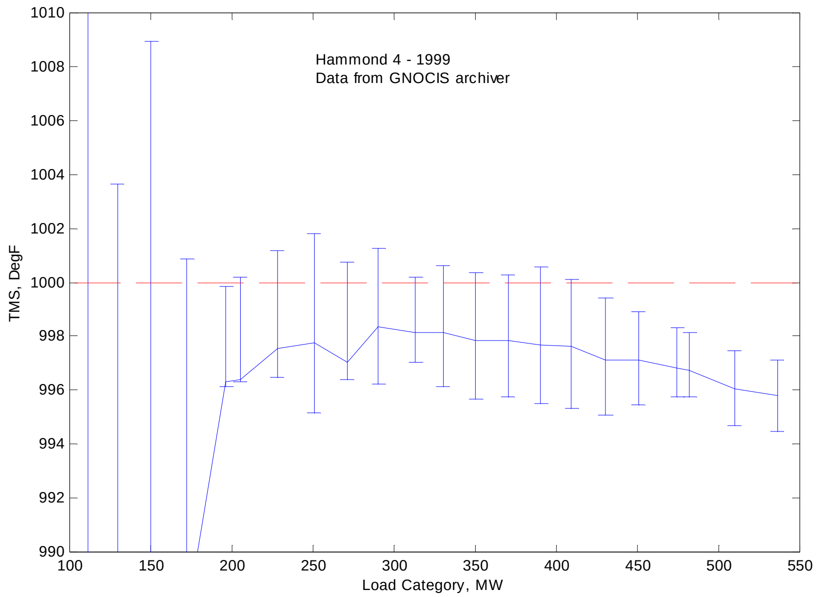


Figure 9-4 Main Steam Temperature vs. Load for 1999

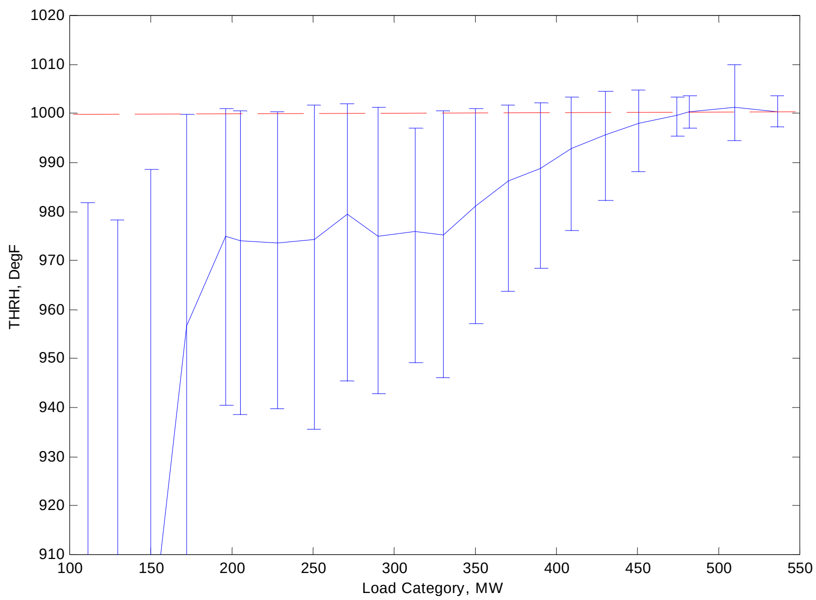


Figure 9-5 Hot Reheat Temperature vs. Load for 1999

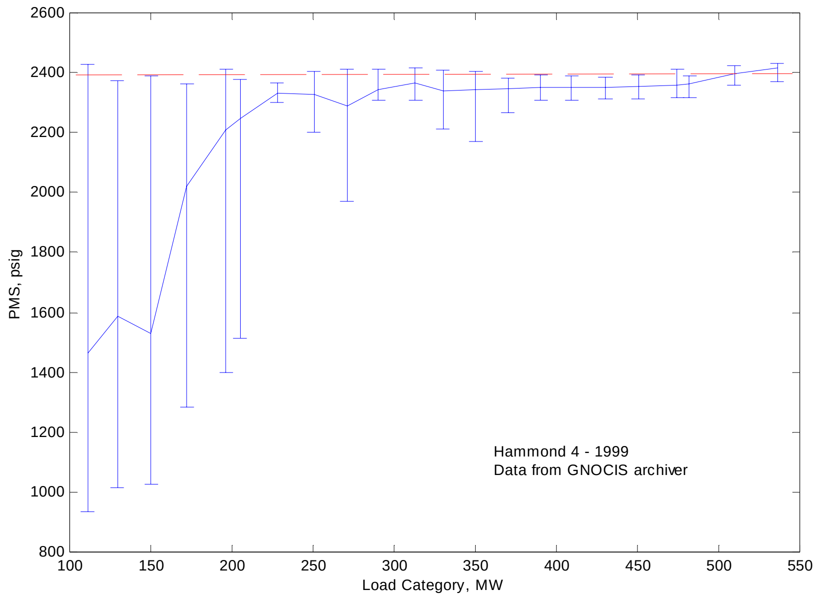


Figure 9-6 Main Steam Pressure vs. Load for 1999

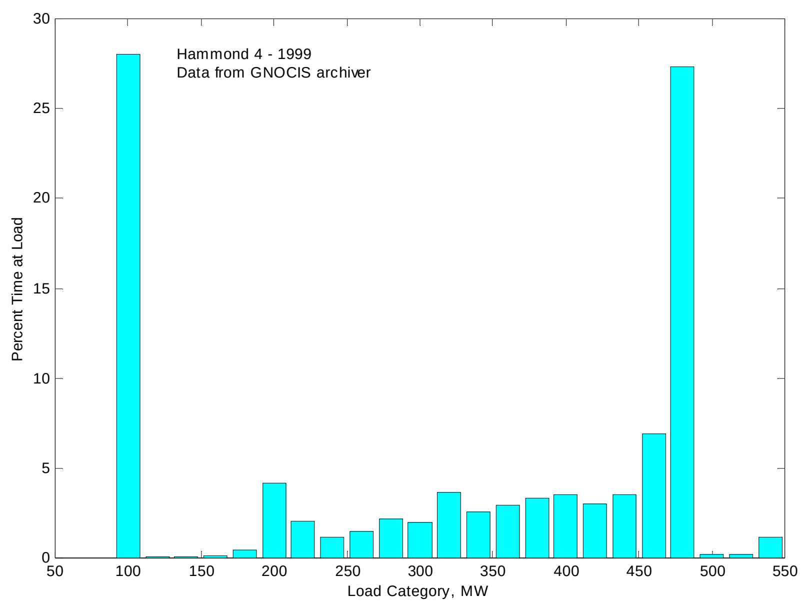


Figure 9-7 Load Profile for 1999

Optimization

There are two aspects of the use of these variables in the optimization mix. The first is that the ability to reach a target temperature and pressure is highly dependent on boiler operating conditions. This suggests that these variables should be the target variables of the boiler optimization system (GNOCIS). Secondly, these variables are the primary determinates of the turbine cycle performance and, therefore, should be outputs of the turbine cycle optimization system. One possible approach is shown in Figure 9-8. In this approach, the turbine cycle optimizer would make recommendations as to main steam pressure (PMS), main steam temperature (TMS), and hot reheat temperature (THRH) (within hard constraints specified in the DCS) to minimize the operating cost of this component. These recommended setpoints are then used by the DCS and boiler optimization program as inputs along with others to generate a boiler configuration in which the boiler control system can achieve the turbine inlet setpoints. The system should be able to work in these modes:

- Full coordination - The unit optimization package would coordinate the two systems, allocating resources among the two systems to minimize overall operating cost.
- Partial Coordination - The unit optimization package would still coordinate the two packages but the direct link between the packages (PMS, TMS, THRH) is broken.
- Independent Operation - The two packages operate independently without unit optimization coordination or direct linkage.

Conceptual control modifications required for the main steam control loop are shown in Figure 9-9. For this loop, the recommended throttle pressure setpoint is compared to the normal DCS generated pressure setpoint. The difference between the two is high, low, and rate limited and added back as a setpoint bias. This logic and limits reside within the DCS and are not modifiable by the optimization software. When the optimizer is disabled (for the loop or entirely), the generated bias is bled to zero, effectively setting the final setpoint to the DCS generated setpoint. When clamped, the bias would be clamped to the existing value. The limits (including rate) could be configured to be a function of other plant operating conditions such as load or stability criteria. A consistent set of high and low limits must be maintained by the DCS and unit optimizer and therefore the optimizer must read the DCS limits every optimization cycle or when the limits are modified. Similar control logic would be used for the main steam and reheat steam temperature controls.

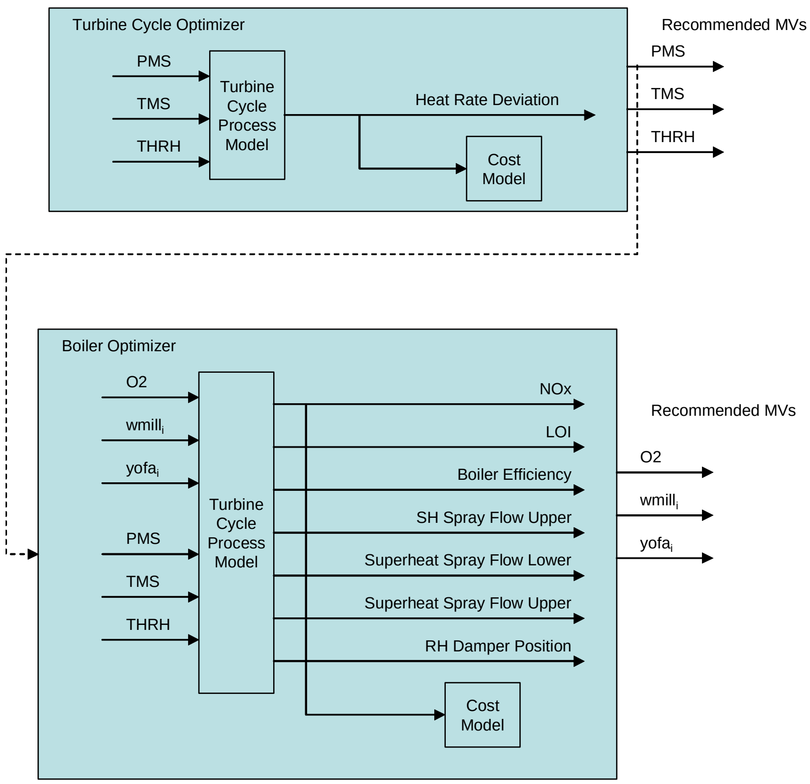


Figure 9-8 Integration of the Steam Cycle / Boiler Optimization Packages

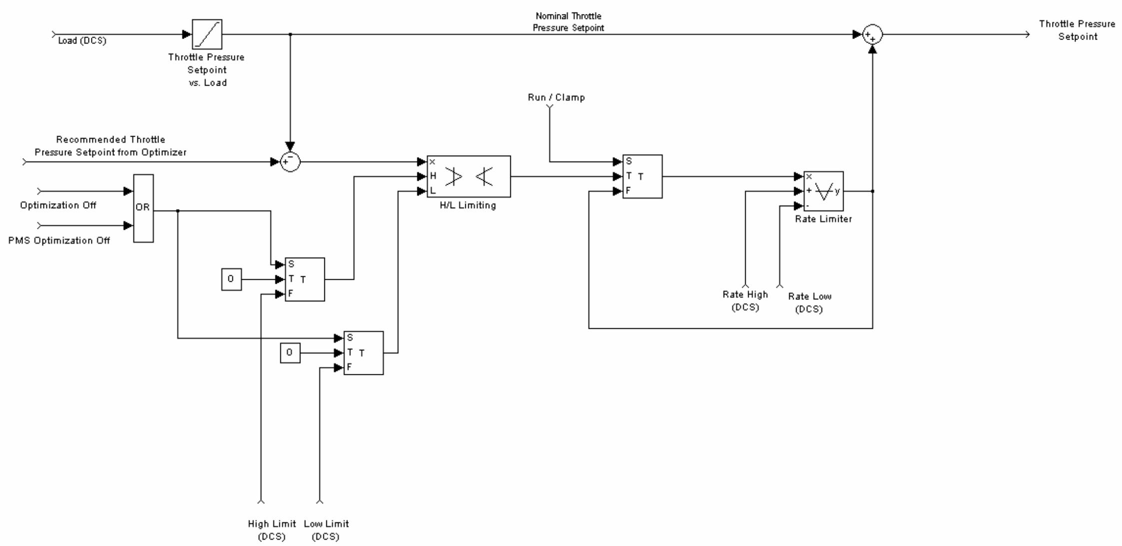


Figure 9-9 Main Steam Pressure Control Loop with Optimizer Interface

Implementation

GNOCIS is used as the core of this optimization package, using largely the same code base as that used for the boiler (see section *Boiler Optimization Package* for a full-description of GNOCIS). Differences between the two packages (GNOCIS/boiler and GNOCIS/turbine) include:

- GNOCIS/turbine is open-loop only while GNOCIS/boiler will run open- or closed-loop
- Different process models

The layout of the GNOCIS/turbine software is shown in Figure 9-10. As with the boiler package, the RTDS serves as an intermediary between the DCS and GNOCIS. Running at typically one-minute intervals GNOCIS/turbine collects information from the RTDS, performs an optimization run, and then transmits the current recommendations to the RTDS. GNOCIS/turbine runs on a separate PC than GNOCIS/boiler[1] mainly due to restrictions imposed by the Pavilion software license manager and loading considerations. In most GNOCIS installations, the recommendations would be transmitted to the DCS for implementation automatically or on operator review, but this has not yet been implemented for the GNOCIS/turbine package.

Although overall less desirable, as an alternative and interim measure, engineer and operator interaction is provided through a program residing on a PC and communicating with GNOCIS/turbine via the RTDS. The panels for this program are shown in Figure 9-11 and Figure 9-12. Through these panels, the user may change limits and view the results of the optimization. Configuration of this panel is through a configuration file, an example of which is shown in Figure 9-13. The configuration of this program should be coordinated with the configuration of the GNOCIS *gnctl* initialization file.

The GNOCIS/turbine model is simple and the inputs and outputs of this model are shown in Table 9-3. The curves used to train the model (Table 9-4) are based on Westinghouse supplied correction factors to turbine-cycle heat rate for variations in throttle pressure, throttle temperature, and reheat temperature. Although the relationship between the output and input is near-linear with the process variables, to use the GNOCIS package, it was necessary to create neural network facsimiles of these curves and the results of this modeling is shown in Figure 9-14 and Figure 9-15.

[1] The RTDS is also hosted on another PC. The RTDS shown in the figure is a surrogate for the actual data repository.

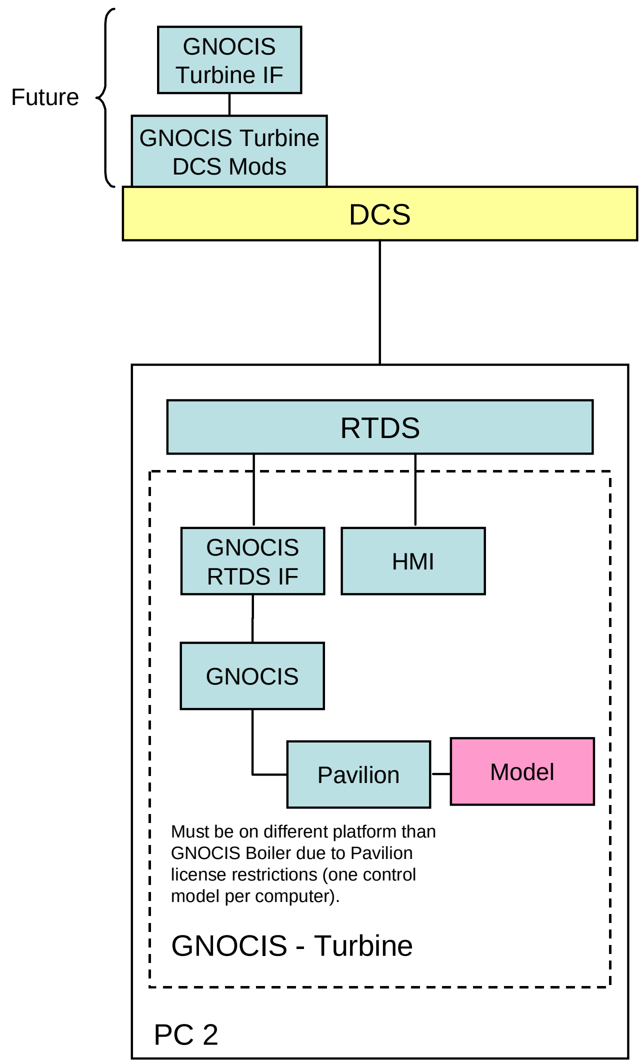


Figure 9-10 GNOCIS Turbine Implementation

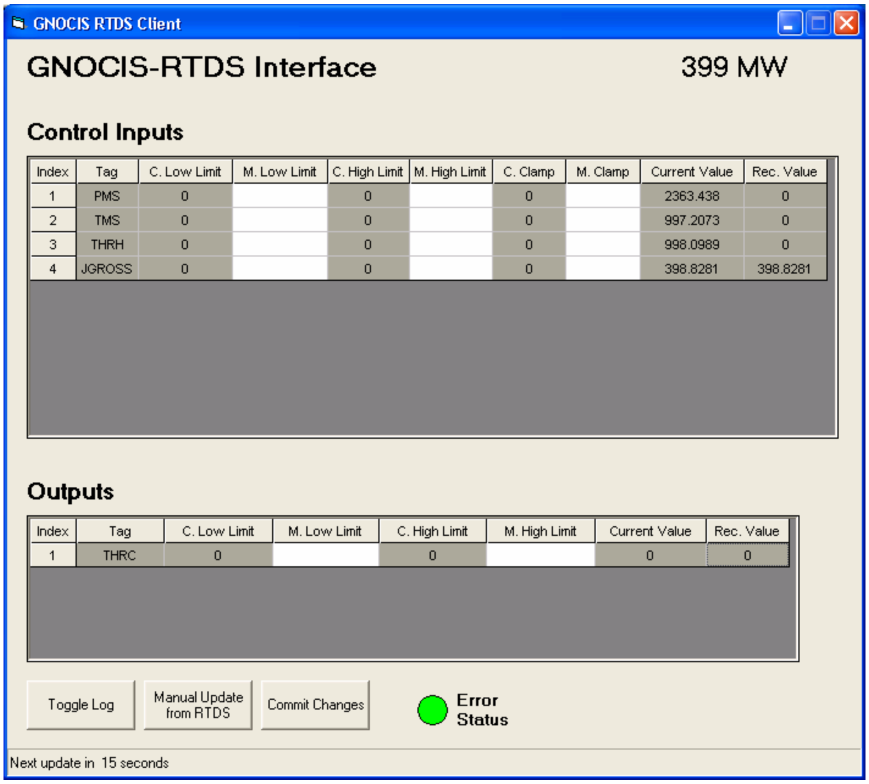


Figure 9-11 GNOCIS-RTDS Interface Used for Turbine Optimization

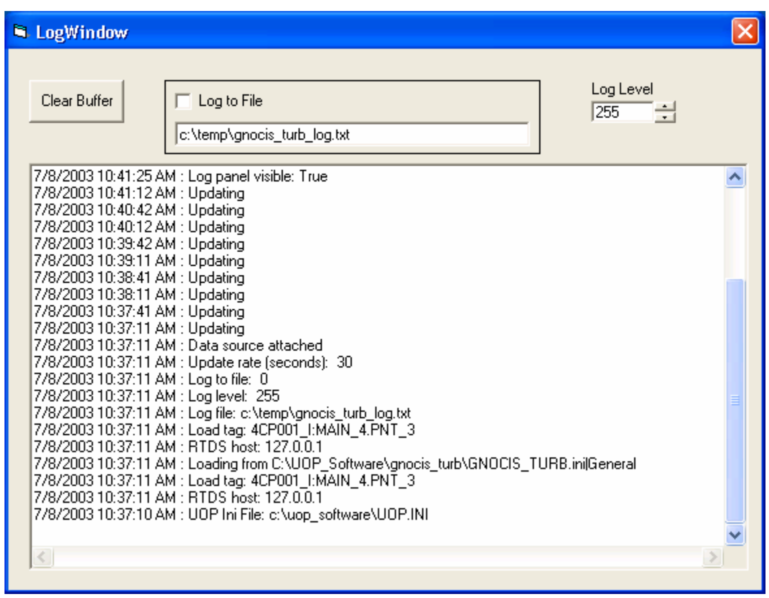


Figure 9-12 GNOCIS-RTDS Interface Used for Turbine Optimization

```
[General]
RTDSHost = 127.0.0.1
UpdateRate = 30
LogLevel = 255
LogFile = c:\temp\gnocis_turb_log.txt
LogToFile = 0
Inputs = PMS,TMS,THRH,JGROSS
Outputs = THRC

[PMS]
SetpointBias = 4UMSBOILER:THTPSP_RATEL.OUT
BoundLower = 4TURB:INMIN_1
BoundUpper = 4TURB:INMAX_1
Clamped = 4TURB:CLAMPED1
CurrentValue = 4CP001_I:MAIN_1.PNT_3
OptimumValue = 4TURB:INRC_1

[TMS]
SetpointBias = 4UPSHSPRAY:FSHOUT_CTRLR.SPT
BoundLower = 4TURB:INMIN_2
BoundUpper = 4TURB:INMAX_2
Clamped = 4TURB:CLAMPED2
CurrentValue = 4CP002_I:MAIN_3.PNT_7
OptimumValue = 4TURB:INRC_2

[THRH]
SetpointBias = 4SHPASS:PASDMP_CTRLR.SPT
BoundLower = 4TURB:INMIN_3
BoundUpper = 4TURB:INMAX_3
Clamped = 4TURB:CLAMPED3
CurrentValue = 4CP002_I:MAIN_2.PNT_8
OptimumValue = 4TURB:INRC_3

[JGROSS]
SetpointBias = 4CP001_I:MAIN_4.PNT_3
BoundLower = 4TURB:INMIN_4
BoundUpper = 4TURB:INMAX_4
Clamped = 4TURB:CLAMPED4
CurrentValue = 4CP001_I:MAIN_4.PNT_3
OptimumValue = 4CP001_I:MAIN_4.PNT_3

[THRC]
BoundLower = 4TURB:INMIN_5
BoundUpper = 4TURB:INMAX_5
CurrentValue = 4TURB:OUTM
OptimumValue = 4TURB:OUTP
```

Figure 9-13 GNOCIS-RTDS Interface Configuration File for Turbine Package

Table 9-3 GNOCIS Turbine Variables (Model ham4_2k_turb2)

Variable	Description	Type
Model Inputs		
GROSS_MW	Gross Unit Load, MW	I
PMS	Main Steam Pressure, psig	M
TMS	Main Steam Temperature, °F	M
THRH	Hot Reheat Temperature, °F	M
Model State		
--- None ---		
Model Outputs		
TOT_PERCENT_HRDEV	Total Percent Heat Rate Deviation	O

Table 9-4 Heat Rate Deviation Transforms (Model ham4_2k_turb2)

Heat Rate Deviation	Transform
Throttle Pressure	$if(!LOAD_PERCENT! < 75.0, !PMS! * (.0516 * !LOAD_PERCENT! - 5.87) / 1000. + 14.75 - .1246 * !LOAD_PERCENT!, !PMS! * (.0094 * !LOAD_PERCENT! - 2.705) / 1000. + 6.534 - .0227 * !LOAD_PERCENT!) " }
Throttle Temperature	$if(!LOAD_PERCENT! < 75.0, !TMS! * (- .006 * !LOAD_PERCENT! - 1.25) / 100. + .06 * !LOAD_PERCENT! + 12.5, - .017 * !TMS! + 17.0) " }
Reheat Temperature	$if(!LOAD_PERCENT! < 75.0, !THRH! * (.003 * !LOAD_PERCENT! - 1.575) / 100.0 + 15.75 - .03 * !LOAD_PERCENT!, - !THRH! * .0135 + 13.5) " }

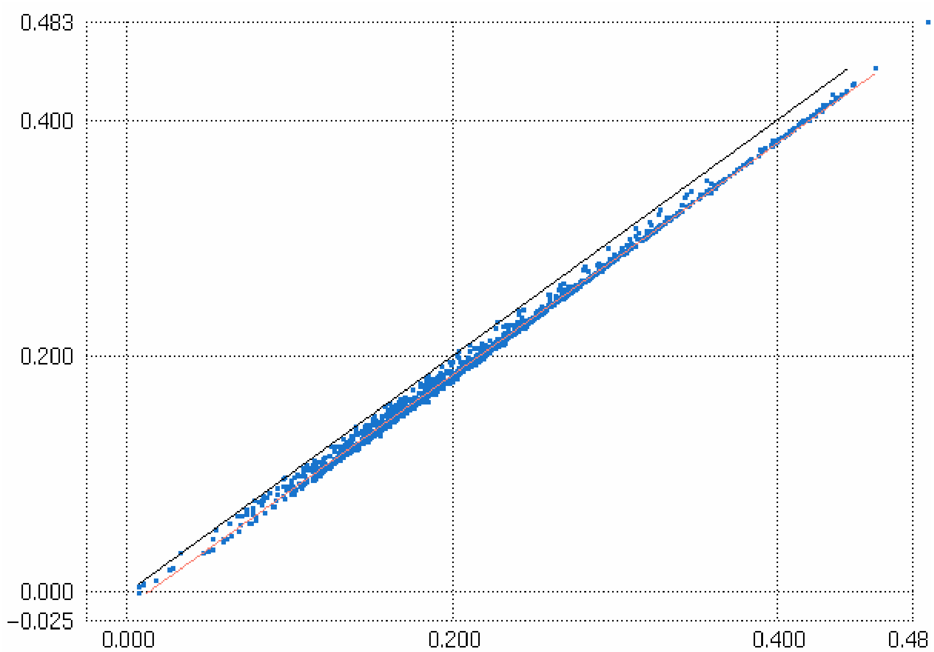


Figure 9-14 Predicted vs. Actual (Model ham4_2k_turb2)

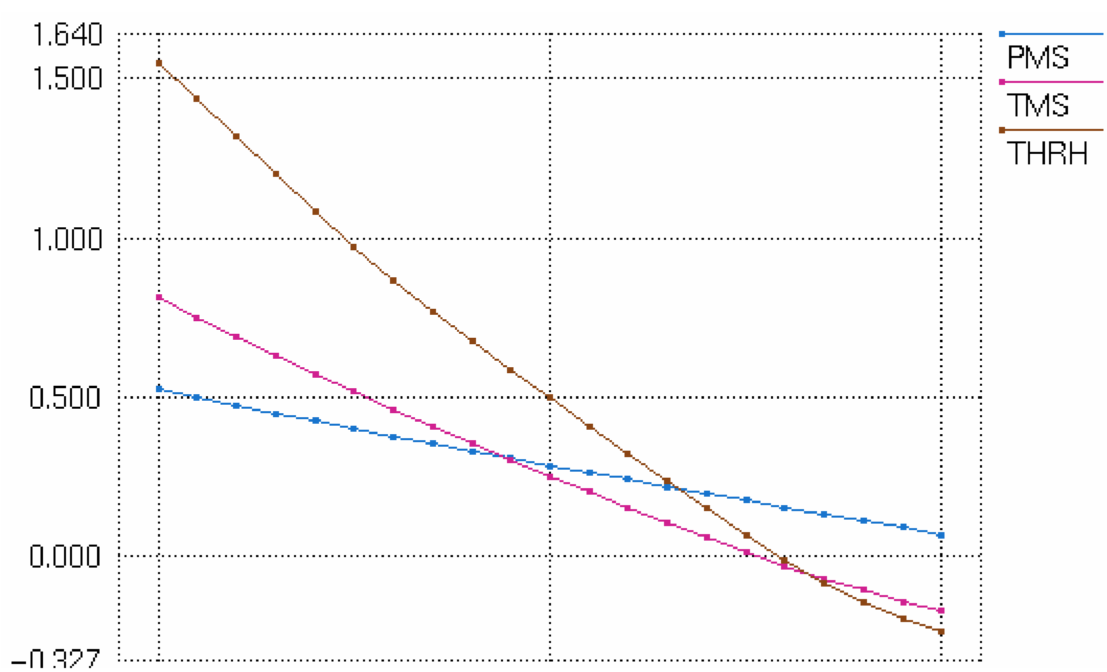


Figure 9-15 Output vs. Input (Model ham4_2k_turb2)

Testing

Insufficient testing was performed to quantify the performance of the system. Open-loop testing with an interim model was conducted during January 2002, but the results of this testing was not positive in part due to the unit being under economic dispatch during the testing and resultant load changes. As the result of that testing, the model structure was revised. Further testing needs to be performed to determine the benefits of this system.

Summary

The GNOCIS package was modified to make recommendations on turbine steam inlet conditions to maximize turbine cycle efficiency. Modifications included development of process models and development of a user interface. Insufficient testing was conducted to quantify the performance of the system. Areas of possible future work include:

- Testing to confirm performance
- Integration with DCS and enabling closed-loop control
- Interfacing with the plant's recently installed plant information network
- Confirmation of performance with the unit optimization package

10
ESP PACKAGE

Overview

Precipitator performance such as measured by outlet opacity, particulate removal rate, and energy consumption is greatly dependent on precipitator inlet conditions. These conditions are in turn a function of boiler operating conditions and possibly other post-combustion emission control technologies (SNCR and SCR). Given the dependence of ESP performance on upstream operating conditions and importance of its operation on environmental performance, it was felt that the ESP should be brought into the optimization envelope.

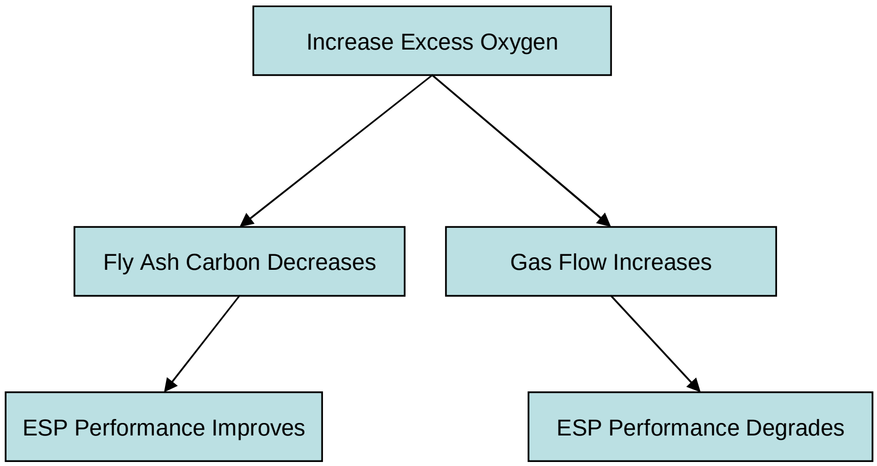
EPRI's ESPert was installed at the site as part of this project. The ESPert package, originally developed in the 1990s, is a diagnostic and predictive model for ESPs designed to evaluate and predict ESP performance and diagnose problems. ESPert interfaces with PCAMS, a supervisory control system for the ESP. Initial expectations were to use the ESPert/PCAMS software as an optimization platform; however to date, it has been used only as a predictive model.

Process Description

Hammond Unit 4 is equipped with a Research-Cottrell precipitator. Although the unit had been equipped with an ESP since 1979, it proved inadequate with the installation of the LNB and AOFA [SCS97A], and therefore, major modifications were made to it in 1994. The current design specifications are shown in Table 10-1. The flue gas exiting each side of the furnace flows into a separate duct designated the A or B side (Figure 10-1). Effectively four ESPs, two each for the A and B sides, remove particulate matter from the flue gas. The four sections are labeled A1, A2, B1, and B2. The ESP is a cold-side design, located downstream of the air heaters having a design gas inlet temperature of 330°F, velocity of 5.15 ft/s, and specific collection area of 379 ft^2. Design efficiency at full load is 99.65%.[1] An SO_3 flue gas conditioning system is installed and is used, as needed, to change the resistivity of the ash to improve its collection. Actual operating conditions and resultant opacity levels for 2002 are shown in Figure 10-2 through Figure 10-8. Generally the input conditions and stack opacity remained well within the design parameters and regulatory limits over the load range. Excess

[1] ESP Efficiency = (ESP_Inlet_Ash_Loading – ESP_Outlet_Ash_Loading) / (ESP_Inlet_Ash_Loading)*100

oxygen levels affect fly ash carbon levels and flue gas flow rates and these in turn affect ESP performance, generally as follows:



Given the counteracting effects, the direction of the response may be either an improvement or deterioration. The impact on how it may have impacted opacity levels on Hammond 4 is shown in Figure 10-8 in which it appears that, at least at higher loads, opacity reductions are associated with both low and high excess oxygen levels.

Dry ash collected in the economizer and ESP hoppers is pneumatically transported to a tank where it is mixed with water and sluiced to a settling pond. Bottom ash from the boiler is sluiced to a separate settling pond. The water used for ash sluicing is recycled water from the settling ponds.

Table 10-1 Hammond 4 ESP Design Characteristics

Manufacturer	Research-Cottrell
Hot or Coldside	Cold
Design Conditions	
Gas Inlet Temperature	330°F
Total Flue Gas Volume	2,450,000 ACFM
Gas Velocity	5.15 ft/s
Gross SCA	213 ft^2/1000 ACFM
Equivalent Cross Sectional Area	7929 ft^2
Efficiency	99.65%
Regulations	
Particulate Emission Limit	0.240 lb/MBtu
Opacity	40%
Fuel	
Sulfur	1.5%
Ash	11%
Moisture	5%
Higher Heating Value	13,000 Btu/lb
Conditioning System	SO_3
T-R Set	
T-R Controls	NWL
Number of T-R Sets	24
Rappers	
Plate Rapper Type	MKII
Plate Area/Rapper	2716 ft^2

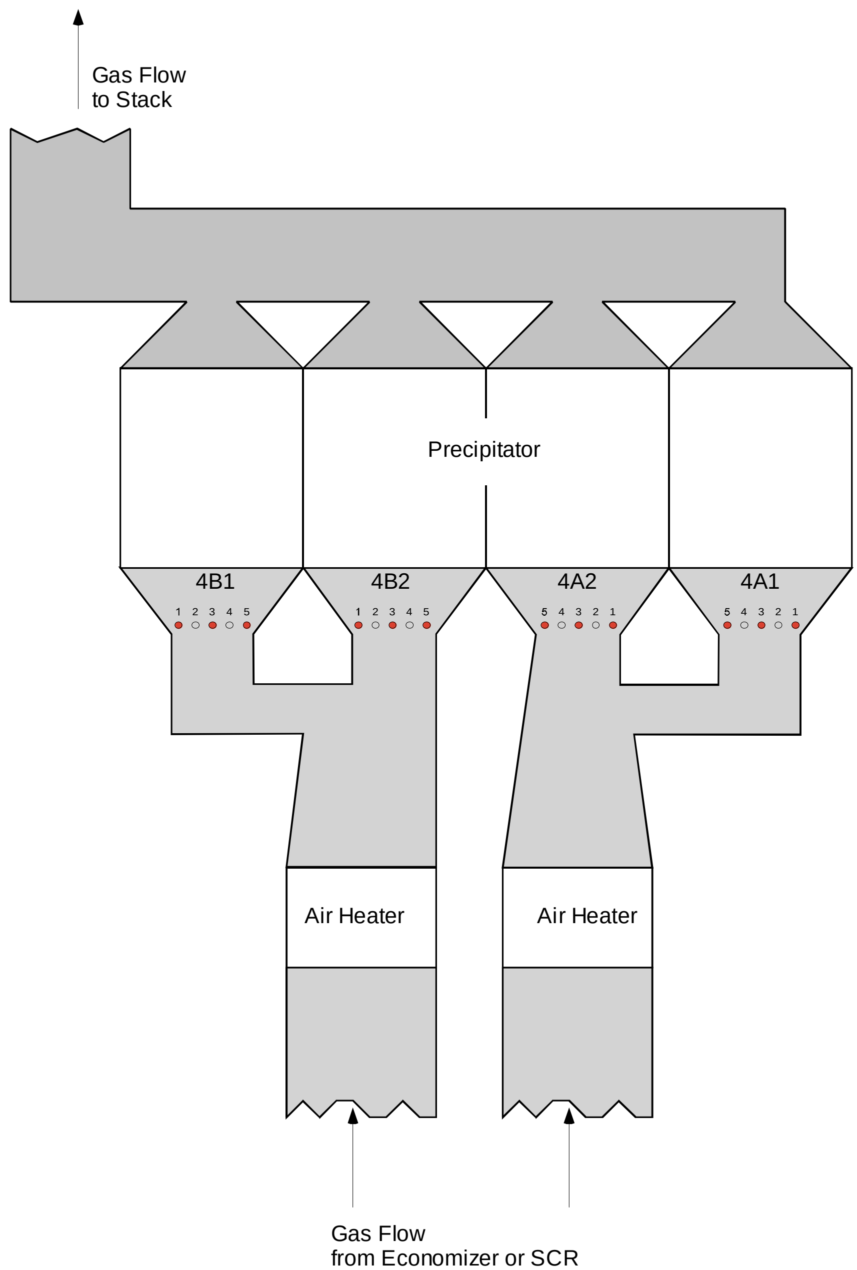


Figure 10-1 Precipitator Layout at Hammond 4

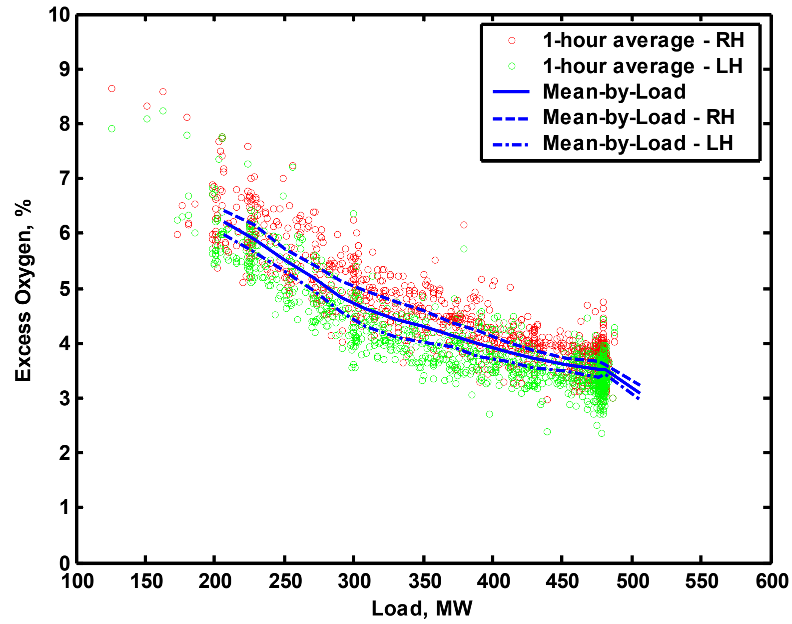


Figure 10-2 Excess Oxygen at Economizer Outlet vs. Load for 2002 / One-Hour Averages

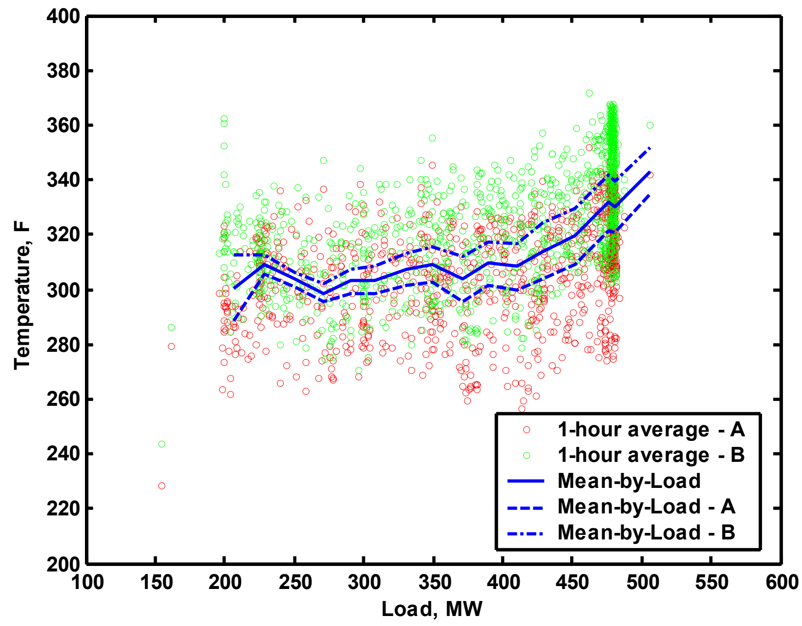


Figure 10-3 ESP Gas Inlet Temperature vs. Load for 2002 / One-Hour Averages

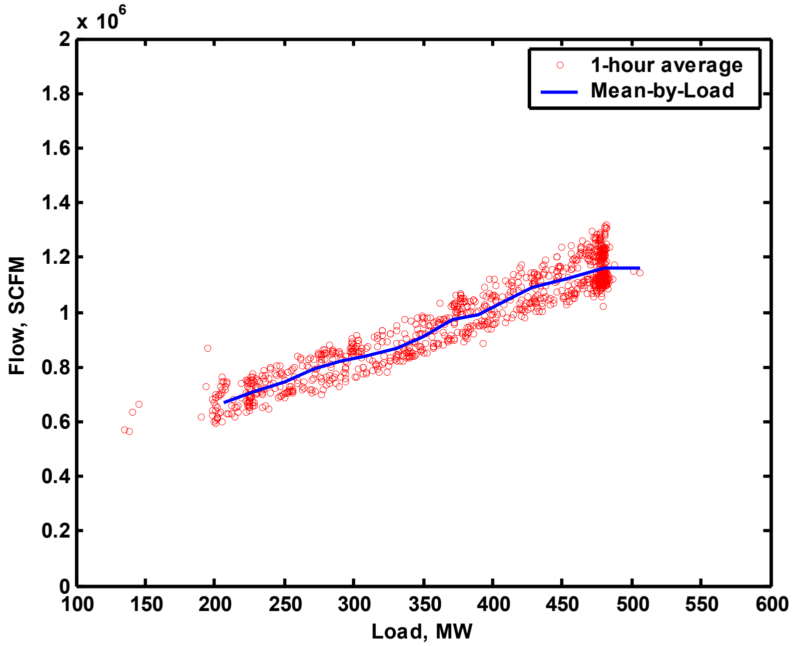


Figure 10-4 Stack Gas Flow (SCFM) vs. Load for 2002 / One-Hour Averages

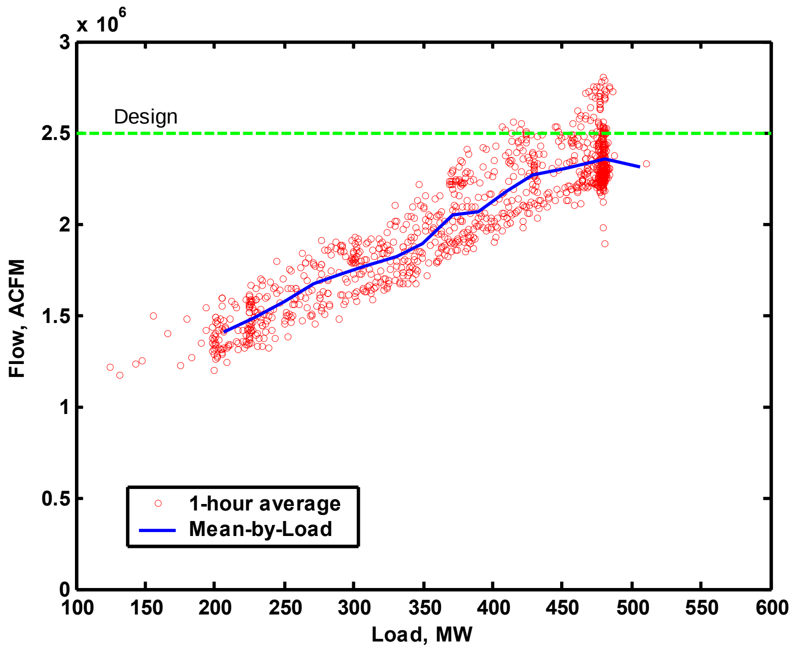


Figure 10-5 Stack Gas Flow (ACFM) vs. Load for 2002 / One-Hour Averages

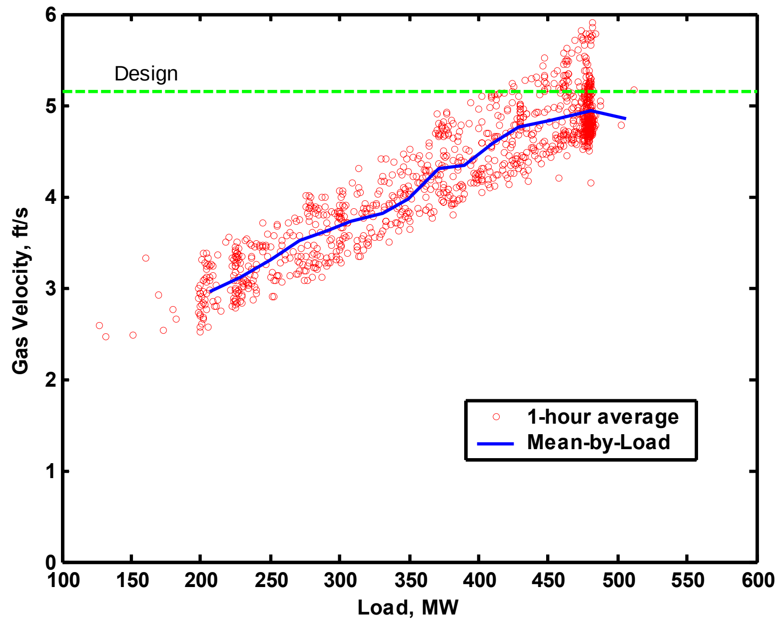


Figure 10-6 ESP Gas Inlet Gas Velocity vs. Load for 2002 / One-Hour Averages

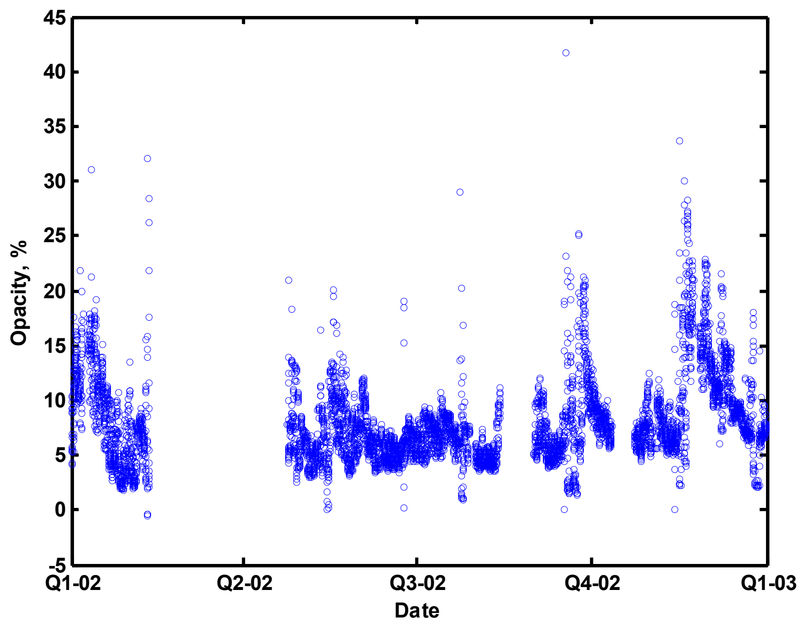


Figure 10-7 Opacity for 2002 / One-Hour Averages

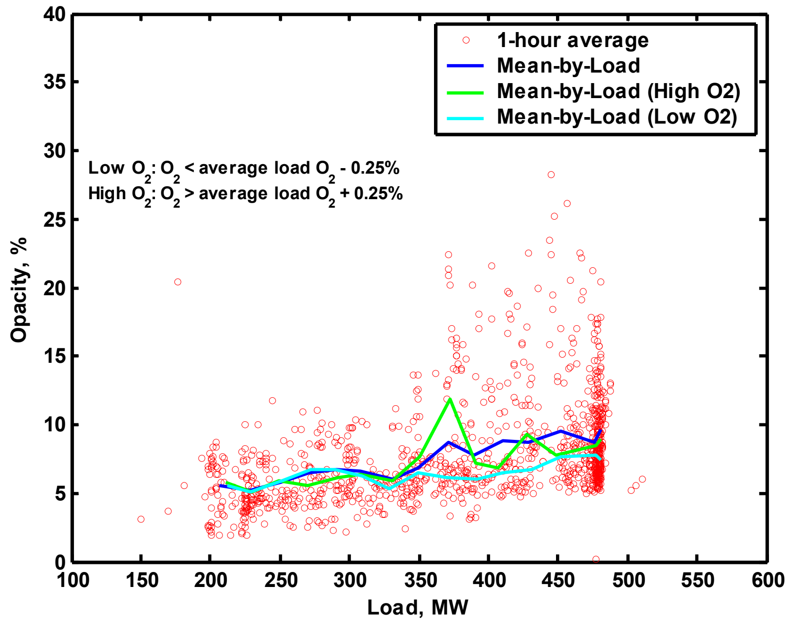


Figure 10-8 Opacity vs. Load for 2002 / One-Hour Averages

PCAMS

NWL's Precipitator Control & Management System (PCAMS) is a supervisory system used for remote control and data acquisition of the ESP [NWL00a][NWL00b]. The major features are:

- Can support multiple precipitators at one time
- Local and remote monitoring and control of transformers/rectifiers, rappers, and hoppers
- Trending and archiving of operating parameters
- Energy management system designed to reduce operating cost while maintaining opacity levels

PCAMS interfaces with the PLCs controlling the ESP. As mentioned, PCAMS has some limited capacity for optimizing ESP control parameters based on feedback from an opacity monitor and provides for two energy management modes. In the *Optimize Mode*, PCAMS continuously adjusts T/R power to maintain a balance between energy use and opacity. In the *Maintain Opacity Setpoint Mode*, PCAMS will adjust power to maintain opacity at a given level.[1] When utilizing energy management, the user has a choice of how to phase power by fields which may be defined within PCAMS. PCAMS can phase back power on all fields at once or it can cascade

[1] The Georgia Environmental Protection Division (EPD) requires the site to run the ESP to minimize opacity and particulate emissions and therefore the energy management features of PCAMS are not used at this site.

the power reductions among the fields, reducing the likelihood of sparking and opacity spikes.

ESPert

ESPert is an ESP monitoring and troubleshooting program that continuously receives and interprets data from the ESP control system, CEM system, and boiler controls [EPR94]. The program continuously estimates ESP performance, including opacity, based on these inputs and diagnoses the probable causes of any divergence between measured and predicted opacity. The core model used for the basic performance calculations is the Southern Research Institute ESP performance model whose development was funded by the US Environmental Protection Agency [FD84].

Although ESPert provides ESP performance estimates that can be compared with test results, its primary intended use is as an aid for plant staff to diagnose ESP operational, mechanical, and electrical problems. At least for the purposes of this project, perhaps a more important feature of the tool is that it allows for "*what-if*" analyses where operational scenarios may be tested before being actually implemented in the plant. The user interface to ESPert is shown in Figure 10-9 and Figure 10-10. As shown, ESPert provides some capability for trending and archival of data.

ESPert requires considerable plant and ESP data to effectively model the performance of the ESP and predict the outlet conditions. A summary of the required parameters is provided in Table 10-2. As can be inferred from this table, the effort involved in setting up ESPert is considerable even if the information were readily available.

Table 10-2 Summary of ESPert Data Requirements

Operating Data
Coal Properties (up to nine coals) – analysis
Ash properties – Ash mineral analysis Electrical data for each T/R set Volts, amps, sparking, arcing, T/R status
Boiler / Opacity data
Load, heat rate, opacity, flue gas conditioning, ESP gas inlet temperature, soot blowing
Dust cleaning Rapping cycles, hopper evacuation
Test data
Inlet and outlet ash loading, particle size, gas sneakage, gas flow, water, oxygen, pressure, resistivity, ESP efficiency
Configuration data Boiler data heat rate, additives, number of sootblowers, gas recirculation, burner type ESP design data Manufacturer, number of fields and gas paths, plate height, ESP pressure, ESP type, passage width, emissions, efficiency, Field data Field length, ESP voltage, T/R sets in field, T/R configuration, primary voltage, primary current, wave form Duct layout Layout, flow per duct Ash cleaning Rapper types, rows, number of rappers/row/gas path Ash removal Number of hoppers, removal periods

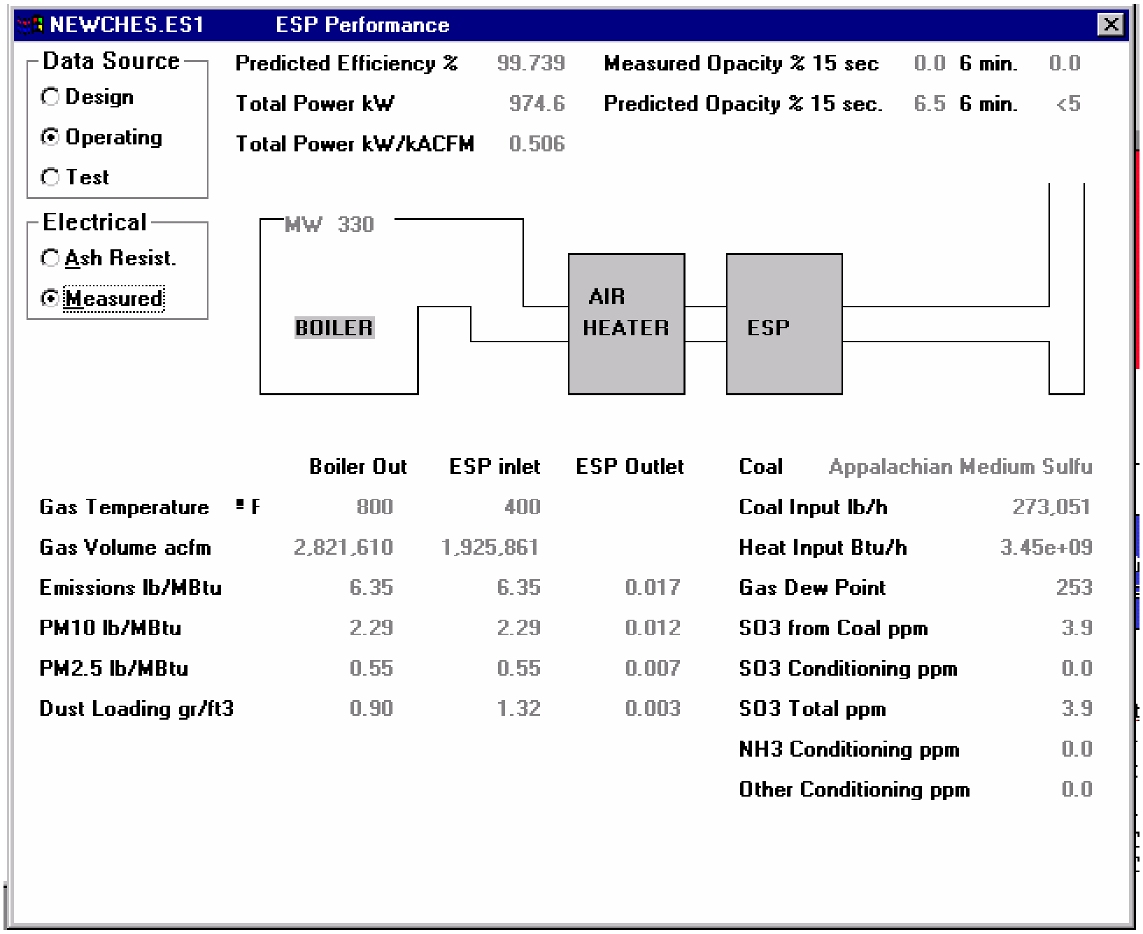


Figure 10-9 ESPert Interface – Main Panel

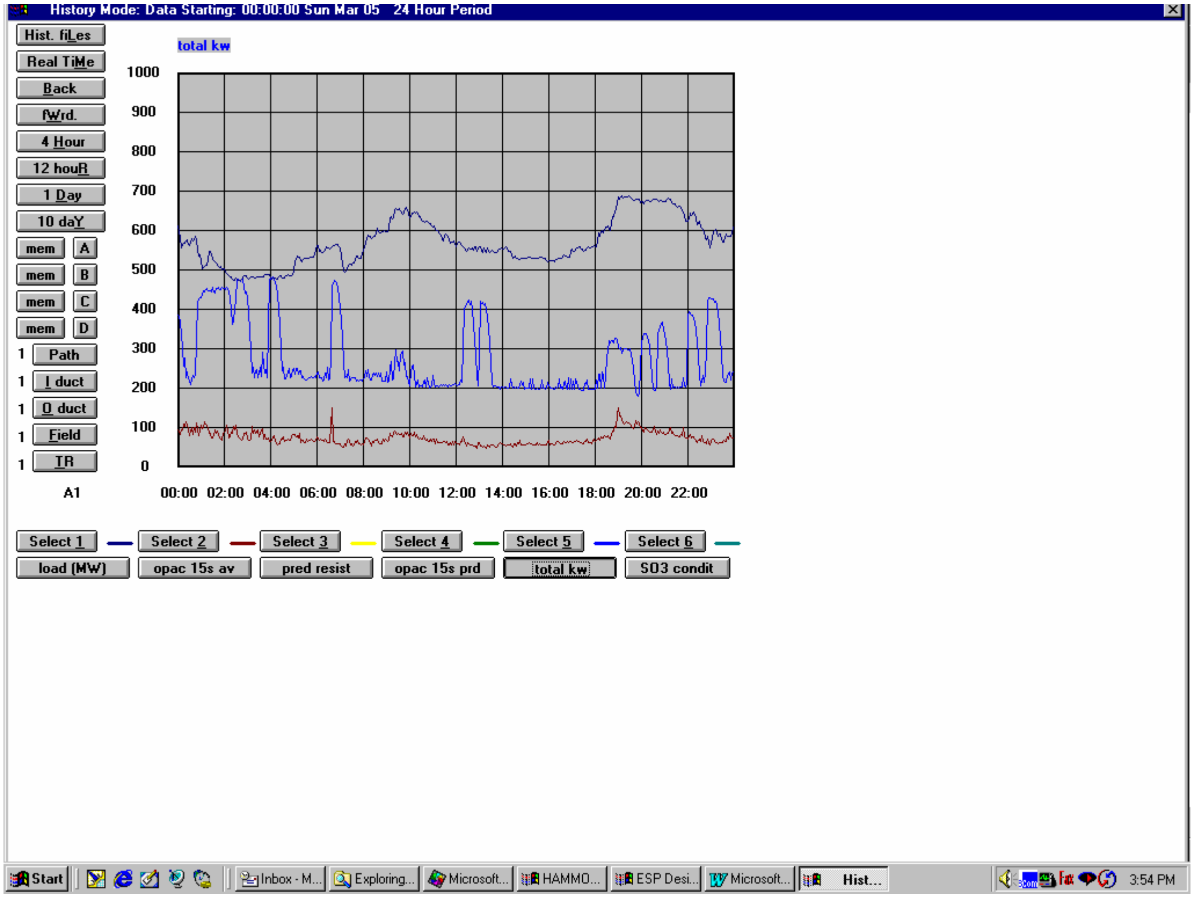


Figure 10-10 ESPert Interface – History Mode

Implementation at Hammond

An overview of ESPert and PCAMS installation is shown in Figure 10-11. ESPert was installed as part of this project whereas PCAMS was installed as part of other work ongoing at Hammond. In order to operate as designed, ESPert requires detailed information concerning the current status of the ESP which, as at Hammond, is typically only available in the standalone control system for the PLC and not in the distributed control system. For the purposes of the project at Hammond, PCAMS serves as a gateway to the detailed ESP operational data. Although not installed as part of this project, EPRI funded modifications to PCAMS to support the project at Hammond, designing and coding an interface from PCAMS to ESPert.

ESPert operates in either a manual or automatic data mode. In the manual mode, all operating data is entered manually into data fields on the data entry screens. Configuration data is stored in files located on local or remote PCs. In the automatic mode, ESPert obtains plant and ESP operating data by shared use of the file *espert.buf*. At Hammond, this information resides in two systems with the plant process information in the DCS and precipitator information within PCAMS. As shown, PCAMS writes to *espert.buf* on user defined intervals. Another program (*esp_rw*) then reads this file and adds process information obtained from the DCS (via the RTDS) and writes a file with the additional information to another location. The location of the files and update intervals are specified in the initialization file *esp_rw.ini* (Table 10-3). An interface to this program is provided to aid in debugging of the system and monitoring (Figure 10-12). The ESPert program reads the augmented *espert.buf* file at 15-second intervals. ESPert also reads event data, such as rapper events, through a separate file *espevent.dat*. This file is read by ESPert at one-second intervals. ESPert calculated results are written to the file *plantdat.dat*. This file is then read in by the program *esp_rw* and a subset of the data, defined by the file *planttag.ini*, is written to the RTDS. Only a small subset of the parameters calculated by ESPert are transferred to the RTDS (those with NULL are not transferred).

PCAMS and ESPert were configured by SCS with support from EPRI and plant staff. On October 27, 2000 the PCAMS/ESPERT system was made fully operational, collecting data from the precipitator and the DCS.

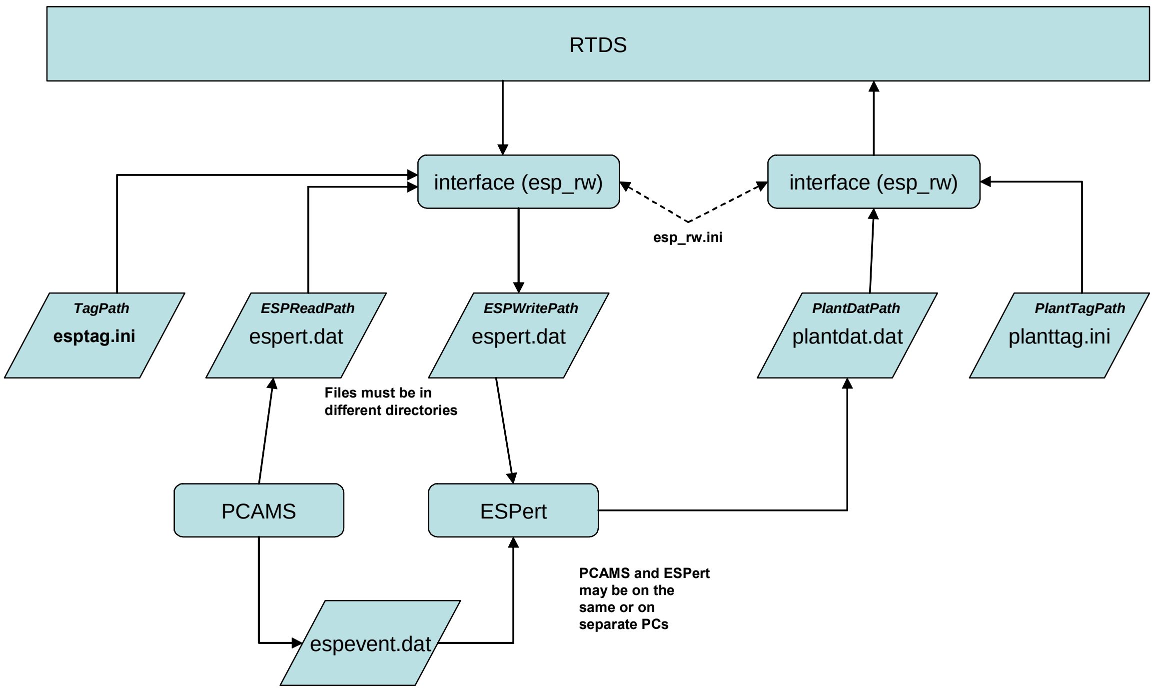


Figure 10-11 UOP / ESpert / PCAMS Operation at Hammond

Table 10-3 ESPert/RTDS Initialization File (esp_rw.ini)

[Settings]	
ESPReadPath=i:\ESPERT BUFFER\espert.buf	; ESPert file written by PCAMS (*espert.dat*)
ESPWritePath=i:\ESPert\espert.buf	; ESPert file modified by esp_rw (*espert.dat*)
TagPath=d:\RTDS\esptag.ini	; Tags to read in from RTDS to build *espert.dat*
PlantDatPath=i:\ESPert\plantdat.dat	; Output file of ESPert program (*plantdat.dat*)
PlantTagPath=d:\RTDS\planttag.ini	; Tags to write from *plantdat.dat* to RTDS
ScanInterval=30	; Scan interval in seconds
DebugLevel=5	; Set debug level (0=no messages)

Table 10-4 ESPert Parameters (esptag.ini)

ESPert Parameter	Source	Note
BoilerLoad4	CP001_I:MAIN_4.PNT_3	Gross Load, MW
HeatRate	DIRECT:NET_HEATRATE	Heat rate calculated by RTDS package
HeatInput	NOT USED	
LOI	4CP003_I:MAIN_4.PNT_6	
CoalFdrRate1	4TTLFUELFLW:TTLFF_LOGIC2.RO04	Total coal flow, lb/hr
CoalFdrRate2	NOT USED	
GasFdrRate	NOT USED	
InDuctDustLd	NOT USED	
OutDuctDustLd	NOT USED	
SO3Rate	NOT USED	
SO3Cond	NOT USED	
NH3Rate	NOT USED	
NH3Cond	NOT USED	
OthRate	NOT USED	
OthCond	NOT USED	
StackGasVol	NOT USED	
StackGasTmp	4AIRHEATER:ASEC_AIRHTR.RO01 and 4AIRHEATER:BSEC_AIRHTR.RO01	Average of secondary air heater gas outlet temperatures
O2	NOT USED	
SO2	NOT USED	
CO2	4CP001_I:MAIN_10.PNT_1	Stack CEM CO_2
CO	NOT USED	
OpacityInst	4CP001_I:MAIN_10.PNT_4	Stack instantaneous opacity
Opacity6min	NOT USED	
DuctOpacity0	NOT USED	
DuctOpacity1	NOT USED	
DuctOpacity2	NOT USED	
DuctOpacity3	NOT USED	
DuctTemp0	4AIRHEATER:ASEC_AIRHTR.RO01	Secondary air heater gas out temp (A)
DuctTemp1	4AIRHEATER:ASEC_AIRHTR.RO01	Secondary air heater gas out temp (A)
DuctTemp2	4AIRHEATER:BSEC_AIRHTR.RO01	Secondary air heater gas out temp (B)
DuctTemp3	4AIRHEATER:BSEC_AIRHTR.RO01	Secondary air heater gas out temp (B)
DuctTemp4	NOT USED	
DuctTemp5	NOT USED	
DuctTemp6	NOT USED	
DuctTemp7	NOT USED	
TotalPower	NOT USED	

Table 10-5 ESPert Parameters (planttag.ini)

Local Tag	RTDS Tag	Local Tag	RTDS Tag
dttm	NULL	dcttemp0	NULL
espcode	NULL	dcttemp1	NULL
coaln	NULL	dcttemp2	NULL
fTR0	NULL	dcttemp3	NULL
fTR1	NULL	dcttemp4	NULL
fTR2	NULL	dcttemp5	NULL
fTR3	NULL	dcttemp6	NULL
fTR4	NULL	dcttemp7	NULL
fTR5	NULL	odctopac0	NULL
fTR6	NULL	odctopac1	NULL
fTR7	NULL	odctopac2	NULL
load	NULL	odctopac3	NULL
heatrate	NULL	odctopac4	NULL
heatinpt	NULL	odctopac5	NULL
loi	NULL	odctopac6	NULL
coal1fd	NULL	odctopac7	NULL
coal2fd	NULL	aciddpt	NULL
gasfd	NULL	totalkw	TOTALKW
indctload	NULL	predRes	PREDRES
outdctload	OUTDCTLOAD	evntrapper0	NULL
so3cond	NULL	evntrapper1	NULL
so3total	NULL	evntrapper2	NULL
nh3gascon	NULL	evntrapper3	NULL
sgvcomb	NULL	evntrapper4	NULL
sgvcem	NULL	evntrapper5	NULL
sgtcem	NULL	evntrapper6	NULL
o2comb	NULL	evntrapper7	NULL
so2comb	NULL	evntsoot	NULL
co2comb	NULL	evnthopper	NULL
h2ocomb	NULL	SecVoltsAmps00	NULL
o2cem	NULL	SecVoltsAmps01	NULL
so2cem	NULL	SecVoltsAmps02	NULL
co2cem	NULL	SecVoltsAmps03	NULL
cocem	NULL	SecVoltsAmps04	NULL
opac15max	NULL	SecVoltsAmps05	NULL
opac15ave	NULL	SecVoltsAmps06	NULL
opac6m	NULL	SecVoltsAmps07	NULL
opac15prd	OPAC15PRD	SecVoltsAmps08	NULL
opac6mprd	OPAC6MRD	SecVoltsAmps09	NULL
effesp	EFFESP	SecVoltsAmps10	NULL
effgp0	NULL	SecVoltsAmps11	NULL
effgp1	NULL	SecVoltsAmps12	NULL
effgp2	NULL	SecVoltsAmps13	NULL
effgp3	NULL	SecVoltsAmps14	NULL
effgp4	NULL	SecVoltsAmps15	NULL
effgp5	NULL	SecVoltsAmps16	NULL
effgp6	NULL	SecVoltsAmps17	NULL
effgp7	NULL	SecVoltsAmps18	NULL
opacgp0	NULL	SecVoltsAmps19	NULL
opacgp1	NULL	SecVoltsAmps20	NULL
opacgp2	NULL	SecVoltsAmps21	NULL
opacgp3	NULL	SecVoltsAmps22	NULL
opacgp4	NULL	SecVoltsAmps23	NULL
opacgp5	NULL	SecVoltsAmps24	NULL
opacgp6	NULL	SecVoltsAmps25	NULL
opacgp7	NULL	SecVoltsAmps26	NULL
		SecVoltsAmps27	NULL

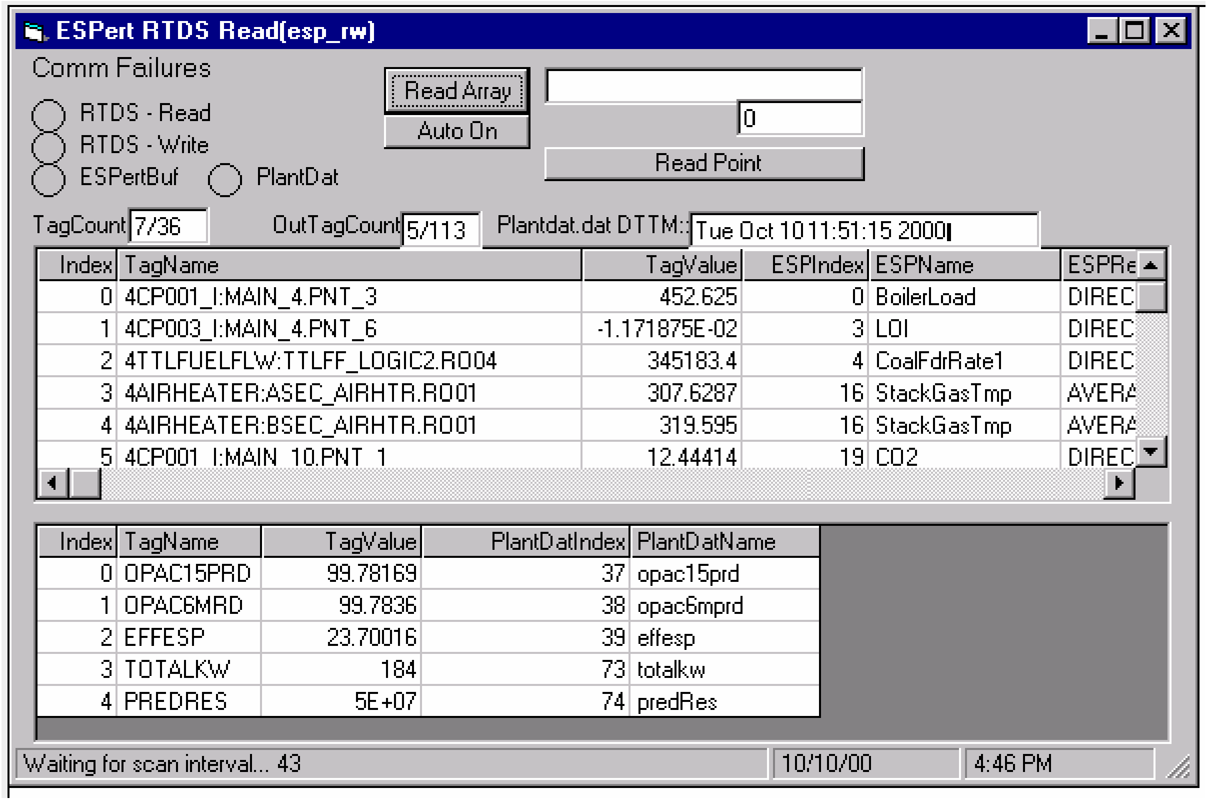


Figure 10-12 ESPert / RTDS Interface (esp_rw)

Performance

ESPert was integrated into the system and became first operational during October 2000. Since then, the joint system (PCAMS, ESPert, RTDS, etc) has been available only a small part of the actual operating time of the plant. The following paragraphs document the performance of the system.

Performance During 2000

During 2000, the ESP package operated approximately 274 hours (Figure 10-13).[1] Most of the hours off-line can be attributed to software development during this period. Though the predicted opacity levels were generally below the actual measured values during the October/November period (Figure 10-14 and Figure 10-15), the package was able to predict increases in opacity. Although not easily verifiable, predicted ESP efficiency is within the range to be expected (Figure 10-16 and Figure 10-17), above the 99.65% design with efficiency decreasing with increasing load. Predicted power averaged approximately 230 kW, again within design expectations (Figure 10-18 and Figure 10-19). Predicted ESP power was not highly influenced by load.

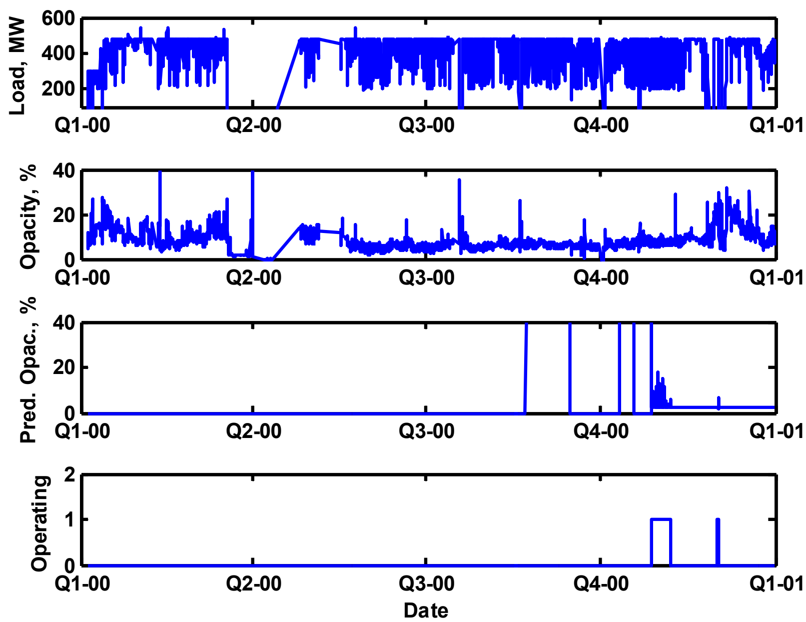


Figure 10-13 ESP Package – Operating Hours for 2000

[1] Operating hours were estimated by assuming that the system was not operational when predicted opacity: (1) was the same for two consecutive hour intervals, (2) less than 1%, or (3) greater than 30%.

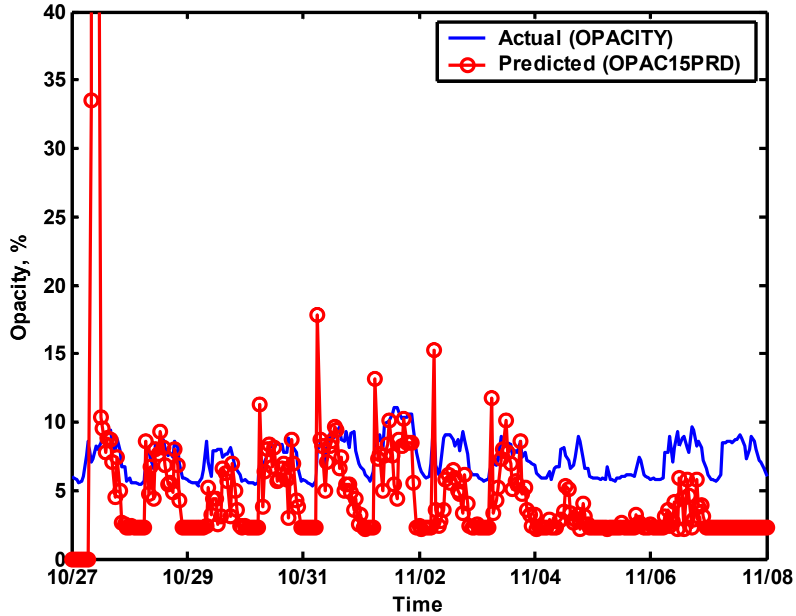


Figure 10-14 ESP Package – Actual and Predicted Opacity for Oct 27 – Nov 7, 2000

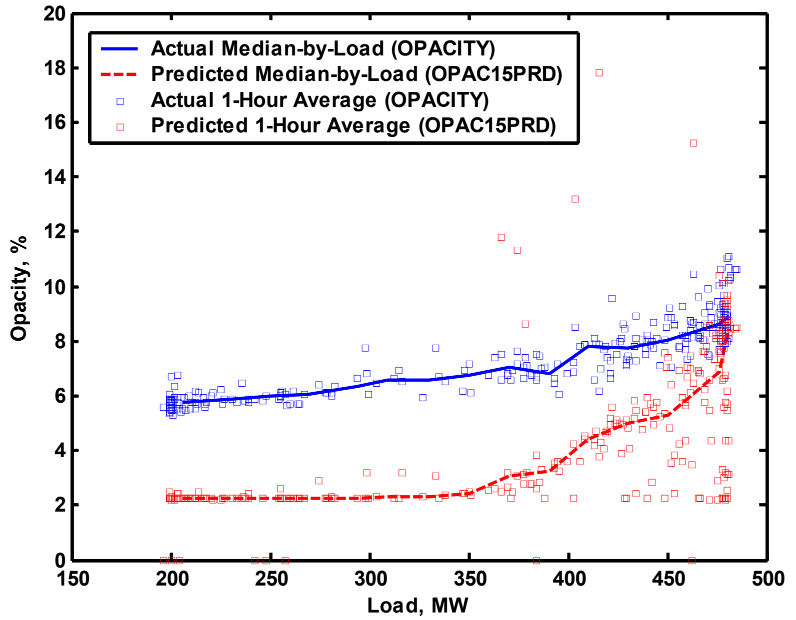


Figure 10-15 ESP Package – Actual and Predicted Opacity for Oct 27 – Nov 7, 2000

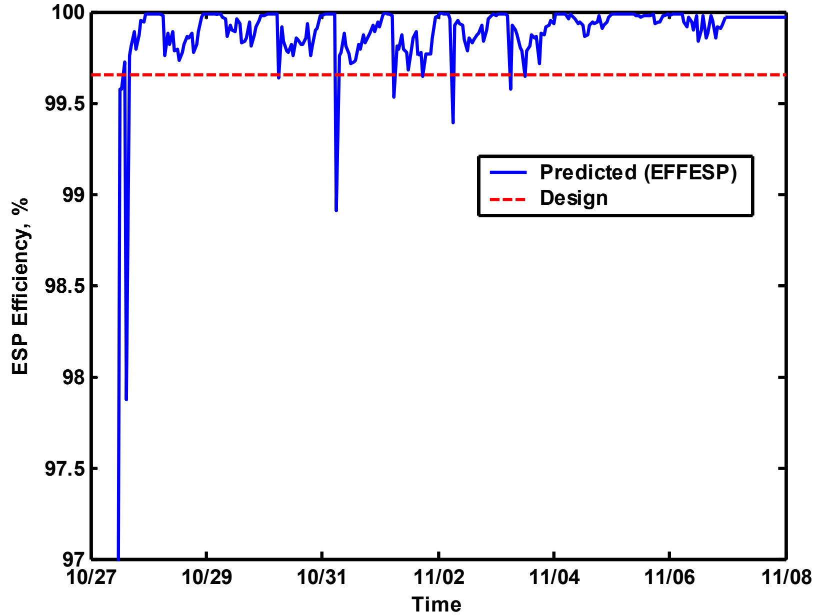


Figure 10-16 ESP Package – Predicted ESP Efficiency for Oct 27 – Nov 7, 2000

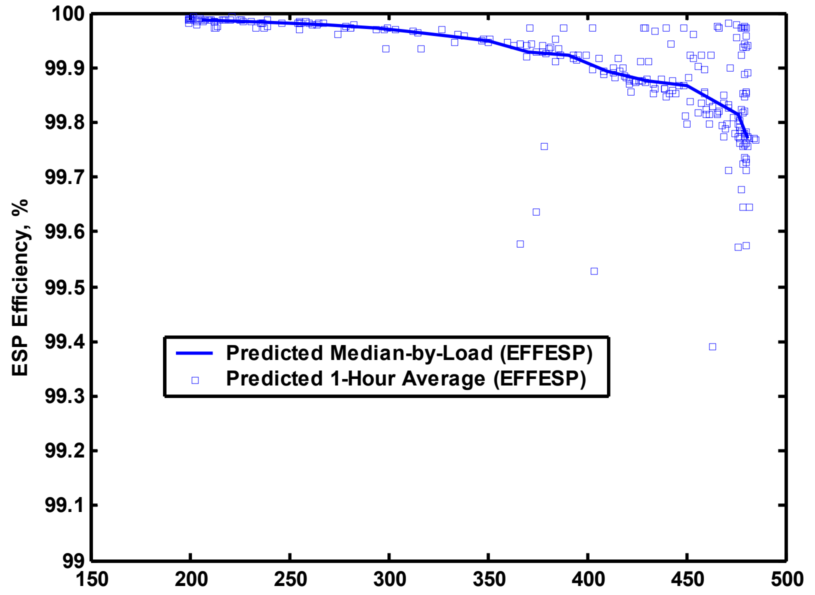


Figure 10-17 ESP Package – Predicted ESP Efficiency vs. Load for Oct 27 – Nov 7, 2000

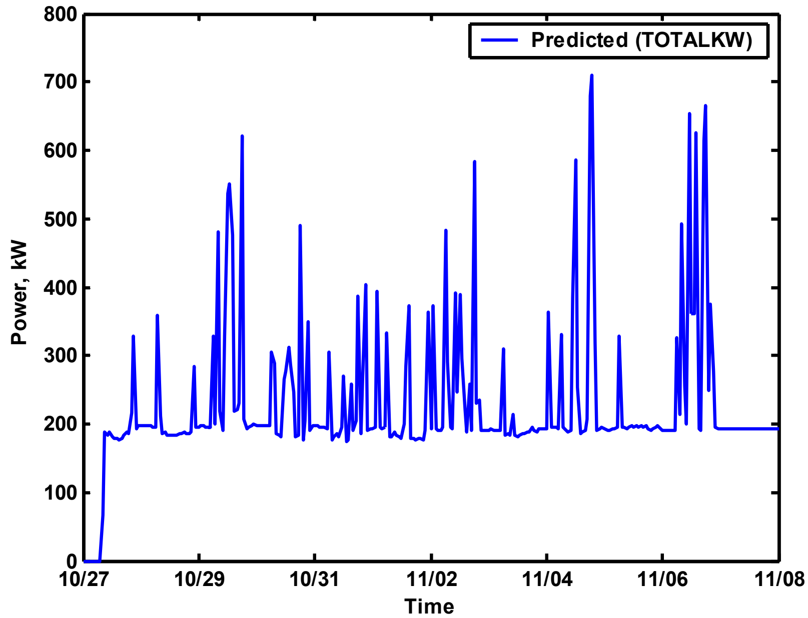


Figure 10-18 ESP Package – Predicted ESP Power for Oct 27 – Nov 7, 2000

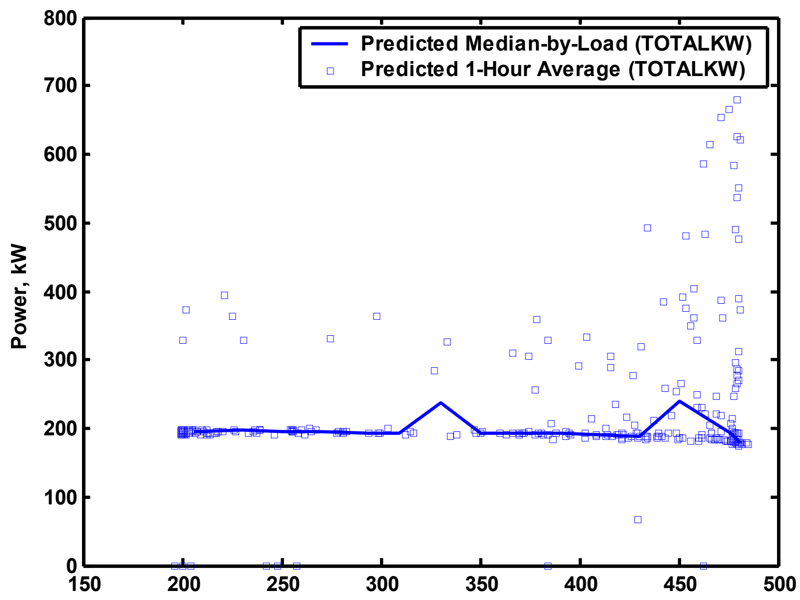


Figure 10-19 ESP Package – Predicted ESP Power vs. Load for Oct 27 – Nov 7, 2000

Performance During 2001

During 2001, the ESP package operated approximately 341 hours, slightly more than it had in 2000 (Figure 10-20). As in 2000, many of the hours off-line can be attributed to software development during this period but PCAMS was also not available during much of this period. Predicted opacity levels were again generally below the actual measured values and exhibited much more variability than the measured values (Figure 10-21 and Figure 10-22). The predicted value was also much more dependent on load than the measured value. As before, predicted ESP efficiency was within the range to be expected, above the 99.65% design with efficiency decreasing with increasing load (Figure 10-23 and Figure 10-24). Predicted power averaged approximately 303 kW, again within design expectations but considerably above (a 50% increase) the mean in 2000 (Figure 10-25 and Figure 10-26). During this period, predicted ESP power appeared to be influenced by the load, but this may be an artifact of having a low data count for the lower loads.

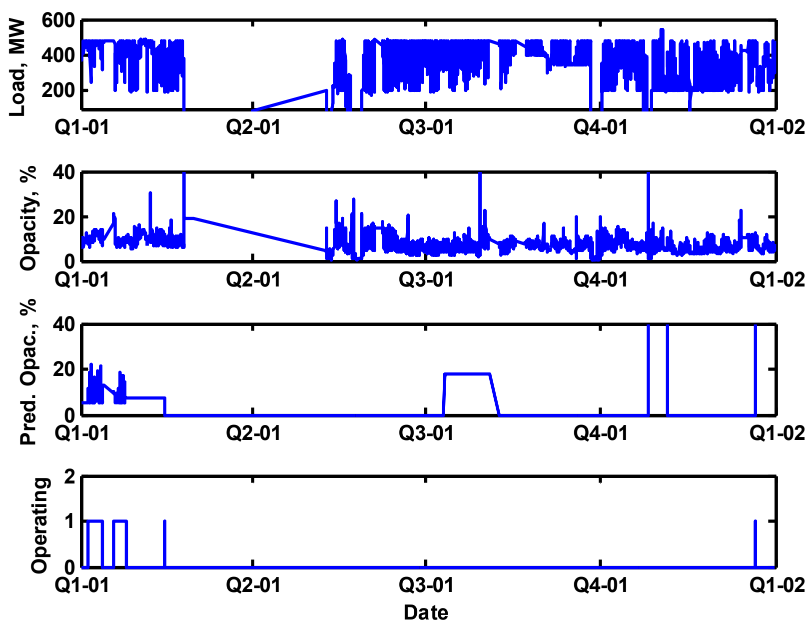


Figure 10-20 ESP Package – Operating Hours for 2001

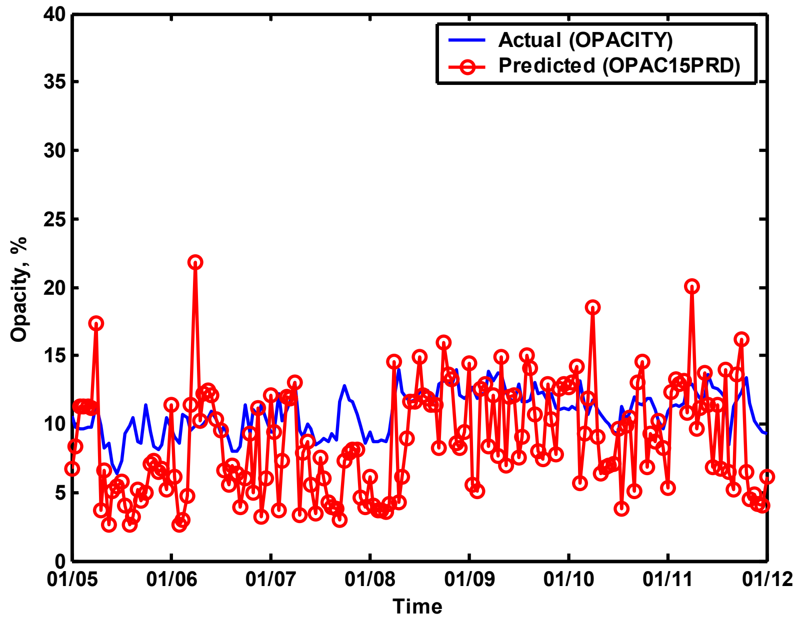


Figure 10-21 ESP Package – Actual and Predicted Opacity for Jan 5 – Jan 11, 2001

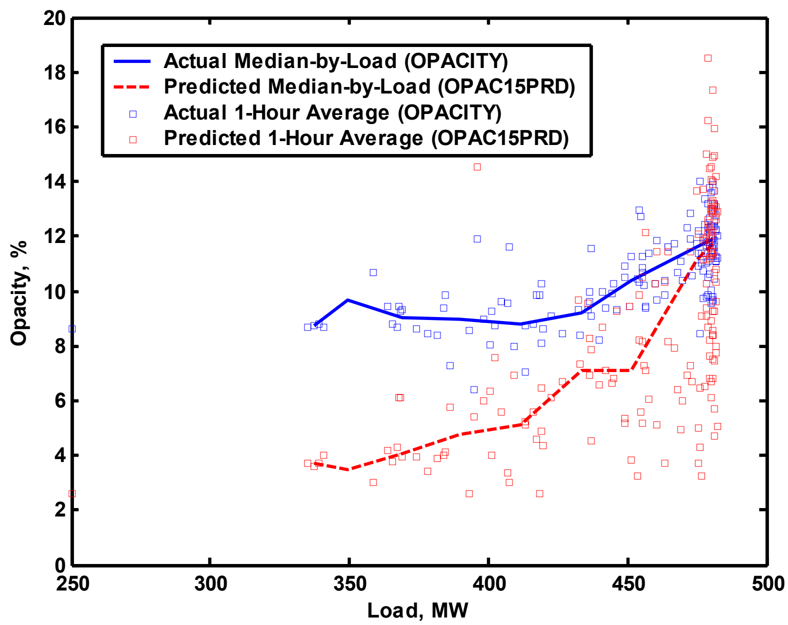


Figure 10-22 ESP Package – Actual and Predicted Opacity for Jan 5 – Jan 11, 2001

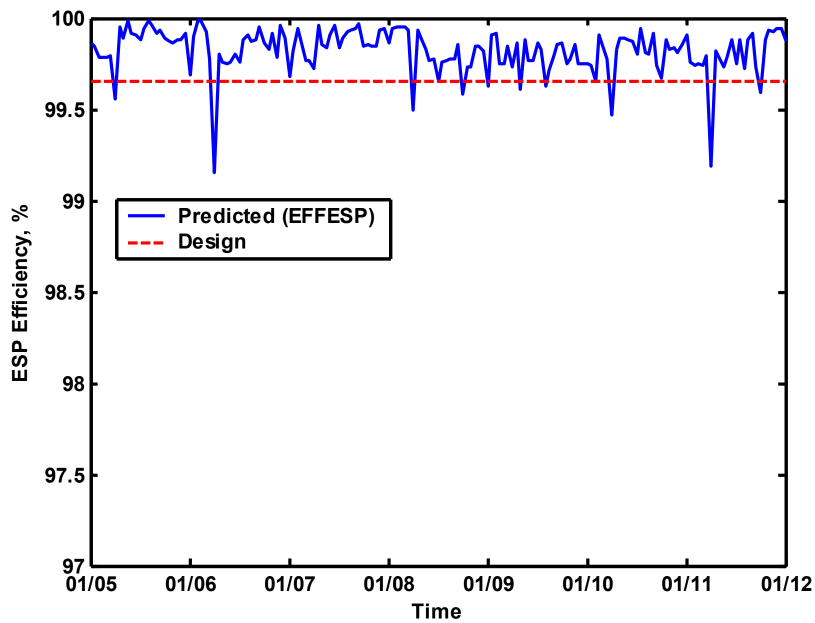


Figure 10-23 ESP Package – Predicted ESP Efficiency for Jan 5 – Jan 11, 2001

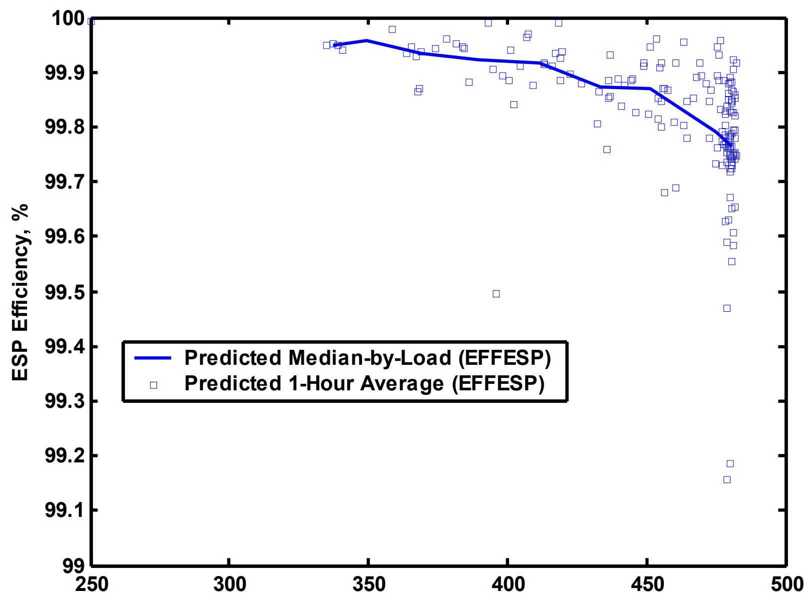


Figure 10-24 ESP Package – Predicted ESP Efficiency vs. Load for Jan 5 – Jan 11, 2001

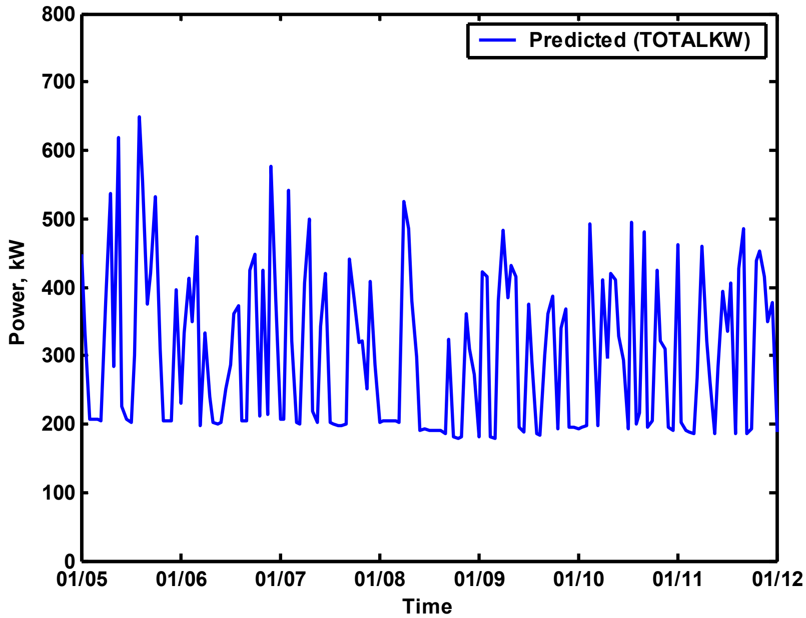


Figure 10-25 ESP Package – Predicted ESP Power for Jan 5 – Jan 11, 2001

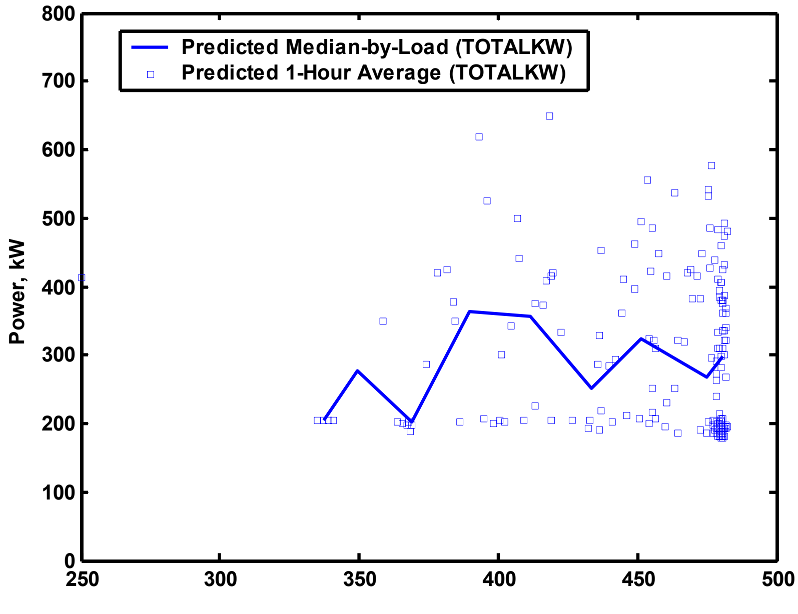


Figure 10-26 ESP Package – Predicted ESP Power vs. Load for Jan 5 – Jan 11, 2001

Performance During 2002

During 2002, the ESP package operated approximately 599 hours, considerably above the operating time in the two previous years (Figure 10-27). As before, many of the hours off-line can be attributed to software development. As opposed to earlier years, predicted opacity levels were much higher than the actual measured values and again exhibited much more variability than the measured values (Figure 10-28 and Figure 10-29). The predicted value was also much more dependent on load than the measured value. As may be expected because of the high predicted opacities, predicted ESP efficiency was generally below design, particularly at high loads (Figure 10-30 and Figure 10-31). Predicted power averaged approximately 235 kW (Figure 10-32 and Figure 10-33). During this period, predicted ESP power appeared to be influenced by the load, but this may be an artifact of having a low data count for the lower loads.

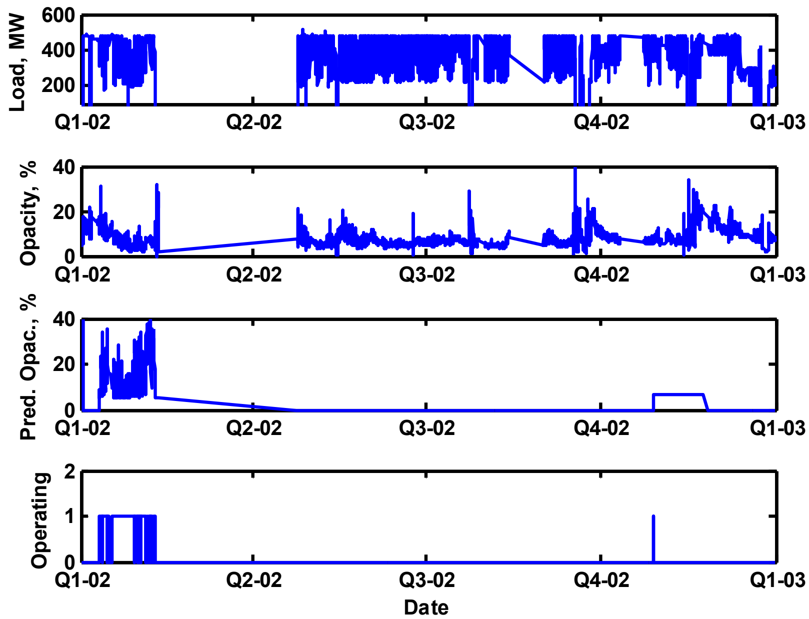


Figure 10-27 ESP Package – Operating Hours for 2002

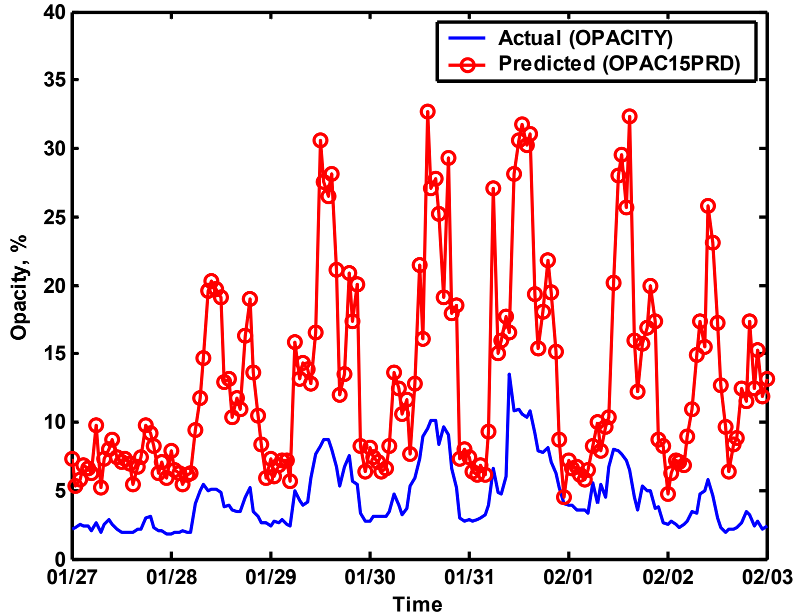


Figure 10-28 ESP Package – Actual and Predicted Opacity for Jan 27 – Feb 2, 2002

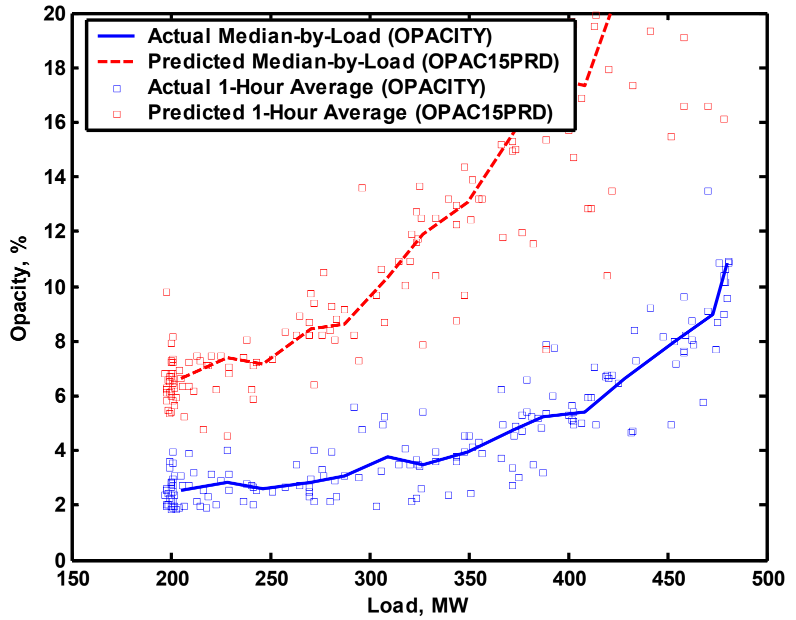


Figure 10-29 ESP Package – Actual and Predicted Opacity for Jan 27 – Feb 2, 2002

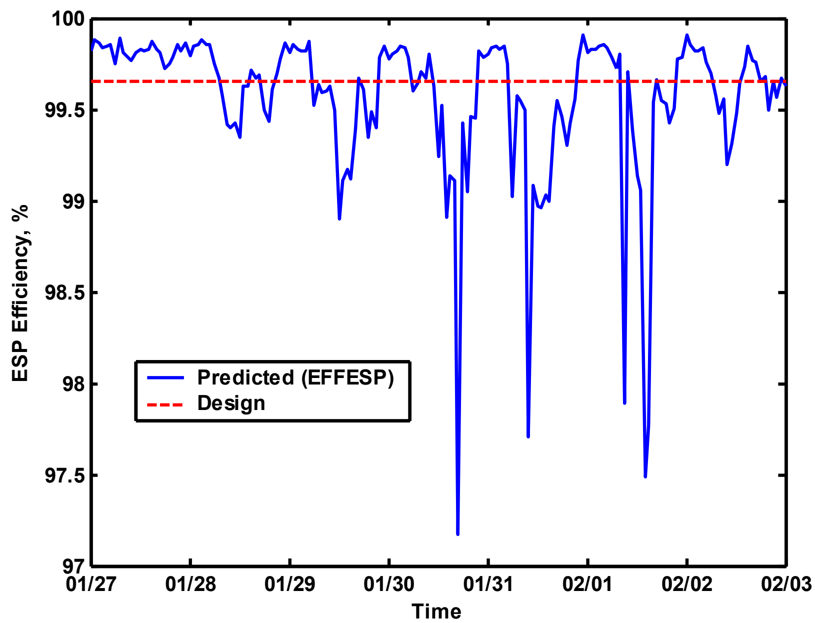


Figure 10-30 ESP Package – Predicted ESP Efficiency for Jan 27 – Feb 2, 2002

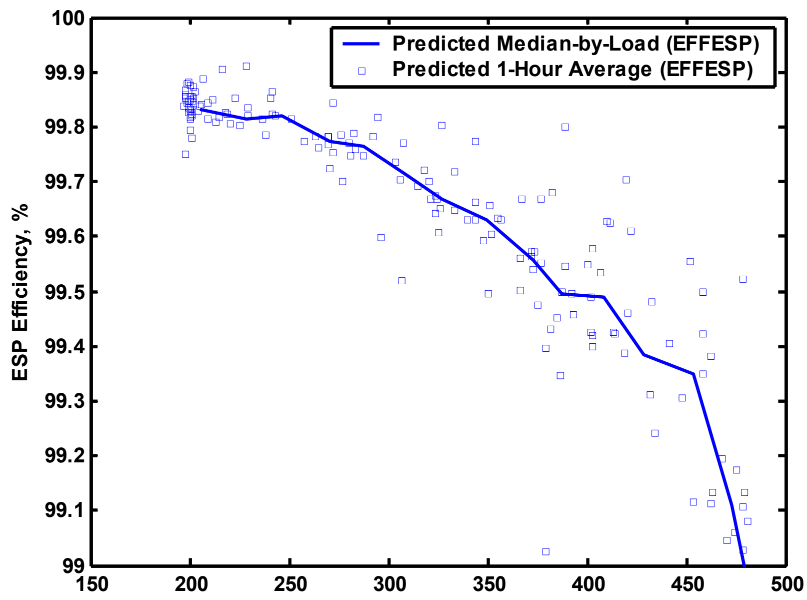


Figure 10-31 ESP Package – Predicted ESP Efficiency vs. Load for Jan 27 – Feb 2, 2002

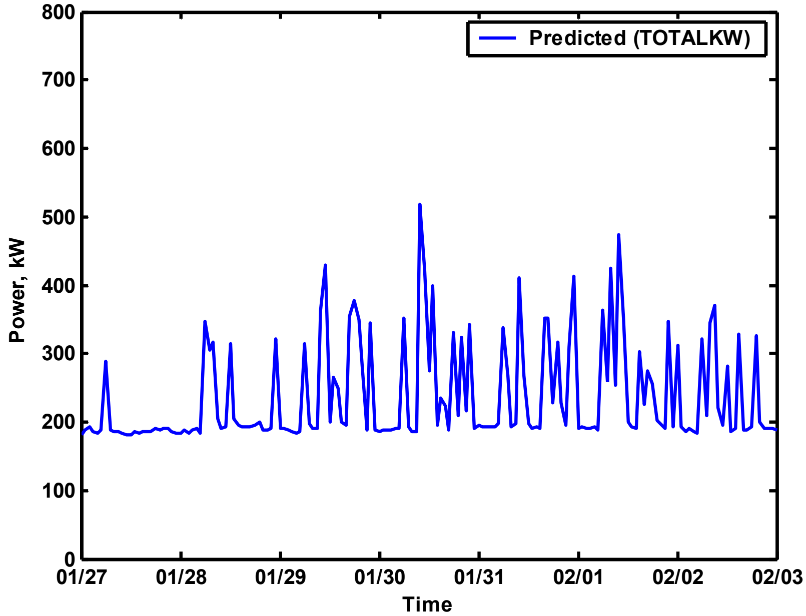


Figure 10-32 ESP Package – Predicted ESP Power for Jan 27 – Feb 2, 2002

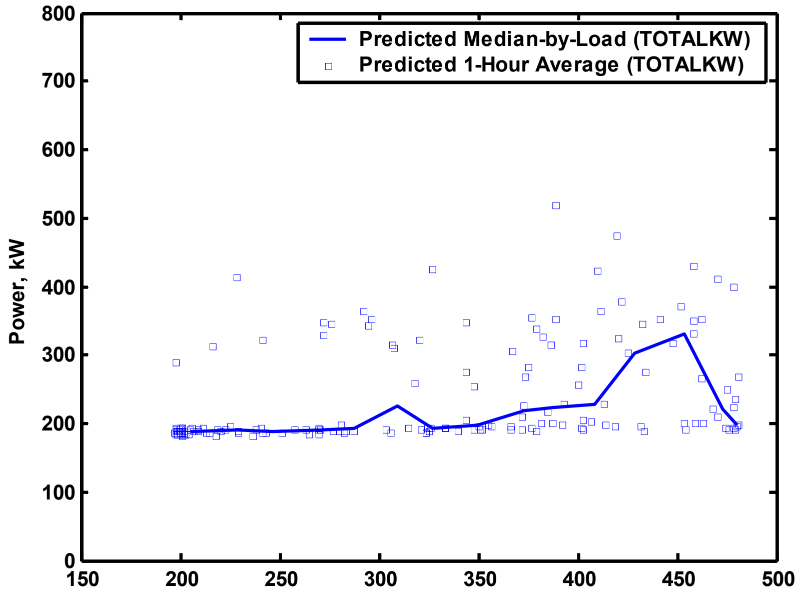


Figure 10-33 ESP Package – Predicted ESP Power vs. Load for Jan 27 – Feb 2, 2002

Summary

Precipitator performance such as measured by outlet opacity, particulate removal rate, and energy consumption is greatly dependent on precipitator inlet conditions. These conditions are in turn a function of boiler operating conditions and possibly other post-combustion emission control technologies (SNCR and SCR). Given the dependence of ESP performance on upstream operating conditions and importance of its operation on environmental performance, it was felt that the ESP should be brought into the optimization envelope.

The ESP package consists of a number of components, some of which were either developed or installed as part of this project and others which were not part of the scope of the project but provided the required infrastructure. The components can be categorized as follows:

- ESPert – EPRI's ESPert is an ESP monitoring and troubleshooting program. ESPert was configured and deployed as part of the project. SCS performed the configuration.
- PCAMS – NWL's Precipitator Control & Management System (PCAMS) is a supervisory system used for remote control and data acquisition of the ESP. PCAMS was installed separately from the project but EPRI provided funding to support modification of the PCAMS software so that it could interface with ESPert.
- Supporting software – Several software components were developed, generally by SCS with project funding, to support data gathering and program interfaces.

Initial expectations were to use the ESP package as an optimization platform; however to date, it has been used only as an on-line predictive model (using ESPert). Since becoming first operational during October 2000, the package has operated approximately 1200 hours. Potential areas for further work on this package include:

- Further testing to confirm performance of the system
- Modifications to the software and operating procedures to make the system more robust to accumulate additional operating time
- Modification of the software to allow a user to run ESPert to perform "what-if" studies
- Interfacing of the package with the plant's recently installed plant information network
- Development of an interface to allow the ESPert models to be called from the unit optimization software

11
SUMMARY AND FURTHER WORK

Status and Results

The project goals were achieved with varying degrees of success. Specifically, due to delays resulting in several project extensions, there was insufficient plant testing to fully quantify the benefit of the technologies.

ESP Package – EPRI's ESPert was installed at the site as part of this project. The ESPert package, originally developed in the 1990s, is a diagnostic and predictive model for ESPs designed to evaluate and predict ESP performance and diagnose problems. ESPert interfaces with the PCAM system, a supervisory control system for the ESP. Initial expectations were to use the ESPert/PCAM software as an optimization platform; however to date, it has been used only as a predictive model.

GNOCIS/Boiler – GNOCIS is a real-time, closed-loop system for performing boiler optimization. GNOCIS was first installed at Hammond 4 in 1996 and was upgraded as part of this current project. A major improvement was the development and incorporation of on-line model error correction. This error correction greatly improves the accuracy and robustness of the neural-network combustion models. An operator interface exists on the DCS for this component and this system is capable of both open- or closed-loop operation. The current configuration makes recommendations on excess O_2, feeder coal flows, and overfire air for optimizing NOx emissions, boiler efficiency, and fly ash unburned carbon. Previous testing of GNOCIS at this site shows substantial benefits may be obtained by its application.

GNOCIS/Turbine – GNOCIS was adapted to be applied to steam cycle optimization. This package uses the same code base as that used by the GNOCIS/Boiler; however, a different model (for the turbine) is used. At present, this is an advisory system only, lacking the DCS configuration modifications required to be closed-loop. Also, the operator interface runs on a local or remote PC and not on the DCS. This system is configured to make recommendations on main steam and hot reheat temperatures and main steam pressure to optimize turbine cycle heat rate.

Intelligent Sootblowing System (ISBS) Package – The ISBS is an advisory system providing guidance on sootblower operation. Powergen developed the dynamic link library (DLL) implementing the fuzzy rule-base and SCS developed the interface and other supporting code. This package is a rule-based advisory system and not an optimizer as are the two GNOCIS packages. The user interface for the package runs on a PC, either local or remote, and not on the DCS. Brief testing of the technology indicated that the application would provide substantial benefits primarily in reducing sootblowing activity. The ISBS package is installed and available for operation at Hammond.

Real-Time Heat Rate Package - The Center for Electric Power at Tennessee Technological University developed a set of on-line unit heat rate and boiler performance calculations for the unit. SCS interfaced this package to the balance of the software system. The software was installed to provide more information concerning the real-time unit performance than previously available. Although not a primary goal of the project, plans are being made to compare the outputs of the program (heat rate, boiler efficiency, coal flow, coal higher heating value, and coal nitrogen content) to that generated by other methods.

Unit Optimization Package – The focus of this package was to develop a framework and software to coordinate multiple process optimizers. This package consists of several components including global optimizers and adaptations of the “package” optimizers (and sub-optimizers) to allow communication to the global optimizer. Although the framework and software will support other global optimizers, two were included in this scope. SCS adapted a Powergen developed proof-of-concept global optimization algorithm to fit within the framework. The other global optimizer incorporated was one developed by Synengco and marketed in the US by URS. Although functional, this software requires further testing to ensure that it is operating robustly and reliably.

Further Work

As of March 2003, other than the current project, the authors know of no other active attempts to apply coordinated optimization to power plants. As part of DOE’s recent Clean Coal Power Initiative, DOE has proposed to co-sponsor a project of similar scope with project completion in 2006. Given the possible great returns by the application of these technologies, additional work is planned including improvements to the software and further testing. Areas of work and improvement that may prove beneficial include:

- Further testing on both simulator and plant to fully quantify performance and emission benefits of applying the software
- Refinement of software components to improve robustness and flexibility

- Development of training and operating manuals for plant staff
- Improvement of user interfaces for operations personnel
- Migration of software to use the recently installed plant operations information system (ASPEN Technologies InfoPlus 21)
- Add capabilities for closed-loop operation on steam cycle optimization package
- Install the most recent version of ESPert and modify software to take advantage of limited optimization capabilities of ESPert and PCAMS
- Complete enhancement of ISBS package so that it may be operated as an optimizer and investigate potential closed-loop operation

Bibliography

[Ame76] American Society of Mechanical Engineers, New York, NY. *ASME Power Test Codes - Steam Turbines PTC 6*, 1976.

[Ame85] American Society of Mechanical Engineers, New York, NY. *ASME Power Test Codes - Test Code for Steam Generators PTC 4.1*, 1985.

[BBP97] M. Bangham, A. Brostrom, and B. Pitt. Total optimization process advisor (TOPAZ) — improving performance through optimization. In *EPRI/ESEERCO Optimization Workshop Proceedings, Report TR-108687*, Palo Alto, CA, April 1997. EPRI.

[Bis96] Christopher M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, Oxford, UK, January 1996.

[BPAL98] M. Bangham, J. Patton, H. Abeledo, and I. Liberatore. Intelligent controller for optimized sootblowing — phase II results. Technical Report Project Id: P/CH-FG02-95ER86036, US Department of Energy, 1998. Small-Business Technology Transfer (STTR) Project.

[CDFK03] David W. Corne, Kalyanmoy Deb, Peter J. Fleming, and Joshua D. Knowles. The good of the many outweighs the good of the one: Evolutionary multi-objective optimization. *Connections. The Newsletter of the IEEE Neural Networks Society*, 1(1):9–13, February 2003.

[CE98] M.A. Cremer and E.G. Eddings. 500 MW demonstration of advanced wall-fired combustion techniques for the reduction of nitrogen oxide (NOx) emissions from coal-fired boilers — CFD modeling of SNCR performance in Georgia Power Company's Hammond Unit 4 and Wansley Unit 1. Technical report, Southern Company Services, Birmingham, AL, 1998. Prepared by Reaction Engineering for Southern Company Services under contract to the US Department of Energy.

[Cha97] D. Chappell. *The Next Wave - Component Software Enters the Mainstream*. Chappell and Associate, April 1997.

[CWSL98] D. Cramer, S. Williams, N. Sarunac, and E. Levy. Application of boiler op to utility boilers: Field results. In *EPRI 1998 NOx Controls for Utility Boilers Workshop*, Palo Alto, CA, August 1998. EPRI. Baltimore, Maryland, August 25-27, 1998.

[EPA00] Code of federal regulations 40 protection of the environment part 75. US Environmental Protection Agency, July 2000.

[EPR94] ESPERT - electrostatic precipitator performance diagnostic model. Technical Report TR-104690, EPRI, Palo Alto, CA, 1994.

[EPR96] Integrated knowledge framework (IKF) for coal-fired power plants - an analysis of the data, information, and knowledge requirements for the economic operation and maintenance of coal-fired power plants volume 1: Summary. Technical Report TR-106211-V1, EPRI, Palo Alto, CA, 1996.

[EPR97a] Evaluation of heat rate discrepancy from continuous emission monitor systems. Technical Report TR-108110, EPRI, Palo Alto, CA, August 1997.

[EPR97b] Retrofit NOx controls for coal-fired boilers — 1996 update addendum. Technical Report TR-102906-Addendum, EPRI, Palo Alto, CA, 1997.

[EPR98a] Power plant optimization guidelines. Technical Report TR-110718, EPRI, Palo Alto, CA, 1998.

[EPR98b] Total plant optimization. Technical Report MI-111573, EPRI, Palo Alto, CA, 1998.

[EPR98c] Workshop on intelligent sootblowing control. Proceedings TR-111631, EPRI, Palo Alto, CA, 1998.

[EPR99] Demonstration of an advanced sootblowing control system case study: Application of intelligent sootblowing at powergen's kingsnorth power station. Technical Report TR-114420, EPRI, Palo Alto, CA, 1999. Prepared by Powergen plc under contract to EPRI.

[EPR02] Power plant optimization – overview of industry experience. EPRI, Palo Alto, CA, 2002. EPRI web page.

[EPR] Proceedings: Second annual EPRI workshop on power plant optimization. Proceedings TR-111316, EPRI, Palo Alto, CA.

[FD84] M. Faulkner and J. DuBard. A mathematical model of electrostatic precipitation (revision 3). Technical Report EPA-600/7-84-069a, b, c, US Environmental Protection Agency, Research Triangle Park, NC, 1984. Volume I, Modeling and Programming: NTIS PB84-212-679; Volume II, User's Manual: NTIS PB84-212-687; FORTRAN Source Code Tape: NTIS PB84-232-990.

[Gen98] G2 strategic intelligent systems for operations management. Gensym Corporation, Cambridge, MA, 1998. Sales literature.

[GMTS89] M. Gadiraju, S. Munukutla, G. Tsatsaronis, and Ora Scott. Steady state performance simulation model for j. m. stuart station unit 2. In *Proceedings of the 1989 ASME Joint Power Generation Conference*, New York, NY, October 1989. American Society of Mechanical Engineers. Paper No. 89-JPGC-PWR-22.

[LMB+86] E. K. Levy, S. Munukutla, O. Badr, S. Williams, and J. Fernandes. Optimization of unit heat rate through variations in fireside parameters. In *Proceedings of the 1986 EPRI Power Plant Performance Monitoring and System Dispatch Improvement Workshop*, Palo Alto, CA, September 1986. EPRI.

[LMJ+84] E. K. Levy, S. Munukutla, A. Jibilian, H. G. Crim, J. Cogoli, A. F. Kwasnik, and F. Wong. Analysis of the effects of coal fineness, excess air and exit gas temperature on the heat rate of a coal-fired power plant. In *Proceedings of the 1984 ASME Joint Power*

Generation Conference, New York, NY, October 1984. American Society of Mechanical Engineers. Paper No. 84-JPGC-PWR-1.

[LSC+87] E. Levy, N. Sarnac, H. G. Crim, R. Leyse, and J. Lamont. Output/loss: A new method for measuring unit heat rate. In *Proceedings of the 1987 ASME Joint Power Generation Conference*, New York, NY, October 1987. American Society of Mechanical Engineers. Paper No. 87-JPGC-PWR-39.

[MA03] R. Marler and J. Arora. Review of multi-objective optimization concepts and algorithms for engineering. Technical report, Optimal Design Laboratory, College of Engineering, University of Iowa, Iowa City, IA, January 2003.

[Mak98] J. Makansi. Plants gain confidence in optimization software. *Power Magazine*, 142(5), September/October 1998.

[Mat02a] The Mathworks Inc., Natick, Massachusetts. *MATLAB - Fuzzy Logic Toolbox Users Guide*, July 2002. Version 2.1.2.

[Mat02b] The Mathworks Inc., Natick, Massachusetts. *MATLAB - Neural Network Toolbox Users Guide*, July 2002. Version 4.

[Mat02c] The Mathworks Inc., Natick, Massachusetts. *MATLAB - Using MATLAB*, August 2002.

[May01] Ian Mayes. Unit optimization. Technical Report PT/01/BD1009/R, Powergen, plc, Nottingham, UK, September 2001. Prepared under contract to the U.K. Departement of Trade and Industry, DTI Contract 140 (Accct Paper No: 13).

[MCK95] S. Munukutla, M. Craddock, and F. Khodabakhsh. Incorporating CEMs data into heat rate evaluation methods, May 1995. Presented at the Utility Continuous Emissions Monitoring Systems Users Group Meeting, Atlanta, GA.

[MCO91] S. Munukutla, P. Chodavarapu, and D. C. O'Connor. On-line coal analysis from measurement of flue gas components. In *Proceedings of the 1991 ASME Joint Power Generation Conference*, New York, NY, October 1991. American Society of Mechanical Engineers. Paper No. 91-JPGC-PWR-17.

[MK95] S. Munukutla and F. Khodabakhsh. Enhancement of boiler performance evaluation methods using CEMs data. In *Proceedings of the 1995 ASME Joint Power Generation Conference*, volume 29, pages 11–16, New York, NY, October 1995. American Society of Mechanical Engineers.

[MPM88] S. Munukutla, J. Peddieson, and S. Mahajan. Feasibility study for on-line coal analysis and for measuring coal flow in pipes in coal-fired power plants. Technical report, Tennessee Technological University, Cookeville, TN, September 1988. Report prepared for Dayton Power, Duke Power, Pennsylvania Power and Light, Southern Company Services and Virginia Power.

[MTA+88] S. Munukutla, G. Tsatsaronis, D. Anderson, S. Wilson, and J. Harris. FLAPP: A microcomputer software for analyzing the effects of key parameters on plant scherer performance. In *Proceedings of the 1988 EPRI Heat-Rate Improvement Conference*, Palo Alto, CA, May 1988. EPRI.

[NWL00a] Pcams/nt. NWL Inc., Bordentown, NJ, 2000. Product bulletin.

[NWL00b] *PCAMS for Windows NT — Software Manual*, 2000. Software manual.

[OTC03] NEOS guide optimization tree. web page, 2003. Web page developed by US Department of Energy's Argonne National Laboratory Optimization Technology Center.

[Pra98] Praxis optimation software tools. Praxis Engineers Inc., Milpitas, CA, 1998. Sales literature.

[PSTS98] E. Payson, D. Sendro, S. Tavoulareareas, and J. Stallings. Evaluation of power plant optimization options for NOx reduction at allegheny power's armsstrong 1 unit, August 1998. Presented at the EPRI 1998 NOx Controls for Utility Boilers Workshop, Baltimore, Maryland, August 25-27, 1998.

[SCS93] 500 MW demonstration of advanced wall-fired combustion techniques for the reduction of nitrogen oxide (NOx) emissions from coal-fired boilers - field chemical emissions monitoring: Overfire air and overfire air/low NOx burner operation final report. Technical report, Southern Company Services, Birmingham, AL, 1993. Prepared by Radian Corporation for Southern Company Services under contract to the US Department of Energy.

[SCS97a] 500 MW demonstration of advanced wall-fired combustion techniques for the reduction of nitrogen oxide (NOx) emissions from coal-fired boilers — online carbon-in-ash monitors / survey and demonstration. Technical report, Southern Company Services, Birmingham, AL, 1997. Prepared by Southern Company Services under contract to the US Department of Energy.

[SCS97b] Generic NOx control intelligent system (GNOCIS) final report. Technical report, Southern Company Services, Inc. and Powergen, plc., 1997. Report prepared for UK Department of Trade and Industry and US Department of Energy.

[SCS98a] 500 MW demonstration of advanced wall-fired combustion techniques for the reduction of nitrogen oxide (NOx) emissions from coal-fired boilers — final report (phase 1 - 3b). Technical report, Southern Company Services, Birmingham, AL, 1998. Prepared by Southern Company Services under contract to the US Department of Energy.

[SCS98b] 500 MW demonstration of advanced wall-fired combustion techniques for the reduction of nitrogen oxide (NOx) emissions from coal-fired boilers — phase 4 - digital control system and optimization. Technical report, Southern Company Services, Birmingham, AL, 1998. Prepared by Southern Company Services under contract to the US Department of Energy.

[Sin91] Joseph G. Singer, editor. *Combustion Fossil Power*. Combustion Engineering, Inc., Windsor, Connecticut, 4th edition, 1991.

[SK92] S. C. Stultz and J. B. Kitto, editors. *STEAM its generation and use*. The Babcock and Wilcox Company, Barberton, Ohio, USA, 40th edition, 1992.

[SSN+98] J. Sorge, M. Slatsky, J. Noblett, G. Warriner, and J. Stallings. GNOCIS - update on the generic NOx control intelligent system. In *Proceedings of the 1998 EPRI Heat Rate Improvement*

Conference, Palo Alto, CA, September 1998. EPRI. Baltimore, Maryland, September 21-23, 1998.

[TU97] L. H. Tsoukalas and R. E. Uhrig. *Fuzzy and Neural Approaches in Engineering*. John Wiley and Sons, New York, NY, January 1997.

APPENDIX A

GBCORRECT API

GBCorrect API
version 2002.01.09

GENERAL

This paper contains a functional description of the GBCorrect routines used to correct bias between actual and GNOCIS predicted output values.

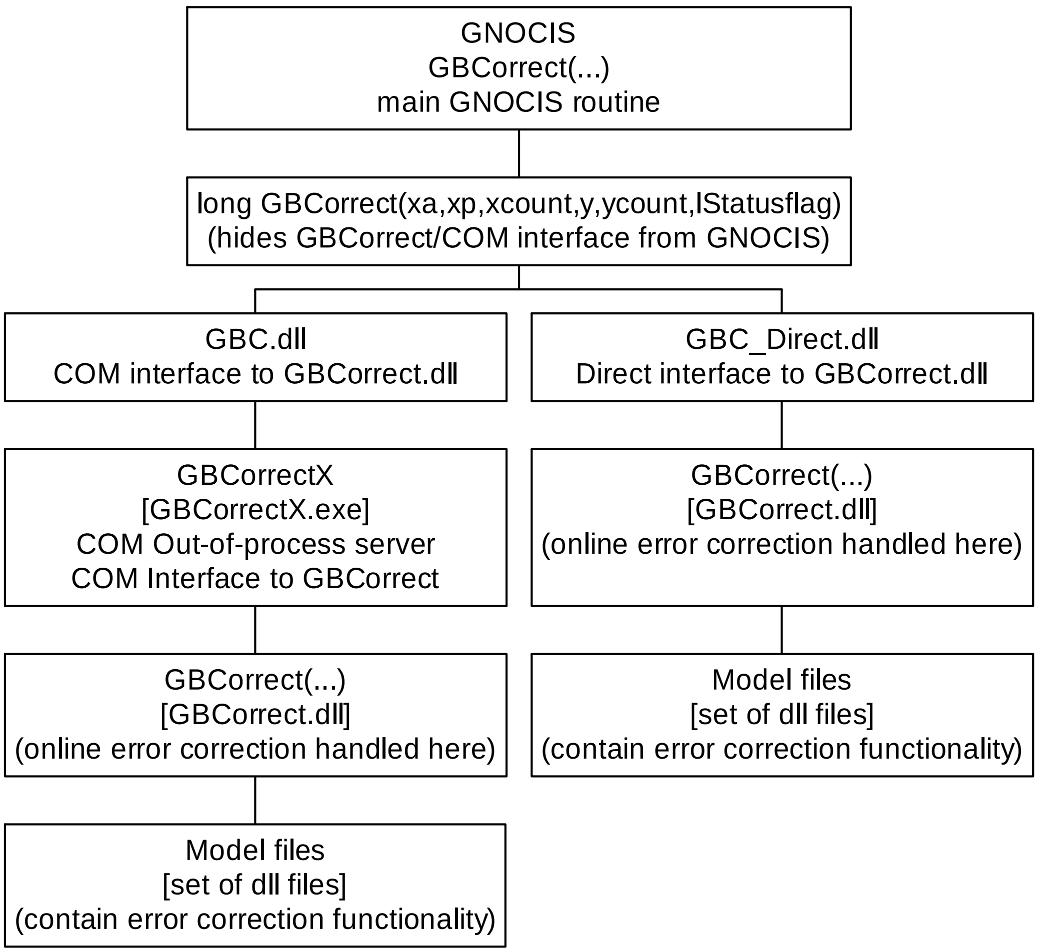


Figure 1. Functional GBCorrect Schematic

FUNCTION DEFINITIONS

long **GBCorrect**(*double* *xa, *double* *xp, *long* xcount, *double* *y, *long* ycount, *long* lStatusFlag)

Description

Executes the bias correction models for GNOCIS.

Parameters

xa: An array of actual values, stored in order of Taglist array (see Initialize()). Consists of two parts: (1) actual input values, stored in the first part of the array and (2) the actual output values, stored in the last part of the array. The actual input values in xp and xa will likely be identical. As used in GNOCIS, for convenience, the order is the same as that used is Process Insights (PI) *Run Model* procedure:

Model Inputs
Model States
Model Outputs

An excerpt from a PI model description file shows this order (see Figure 2).

xp: An array of predicted values, stored in order of Taglist array (see GBInitialize()). It consists of two parts: (1) actual input values, stored in the first part of the array and (2) the predicted output values, stored in the last part of the array. The actual input values in xp and xa will likely be identical and the inputs will generally be ignored in the xp array. This choice of structure was largely driven by the convenience of using the existing array structures in GNOCIS and the PI calls.

xcount: The number of elements in the xa array. Also, the number of elements in the xp array.

y: An array of biases returned to the calling procedure. The size is equal to the size of the xa and xp arrays. The bias array order corresponds to the order of the xa and xp arrays.

ycount: The size of the y array. This number should be equal to xcount, as the order and size correspond directly to the xa and xp arrays.

lStatusFlag: GBCorrect function command. For calls from GNOCIS, this value is equal to the constant GNOCIS_RUN. The constant is defined in the **gbc.h** header file.

Returns

0 if successful; non-zero if unsuccessful.

```
Dataset: /home/gnocis/hammond/ham2000/Ham4_apr_jul_2000
Model: /home/gnocis/hammond/ham2000/ham4_2k_ctrl4
Time Interval:
Filter used: None.

Model Variables:
----------------

     index# (C Language)         control/independent_name   Time Delay
     -------------------         ------------------------   ----------
                       0                        !WMILLAC!            0
                       1                        !WMILLBC!            0
                       2                        !WMILLCC!            0
                       3                        !WMILLDC!            0
                       4                        !WMILLEC!            0
                       5                        !WMILLFC!            0
                       6                        !YAOFAF1!            0
                       7                        !YAOFAR1!            0
                       8                        !YAOFAF2!            0
                       9                        !YAOFAR2!            0
                      10                         !AVG_O2!            0
                      11            !SH_SPRAY_FLOW_UPPER!            0
                      12            !SH_SPRAY_FLOW_LOWER!            0
                      13                    !SUM_BRNR_POS!            0

     index# (C Language)     initial_state/dependent_name   Time Delay
     -------------------     ----------------------------   ----------
                      14                      !AVG_TSAGO!            0
                      15                      !AVG_TPAGO!            0
                      16                      !AVG_TSAAI!            0
                      17                      !AVG_TPAAI!            0
                      18            !AVG_DIV_WALL_INLET_T!           0
                      19                !HOT_REHEAT_PRESS!           0
                      20            !SUPERHEAT_INLET_TEMP!           0

     index# (C Language)   predicted_state/dependent_name   Time Delay
     -------------------   ------------------------------   ----------
                      21                      !AVG_TSAGO!            0
                      22                      !AVG_TPAGO!            0
                      23                      !AVG_TSAAI!            0
                      24                      !AVG_TPAAI!            0
                      25            !AVG_DIV_WALL_INLET_T!           0
                      26                !HOT_REHEAT_PRESS!           0
                      27            !SUPERHEAT_INLET_TEMP!           0

     index# (C Language)                      output_name   Time Delay
     -------------------                      -----------   ----------
                      28                     !NOX_LBMMBTU!           0
                      29                            !CIA!            0
                      30                           !THRH!            0
                      31                            !TMS!            0
                      32                            !PMS!            0
                      33                            !EFF!            0
Raw Tags:
---------
Note: Tag ids are language independent.
```

Figure 2. PI configuration file

long **GBInitialize**(*char* TagListStrings[][32], *long* TagListCount, *LPSTR* ModelName)

Description

Initializes the models with the tag names and the error correction initialization files. This function must be called once before calling GBCorrect().

Parameters

TagListStrings: an array of TagListNames. Each tagname (string) can be up to 32 characters long, including the NULL string terminator. This array order corresponds to the order in which the xa, xp, and y arrays are passed to GBCorrect (see GBCorrect()). The number of elements should also correspond to the number of values in the xa, xp, and y arrays. This array of strings already exists in GNOCIS (see Figure 2).

TagListCount: number of tagnames in the pTagListStrings array.

ModelName: name of the GBCorrect model file (<*ModelName*>.GBCORRECT_INI), without the extension. Will likely correspond to the GNOCIS model name. If given with an extension, such as GBCorrect.ini, then the default ".GBCORRECT_INI" extension will not be appended. If the model name contains a path, the UOPHOME entry in the UOP.ini file will be ignored.

Returns

0 if successful; non-zero if unsuccessful.

void **GBUninitialize**(*void*)

Description

Performs any necessary clean up for objects allocated in memory. Should be called in pairs with GBInitialize(). This function is required for proper termination of the COM object environment for GBCorrectX. Failure to call this routine when using the COM interface could result in memory leaks and memory artifacts of previous processes. Therefore, if the calling process is terminated, the GBCorrectX COM server must be manually terminated to prevent data corruption. This function call is not necessary when using the **GBC_Direct.dll** library.

Parameters

None.

Returns

Nothing.

REQUIRED FILES

GBC.h – contains function prototypes and constants definitions for execution of GBCorrect function using the GBC.dll (COM).

GBC.lib – contains the function mapping for implicit linking to GBC.dll (COM).

GBC.dll – contains the executable for the GNOCIS - COM interface (via dll).

GBC_Direct.h – contains function prototypes and constants definitions for execution of GBCorrect function using the GBC_Direct.dll (non-COM).

GBC_Direct.lib – contains the function mapping for implicit linking to GBC_Direct.dll (non-COM).

GBC_Direct.dll – contains the function calls directly to the GBCorrect module (used for the non-COM version of GBCorrect).

GBCorrectX.exe – contains the out-of-process COM server for GBCorrect.dll. Not directly called by GNOCIS.

A note about GBCorrectX (COM): Since this COM component is an out-of-process server, only one instance is started per machine, not per process. If two applications (i.e., two instances of GNOCIS: one for steam, one for boiler, "what-if") access this COM object running on the same machine, due to the architecture of the application, there is a high probability of data corruption. Whereas this server provides crash protection for other applications, it does not provide exclusive access for each process. Therefore, when two instances of GNOCIS are running on one machine, one or both instances should run with the GBC_Direct.dll (instead of the GBC.dll) to link to GBCorrect in-process. This will provide necessary separation of data for all instances.

GBCorrect.dll – contains the GBCorrect routines. Not directly called by GNOCIS.

GBCorrectLib.dll – contains data structures and methods called by GBCorrect libraries. Not directly called by GNOCIS.

UOP.ini – contains information for unit optimization settings, and is located in the Windows directory. The key, *OLECHOME*, contains the path to GBCorrect initialization

```
[General]
sRTDSHost = NULL
sLoadTag = "4CP001_I:MAIN_4.PNT_3"

[ISBS]
sIniFile = “C:\ISBS\ISBS_Initialization_files”

[OLEC]
OLECHOME = "C:\OLEC\GBCorrect\Initialization_files"
```

Figure 3. UOP.ini file

```
[General]

[CIA_1]
OutputName=CIA_1
Enabled=1
UseModelBias=0
ManualBias=0.577
LowBiasLimit=0.5
HighBiasLimit=1.0
AutoSave=1
ModelPath=C:\OnlineErrorCorrection\lib\nullmodel.dll
ModelData=C:\OnlineErrorCorrection\data\nullmodel.ini|tlm1
```

Figure 4. GBCorrect initialization file

files. *OLECHOME* is located under the *OLEC* section of the **UOP.ini** file.

OPTIONAL FILES

GBC_NULL.dll – for debugging purposes. Contains routines similar to those in GBC.dll, however, without the functionality. Returns zero values in the output array, and will always return successfully. Since function mapping is similar to GBC.dll, no recompilation is required to use this file. Simply replace the functional GBC.dll with this file (renamed to GBC.dll).

GBCORRECT INITIALIZATION FILE

<*ModelName*>.GBCORRECT_INI
Description

Contains settings for the GBCorrect outputs and locations of the model libraries and model settings. This file is automatically loaded when the GBInitialize() is called. If this file cannot be found, the default initialization file, DEFAULT.GBCORRECT_INI is loaded instead. It is suggested that the default file settings disable error correction (*Enabled* flag is set to 0).

Layout

Section: General

Reserved.

Section: <Model_Output_Name>

Sections can be added to this file that correspond to the tagname of the corresponding output. Every output that uses error correction needs a section to define the type of error correction model that will be used to generate its bias. In this example, the output is the

CIA value, and the tagname is CIA_1. Within this section contains the information regarding the error correction settings for that output. These are described as follows:

OutputName: tagname of the output. Should be the same name as the section heading.

Enabled: turns the bias error correction on or off. 0 is off, 1 is on. If *Enabled* is set (1), the bias comes from either the bias models or the *ManualBias*. If cleared, the returned value will be zero.

UseModelBias: turns the calculated model bias on or off. 0 is off, 1 is on. If this value is set to 0, the model bias returned from the calculations is ignored, and the value of *ManualBias* is used instead.

ManualBias: contains the value used to replace the calculated model bias for this output. If *UseModelBias* is 1, this value is ignored.

LowBiasLimit: value of the low constraint on the model bias. Must be less than or equal to *HighBiasLimit.* One way to turn of the bias would be to set both *HighBiasLimit* and *LowBiasLimit* to zero.

HighBiasLimit: value of the high constraint on the model bias. *Must be greater than or equal to LowBiasLimit*. One way to turn of the bias would be to set both *HighBiasLimit* and *LowBiasLimit* to zero.

AutoSave: value of save frequency for the specific model. Given in terms of iterations (a value of one indicates that the model will save its data every iteration). A value of zero will disable the AutoSave feature for this model.

ModelPath: the path to the specific model's dll. All models and sections require this field.

ModelData: the path to the specific model initialization string (contains the model ini file and model name). Most model types will require this field but some simple ones may not.

APPENDIX B

ISBS SOFTWARE DESCRIPTION

Intelligent Sootblowing System (ISBS)

version 2002.07.19

GENERAL

This document describes the Intelligent Sootblowing System (ISBS), and is intended for both end users of the ISBS software as well as those who wish to incorporate the ISBS into custom-coded software.

SOFTWARE OVERVIEW

The system architecture is a client/server application. The client has many incarnations including a standalone engine control-based client, a standalone operator client, and separate ActiveX operator client controls. The server is written as a COM-compliant server. The ISBS server is the calculation engine for the ISBS, whereas the client modules offer limited control over the engine and provide feedback on the engine status. The primary language for the ISBS package is Visual Basic.

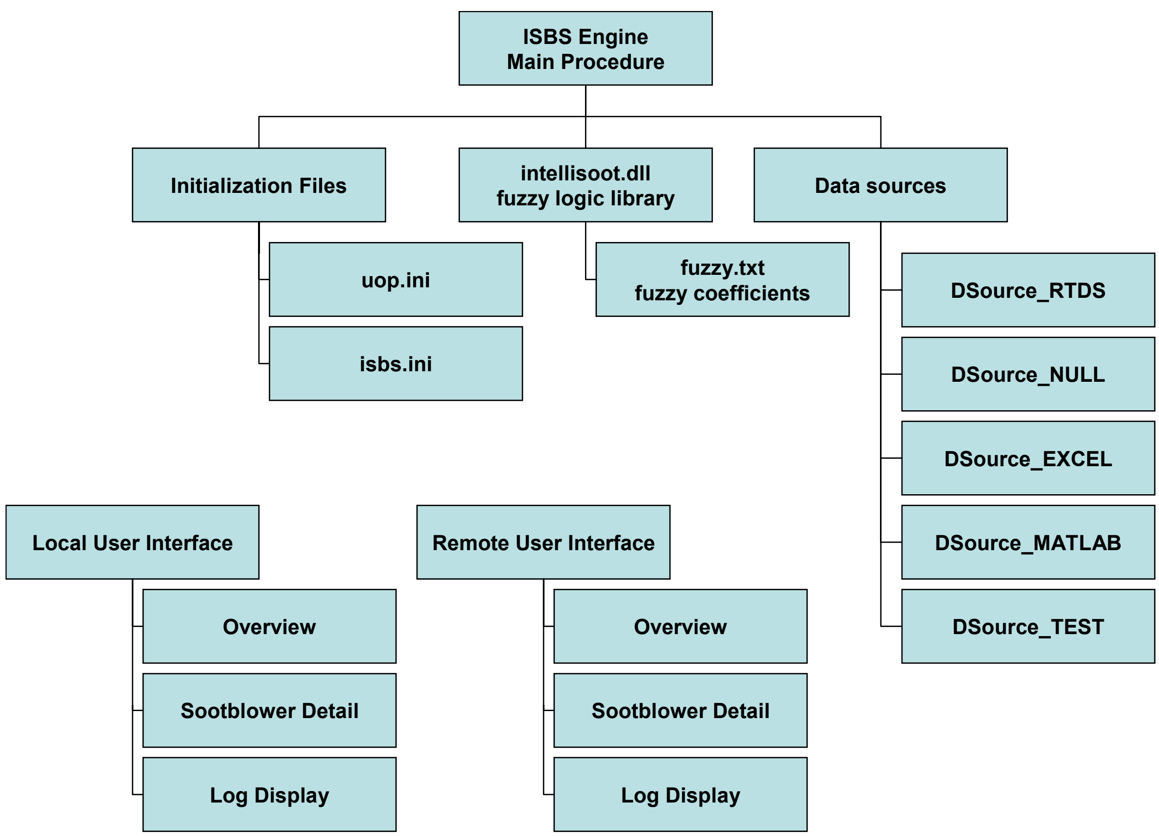


Figure 1. ISBS overview.

CLIENT USER INTERFACES

Description

There are several options for accessing the ISBS engine information during runtime.

- Master client
- Operator client
- Operator client controls

All clients display ISBS engine information, including group recommendations, current sootblower values, engine log messages, and other selected DCS tags such as unit load. The master client has additional control and information available to its user. All clients also possess the capability of starting the ISBS engine; however, with the DCOM configuration set as described above, this should only be possible from the server machine or with the correct administrative userid and permissions. This is by design.

The information presented by the user clients is divided into three panels: main panel, log panel, and the sootblower detail panel. All panels display the current ISBS engine/connection status specific to their individual process.

The main panel shows a graphic display of the status of the sootblower recommendations. A bar chart is used to visually describe the current recommendations to activate a specific sootblower group. If a recommendation exceeds a predetermined threshold, the bar changes color to indicate a need to sootblow. Other information provided by the main panel includes unit load, reheat temperature and damper position, superheat spray flows and temperatures, and sootblower group activity. Also shown are time stamps indicating the last state of a sootblower group.

The log panel displays messages generated by the ISBS engine for the current log level. The log level is determined during initialization of the ISBS engine process (from an initialization file), or by the master client panel.

The sootblower detail panel displays the current state of the individual sootblowers within their respective groups.

Master Client

SCISBSClient.exe
The master client is a standalone application that provides additional control over the lifetime of the ISBS engine process (Figure 2). The master client can display the same information as the operator clients, but can also clear the text log buffer, enable and set the text log buffer file, request a forced ISBS engine shutdown, restart a terminated ISBS engine process, and lock and unlock the ISBS engine into memory (the engine will then run regardless of the number of clients attached).

Operator Client

SCISBSOperatorClient.exe; SCISBSCtlClient.exe
There are two incarnations of the operator client. The first is based upon the master client interface. The second was constructed using the operator client ActiveX controls. Both programs (Figure 5 - Figure 9) are operational. Since the ActiveX controls create independent connections to the ISBS engine, each control will be counted as a client connection. For example, when the control-based client starts, three connections will be made to the ISBS engine, and the master client will display an additional three connections on its main panel.

Operator Client Controls

SCISBSMainCtl.ocx; SCISBSDetailCtl.ocx; SCISBSLogCtl.ocx
There are three operator client controls: the main control, the log control, and the detail control. These controls may be embedded within a web page or within a custom application, such as the operator client. Each control has an independent connection to the ISBS engine, and therefore, will be counted as a connection when activated. If the control is loaded, the connection is active.

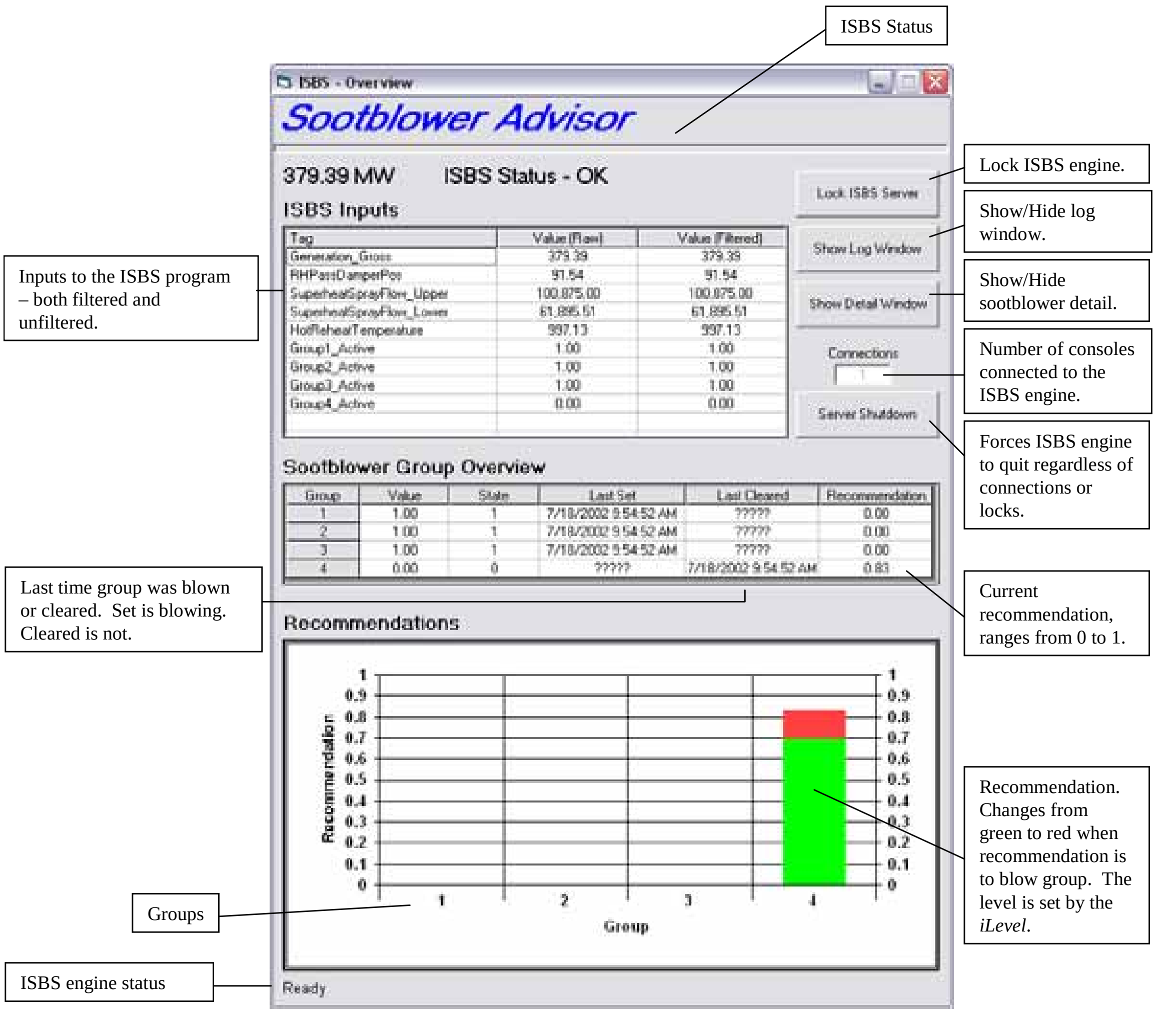


Figure 2. ISBS master client – main display.

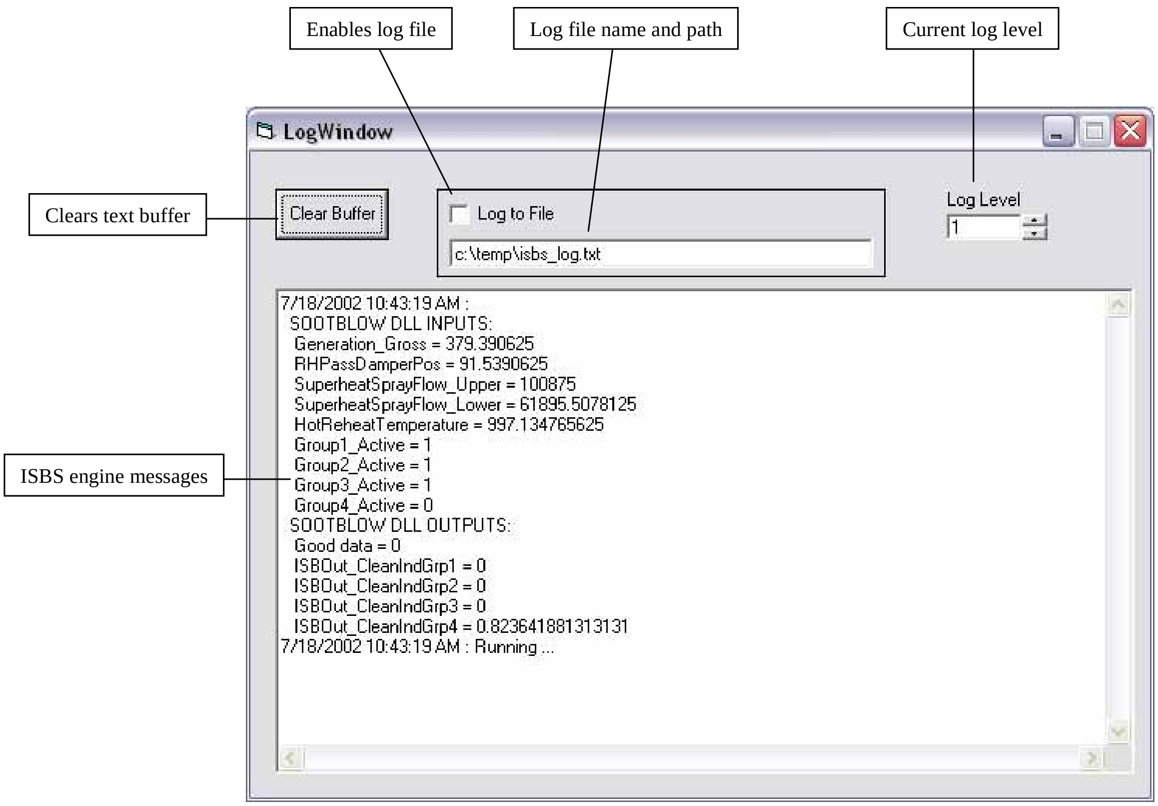


Figure 3. ISBS master client – log display.

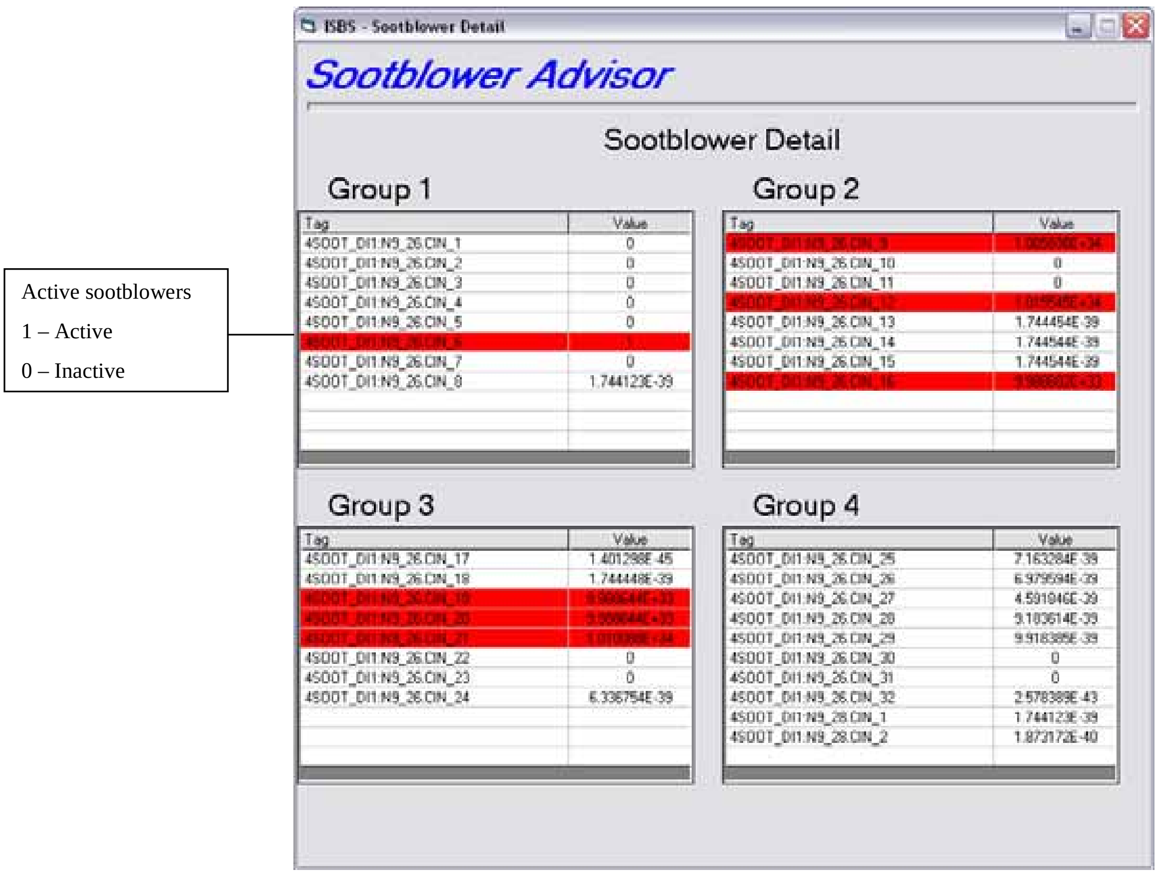


Figure 4. ISBS client – sootblower group detail display.

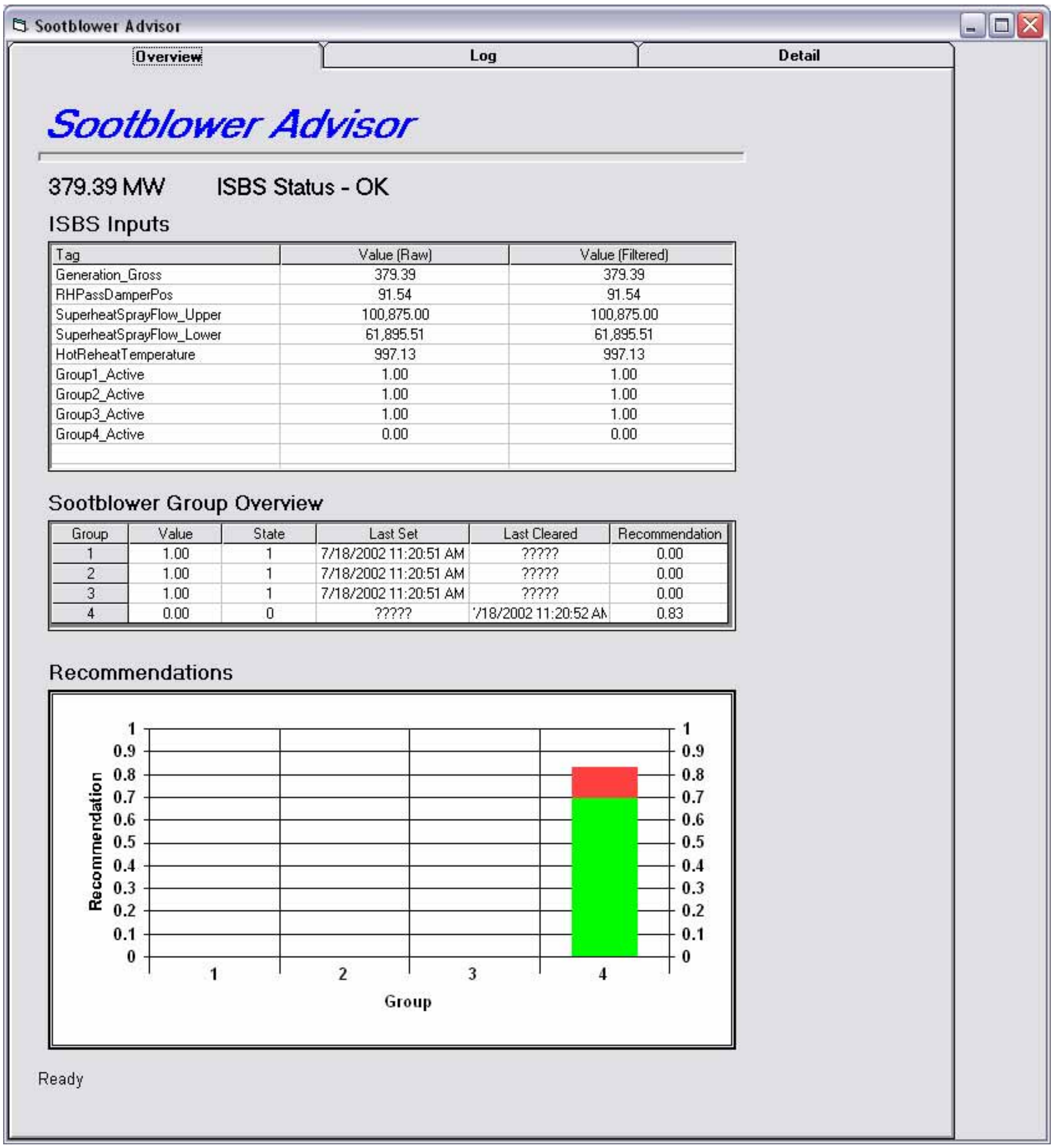


Figure 5. ISBS control-based client – main display.

Figure 6. ISBS control-based client – log display.

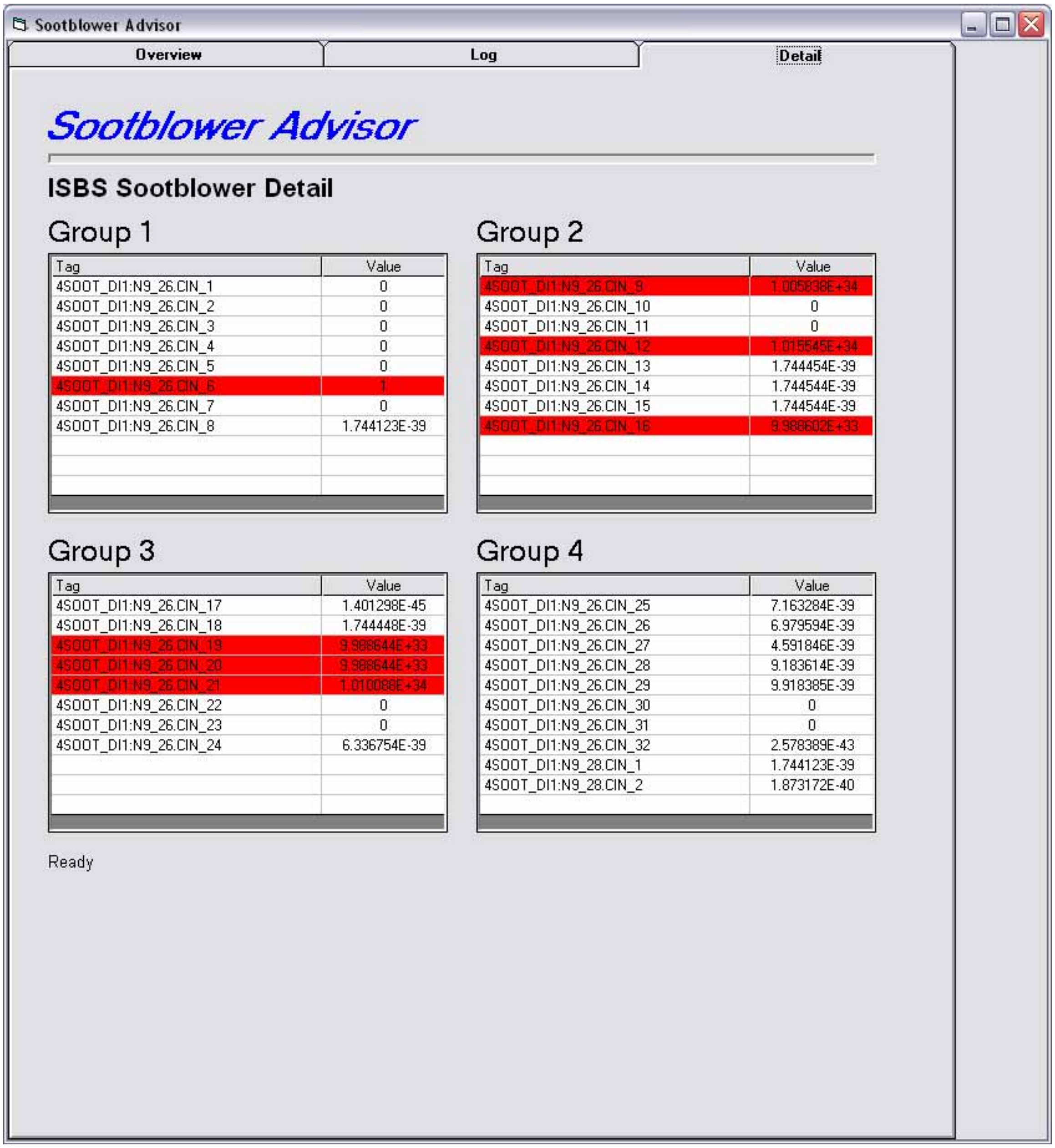


Figure 7. ISBS control-based client – sootblower group detail display.

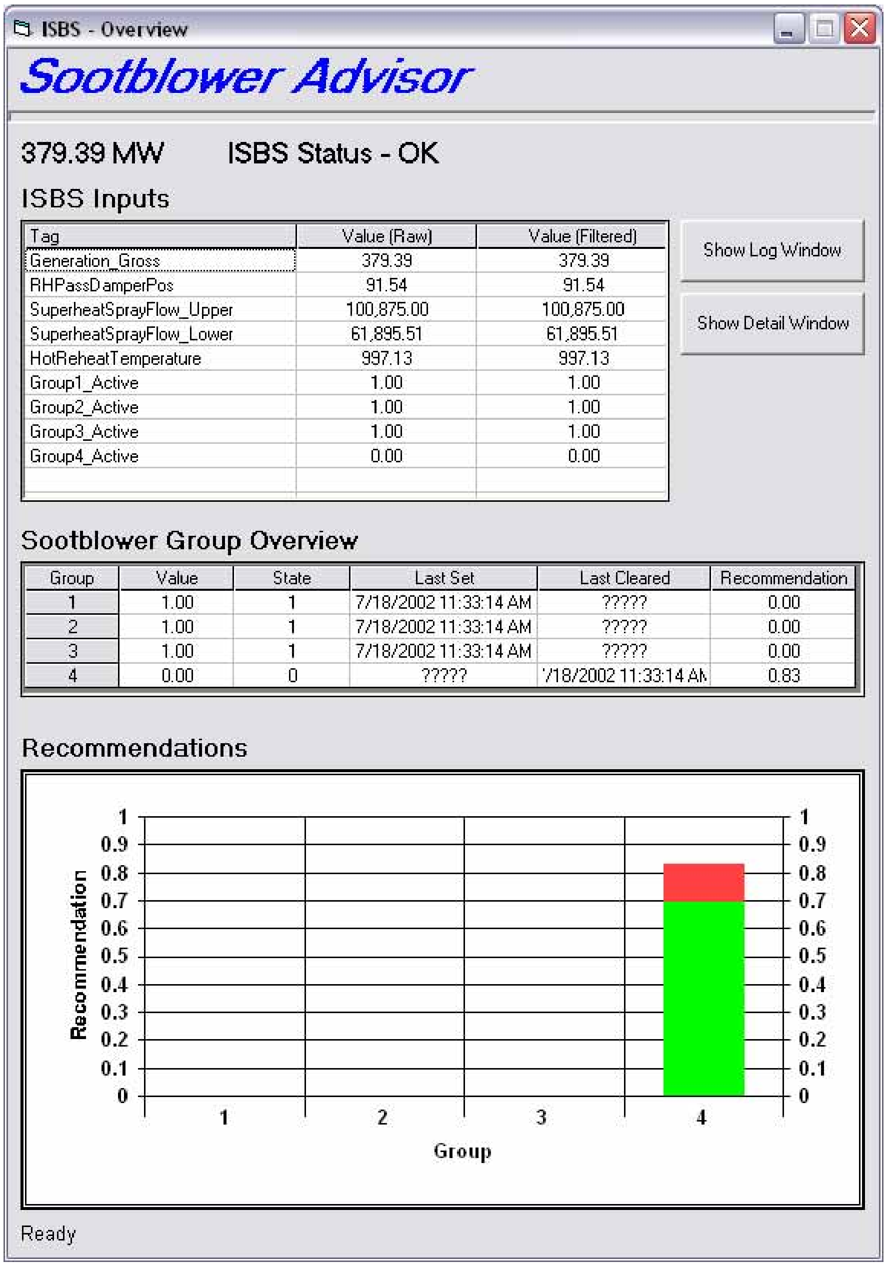


Figure 8. ISBS operator client. – main display.

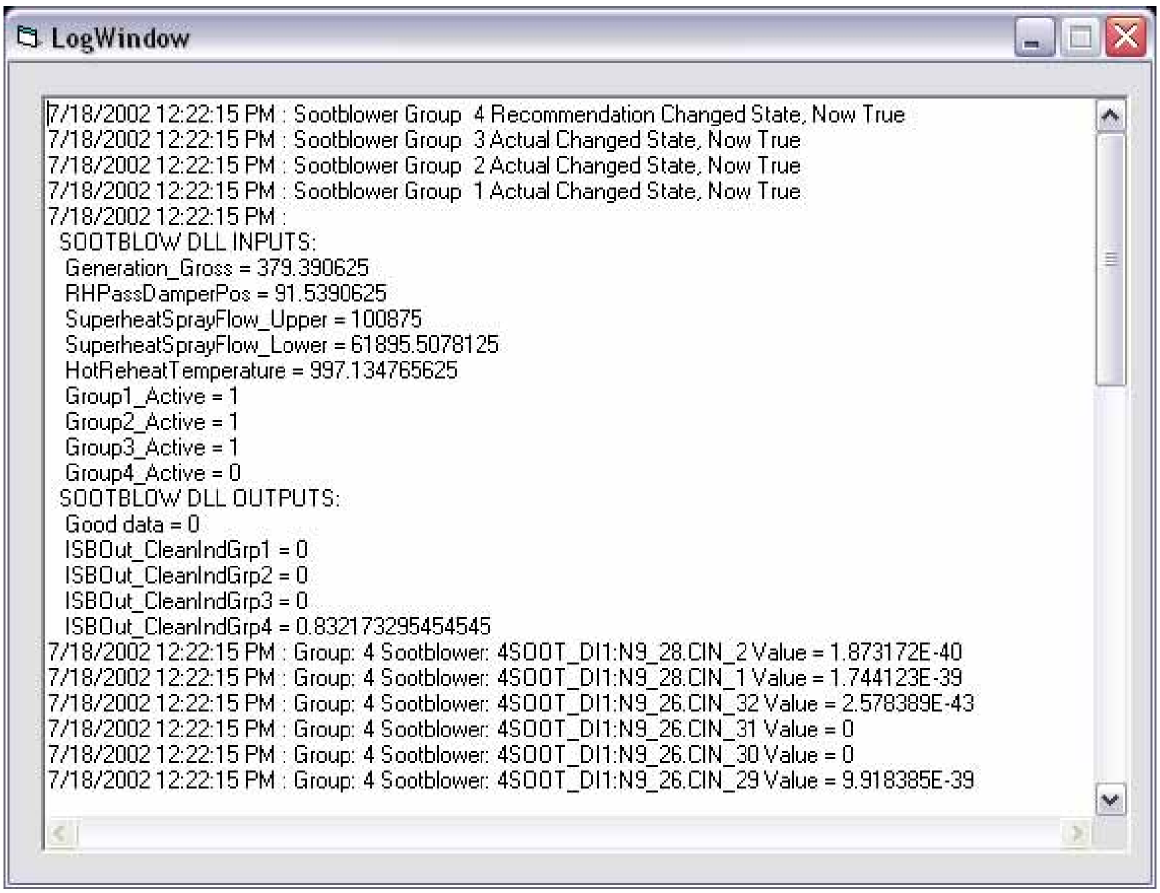


Figure 9. ISBS client – log display.

REQUIRED FILES

Intellisoot.dll – Dynamic link library that implements the fuzzy logic rule base for generating recommendations. The file *fuzzy.txt* contains coefficients for adjusting trapeziums defining the fixed fuzzy rules. Additional information on the rule set can be found in the document *Intelligent Sootblowing System at Plant Hammond*.

Fuzzy.txt – Fuzzy logic settings for Intellisoot.dll.

SCISBSLib.exe – ISBS sootblower engine. There is no integrated user interface in this module. External control consoles are used to monitor and control this engine.

SCISBSClientInterface.dll – Contains the interface implemented in the client/server COM objects used in callback routines.

ISBS.ini – ISBS initialization file.

UOP.ini – Unit optimization initialization file.

Client programs – There are several means of connection and startup of the ISBS engine. All currently implemented methods are done via client programs/controls. A list of these programs is shown below.

- **SCISBSOperatorClient.exe**
- **SCISBSClient.exe**
- **SCISBSCtlClient.exe**
- **SCISBSMainCtl.ocx**
- **SCISBSLogCtl.ocx**
- **SCISBSDetail.ocx**

A description of these files appears later in this documentation.

OTHER FILES

***ISBS_log.txt* –** ISBS log file. The fully qualified path and filename is specified in the ISBS initialization file. This can be changed from the user panels.

***ISBSErrorLog.txt* –** ISBS engine error log file. The fully qualified path and filename is specified in the ISBS initialization file. There is no other means of determining the location or name of this file.

DATA SOURCES

Description

The software is designed so that the data source can be changed. The driver *DSource_RTDS*, an interface to the RTDS, will typically be used. Other drivers have been developed for testing purposes including an interface to MATLAB and a null source interface.

SERVER SOFTWARE

ISBS Engine

SCISBSLib.exe
This software contains the core of the ISBS calculations as well as the client/server communications capabilities. There are several methods of connecting to this engine to obtain its operational and configuration data. It was designed to allow maximal access to data without compromising the stability of the running process. The public objects in **ISBSLib** are DCOM compliant. An internal timer executes the ISBS calculation at a regular user-defined interval (specified in an initialization file). Results are extracted via one of the many client software options, and are also written back to the Real Time Data Server (RTDS).

PowerGen Intellisoot Library

Intellisoot.dll
Developers at PowerGen created a library for calculating sootblowing recommendations. The function call, Intellisoot(), is included in the library, **Intellisoot.dll**. It requires a settings file, **Fuzzy.txt**. This file contains tent weightings for the fuzzy logic control scheme. **Fuzzy.txt** must be located in the same working directory as the process incorporating the function call; otherwise, a runtime error will occur and the process will terminate. The working directory can be specified in the **ISBS.ini** file. The Southern Company program implementing this routine provides checks to prevent this type of error from occurring if the library is unable to locate the settings file.

Details of this routine are given in related PowerGen documentation.

INITIALIZATION FILES

Unit Optimization Initialization File - *UOP.INI*

Description

This is the main initialization file used for the unit optimization project, and contains information for use by the ISBS software. Specifically, it contains the location of the ISBS initialization file, **ISBS.ini**. The UOP file can be located in any user accessible directory on the hard drive, provided the environment variable, *UOPINI*, contains the full path and filename of the UOP file. For example:

UOPINI=C:\Program Files\UOP\UOP.ini

```
[General]
RTDSHost = UOPHostComputer
LoadTag = "4CP001_I:MAIN_4.PNT_3"

[ISBS]
IniFile = C:\DATA\Isbs2_VCOM\isbs.ini
```

Figure 11. ISBS entry in UOP.ini file.

If this variable is set, the ISBS engine will use the UOP.ini file to determine the location of the ISBS.ini file. If the UOPINI variable is not set, the ISBS engine will shutdown immediately, and the client will receive an ambiguous error that the object is not defined. This error is shown below.

Figure 10. Client error when ISBS engine cannot find UOP.ini.

NOTE: An error is also written to the ISBS error log file. However, since this error occurs before the ISBS engine sets all file paths and variables, by default, the error is written to the Windows system directory as the file _ISBS_ERROR_LOG.TXT.

The ISBS entry in the UOP file is shown in Figure 11.

Layout

Section: General

RTDSHost: Default RTDS host platform – where the RTDS resides. This should be a string. The domain name is not needed.

LoadTag: Default tag name for the load parameter in the RTDS.

Section: ISBS

Section reserved for the ISBS.

IniFile: Full path and filename of the ISBS initialization file (ISBS.ini). May contain references to environment variables. References to environment variables are enclosed by ‘%’. For example, if *EnvVar* is a system variable, and is set to **c:\uophome**, then the string

IniFile = %ENV_VAR%\isbs.ini

evaluates to

IniFile = c:\uophome\isbs.ini

in the UOP.ini file.

ISBS Initialization File - *ISBS.INI*

Description

The ISBS.ini file contains detailed settings for the ISBS engine. This file is read when the engine initially starts, and the settings are maintained throughout the life of the engine process unless the parameters are changed via the main control user interface. An example of this file is shown in Figure 12.

Layout

Section: General

RTDSHost: Specified RTDS host platform – where the RTDS resides. If not defined, defaults to that in the UOP initialization file. This should be a string. The domain name is not needed. To use this driver, the RTDS COM object must be registered on the computer where the ISBS is running. Special cases are provided for debugging:

Value	Description
NULL	Inputs are zero and outputs are discarded.
EXCEL	The ISBS program interfaces with EXCEL instead of the RTDS (not implemented yet).
MATLAB	The ISBS program interfaces with MATLAB instead of the RTDS. You must have MATLAB to run in this mode.
TEST	Constant test values.

LoadTag: Specific tag to be used as the load index. If not defined, defaults to that in the UOP optimization file.

CycleTime: Interval (in seconds) program updates, including reading input data, processing, and outputting data. Suggested range is 20 to 120.

FilterConstant1: Constant used to filter inputs. Filtered inputs are defined as:

$$X_f = X_f * FilterConstant1 + (1-FilterConstant1) * X$$

```
[General]
RTDSHost = HostComputerName
LoadTag = "4CP001_I:MAIN_4.PNT_3"

CycleTime = 20
FilterConstant1 = 0.8
FilterConstant2 = 0.8
Level1 = 0.7

LogLevel = 200
LogFile = c:\temp\isbs_log.txt
LogToFile = 0

;Location of the fuzzy.txt file
WorkingDirectory=C:\DATA\Isbs2_VCOM\Code

ErrorLogFile=C:\Data\Isbs2_VCOM\Code\ISBSErrorLog.txt
ErrorLogSize = 255
```

Figure 12. ISBS initialization file.

where X are the unfiltered values read from the data source and X_f are the filtered inputs. If *FilterConstant1* = 0, there is no filtering. Permissible range: [0, 1].

FilterConstant2: Constant used to filter output recommendations. Filtered outputs are defined as:

$$Y_f = Y_f * FilterConstant2 + (1-FilterConstant2) * Y$$

where Y are the unfiltered recommendations and Y_f are the filtered recommendations. The filtered recommendations are stored to the RTDS and displayed. If *FilterConstant2* = 0.0, there is no filtering. Permissible range: [0, 1].

Level1: Sets threshold for recommendation to blow. Permissible range: [0, 1]. Typical value is 0.7. If 0.0, the recommendation is always to activate the sootblower group. As *dLevel1* is increased to 1.0, the recommendation to activate the sootblower group becomes less likely.

LogLevel: Sets threshold for diagnostic messages to the textual log buffer and file. If the message level is less than or equal to *Level1*, then the message is displayed. Otherwise, the message is ignored. In most instances, 0 (the default) is the correct setting, allowing only critical status messages to be produced. If set to –1, all diagnostic messages are ignored. Permissible range: [-1, 255].

LogFile: Sets the default log file for the program. The default is *c:\temp\isbs_log.txt*. The log file can be changed through the log window on the master client console. If the file exists, the messages are appended to the end of the file. If the file does not exist, a file is created.

LogToFile: If set to 0, does not write log messages to file. If set to 1, log messages are written to the log file.

WorkingDirectory: Location of the Fuzzy.txt file. Failure to properly set this variable could result in the failure of the ISBS engine code. The ISBS engine attempts to detect the existence of the fuzzy.txt file on every calculation iteration, and skips the Intellisoot procedure call if this file is not found so that a fatal error is avoided.

ErrorLogFile: Name and path of the error log file. Errors generated by the ISBS engine are written to this file.

ErrorLogSize: Maximum number of textual lines the error log file will accumulate. Additional lines are appended, and the oldest entries are removed when this limit has been reached.

INSTALLATION

Server-side Installation

Since there are a number of files associated with the server software, creating a directory exclusively for the ISBS is recommended. In addition, the **fuzzy.txt** file should be placed in the same directory as the **intellisoot.dll** file. A possible directory structure is shown in Figure 13.

Once these files have been copied to the desired directories, and the initialization files have been edited, the server files must be registered with the system.

The classes contained in this engine must be registered on both the client and server computer systems. Typically, client systems are "packaged" with appropriate keys and values that are inserted into the local system's registry automatically upon installation of the client software. The server software can be registered by either: 1) running the program one time or 2) running the following command from a console prompt.

```
c:\>  scisbslib /regserver
```

The standard interface, *ISCISBSClient*, must be registered. This file is registered on the server machine from a console prompt with the following command:

```
c:\>  regsvr32 SCISBSClientInterface.dll
```

The server must be configured for distributed use. This is done via the Window utility, **dcomcnfg**. The following are the recommended permissions and settings for the server:

```
c:\
├── windows
│     └── uop.ini
├── temp
│     └── isbs_log.txt
└── rtds
      └── isbs
            ├── fuzzy.txt
            ├── isbs.ini
            ├── isbslib.exe
            ├── scisbsclientinterface.dll
            └── intellisoot.dll
```

Figure 13. Suggested directory structure for the ISBS software.

Authentication Level: *Connect*. Client programs are required to have this authentication level, such that a client's id is transmitted to the server upon initial connection to the COM object.

Location: *Run program on this computer*.

Identity: *ISBSUser*. Create a unique userid (such as *ISBSUser*) on the local computer domain and assign this user Administration rights to the workstation. This user should then be set up as the executor of the **SCISBSLib** program.

Launch Permissions: *LocalDomain/Administrators*. The users that appear in this list should include all administrators of the local machine. This, by default, will include the unique userid created in **Identity**, provided that user was added to the *Administrators* list for the local computer. This will restrict the starting of the ISBS engine to local domain administrative users.

Access Permissions: *Everyone*. Among the users that appear in this list should appear *Everyone*. This will allow a broad audience to view the ISBS engine status. However, *Everyone* will not be allowed to launch the engine.

Client-side Installation

Because of the distributed nature of this software package, it is recommended that clients should be installed via "packaged" programs that contain installation scripts. These scripts correctly register the remote ISBS server classes. It is necessary to provide the setup program with the server's IP address. This may also be the machine name, if the name can be resolved by domain name services.

INTELLIGENT SOOTBLOWING SYSTEM (ISBS) API

General

This section describe the programmatic interfaces used to implement the ISBS software. These function calls also provide the necessary documentation for implementing new clients of the ISBS server.

Interface: **ISCISBSClient**

Description

This interface was written in Visual Basic and is intended for standalone client-server applications. Implementation of *ISCISBSClient* guarantees to the server that methods will be present when a client COM object is registered with the ISBS server for callback functions. These are functions implemented in the client and called by the ISBS server.

Methods

Sub **ISBSUpdate**()
Called when data is updated.

Sub **ISBSError**(ByVal ErrorCode as *Long*)
Called when an error occurs in the ISBS engine. *ErrorCode* is the numeric value of the error.

Sub **ISBSOK**()
Typically called after an error state, if and when the ISBS engine has resolved previous errors.

Sub **ServerShutdown**()
Called when a command is issued to shutdown the ISBS engine. This allows the client to cleanly terminate its connections to the server process.

Sub **ISBSLogUpdate**()
Called when only the textual log data has been updated.

Sub **ProcessActivity**()
Called whenever there is multi-process activity, such as when a new client connects to the server or an existing client disconnects from the server.

Sub **ServerLocked**()
Called when ISBS engine has been “locked” into memory.

Sub **ServerUnlocked**()
Called when ISBS engine has been “unlocked” from memory.

Properties

ClientID as *Long*
A number designated by the ISBS server engine instance, used to uniquely identify and account registered clients. This property should only be set implicitly by the ISBS server on a client registration call.

Class: **SCISBSClientServer**

Implements: *ISCISBSClient*

Description

This class is a COM realization of the *ISCISBSClient* interface. Standalone clients should contain one instance of this class to allow communication between the client and the ISBS engine server via callback functions. When a method of a registered *SCISBSClientServer* object is called by the ISBS server, the object then raises an event corresponding to that method. This allows the client application to respond to these calls initiated by the ISBS server, by responding to the events that are raised. (See *ISCISBSClient* for additional information).

Methods

The methods are not called explicitly by the client object. Instead, the client code reacts to events raised by the object as a result of callback functions from the ISBS engine. The events are trapped in the client code.

Sub **ISBSUpdate**()
Raises ISBSUpdate event.

Sub **ISBSError**(ByVal ErrorCode as *Long*)
Raises ISBSError event. *ErrorCode* is the numeric value of the error, as defined by the server.

Sub **ISBSOK**()
Raises ISBSOK event.

Sub **ServerShutdown**()
Raises ServerShutdown event.

Sub **ISBSLogUpdate**()
Raises ISBSLogUpdate event.

Sub **ProcessActivity**()
Raises ProcessActivity event.

Sub **ServerLocked**()
Raises ServerLocked event.

Sub **ServerUnlocked**()
Raises ServerUnlocked event.

Properties

The properties are not called explicitly by the client object.

ClientID as *Long*

A number designated by the ISBS server engine instance, used to uniquely identify and account registered clients. This property should only be set implicitly by the ISBS server on a client registration call.

Class: **SCISBSConnector**

Description

This class allows client applications to obtain connectivity to a shared ISBS engine object via COM. The ISBS engine keeps a current count of the total number of these objects that have been instantiated. All clients (standalone and controls) use this object to communicate and obtain the interface to the ISBS engine. This object is encapsulated in the *SCISBSClientServer* class.

Methods

Sub **CloseAllConnections**()
Commands the global ISBS engine object to close all communications with clients using callback routines and sockets. This allows the client to cleanly terminate its connections to the server process.

Sub **LockServer**()
Prevents the ISBS engine from terminating even if there are no clients attached to the object.

Sub **UnlockServer**()
Permits the ISBS engine to terminate if the last connected client terminates.

Sub **GetISBS**(oISBS as *Object*)
Obtains a reference to the global ISBS engine object and returns it via the parameter list.

Properties

IsServerLocked as *Long*
Returns current locking status of the ISBS engine.
0 = unlocked; 1 = locked

ConnectionCount as *Long*
Returns current number of clients (ISBSConnector objects) that are connected to the ISBS engine.

ISBS as *SCISBS*
Obtains a reference to the global ISBS engine object (SCSISBS). This property is read-only.

Class: **SCISBS**

Description

This class encapsulates the calculation engine of the ISBS. PowerGen calculations are called at a specified time interval and results are stored and made available in publicly accessible memory locations. Only one instance of this object is started per SCISBSLib server. This object is accessed via the ISBSConnector object. This engine is responsible for calculating new sootblowing recommendations as well as informing and managing all connected clients of the ISBS engine.

Methods

Sub **SendErrorEvent**(ByVal ErrorCode as *Long*)
Indicates an engine error state to all connected clients.

Sub **SendISBSOKEvent**()
Indicates to all connected clients that previous engine errors have been resolved.

Sub **SendServerShutdownEvent**()
Issues a server shutdown request to all clients (clients terminate their own connections). When the last client has released its ISBSConnector object, the ISBS engine will terminate, regardless of the engine lock status.

Sub **ShutdownServer**()
Terminates all client connections and quits the ISBS engine, regardless of engine lock status.

Sub **SendProcessActivityEvent**()
Indicates to all clients that there has been some miscellaneous activity (typically, a new client has joined or an existing client has terminated). This is currently used by the client to update the number connections displayed.

Sub **SendServerLockedEvent**()
Indicates to all clients that the server has been locked, and therefore, will continue executing without attached clients.

Sub **SendServerUnlockedEvent**()
Indicates to all clients that the server has been unlocked, and therefore, the engine will terminate when the last client terminates its connection to the engine.

Sub **RegisterClient**(isb as *ISCISBSClient*)
Registers a COM client with the server as a callback client. Messages sent to this client will be done via callback methods of the implemented *ISCISBSClient* object.

Sub **ReleaseClient**(isb as *ISCISBSClient*)
Releases a COM client from the server that was previously registered using the RegisterClient method.

Function **RegisterSocketClient**(hostname as *String*, portnumber as *Integer*, ErrorMessage as *String*) as *Long*
Registers a client that will transmit message events via sockets. COM will still be used for data transfer between client and server, however, notification of events will be handled via sockets. This is implemented in the ActiveX controls (and therefore, the web pages). These controls are also used in the operator standalone control client application, and therefore, the operator client will use sockets as its notification medium. *hostname* is the name of the local client computer, *portnumber* is the local port that has been opened and is listening for server messages, and *ErrorMessage* is used to return any errors that the ISBS engine may encounter while attempting to connect to the client socket during this function call. This function returns the client id assigned by the ISBS server, or, if failure occurs, the associated error number.

Sub **ReleaseSocketClient**(lClientID as *Long*)
Releases a client previously registered with RegisterSocketClient. *lClientID* is the client id previously assigned by the RegisterSocketClient.

Properties

Log as *SCISBSLogData*

Allows retrieval of messages written to the ISBS engine log. This is a read-only property, however, properties of the object (*SCISBSLogData*) are read-write.

Data as *SCISBSData*

Allows retrieval of sootblowing data from the ISBS engine. This is a read-only property, however, properties of the object (*SCISBSData*) are read-write.

Detail as *SCISBSGroupDetail*

Allows retrieval of individual sootblower group data from the ISBS engine. This is a read-only property, however, properties of the object (*SCSISBSGroupDetail*) are read-write.

Out0 as *Long*

Outcome of the last attempted function call to the Intellisoot calculation routine.

dLevel as *Double*

Level at which the ISBS recommends a particular sootblower group should be activated.

Class: **SCISBSLogData**

Description

This class handles all messages generated by the ISBS engine intended for textual feedback for the engine calculation status. Depending upon the log level set within this class, a buffer stores these messages up to a predetermined amount and, thereafter, older messages are discarded by new text additions.

Methods

Sub **Clear**()
Erases the contents of the text buffer.

Function **Init**() as *Integer*
Initializes the object. Returns 0.

Function **Add**(sString as String) as *Integer*
Appends sString to the text buffer. Returns 0.

Function **AddwL**(sString as String, iLevel as Integer) as *Integer*
Appends sString to the text buffer as an iLevel message. Returns 0.

Properties

LogLevel as *Long*
Current level of messages logged to the text buffer.

LogFileName as *String*
Current name of the file to which the text buffer is written.

LogToFile as *Long*

Starts or stops messages being written to the log file.

LogMessages as *String*
The contents of the current text log buffer.

Class: **SCISBSData**

Description

This class encapsulates the information regarding the sootblowers and their groupings.

Methods

There are no methods for this class.

Properties

DSCount as *Integer*
Returns the number of data items currently in memory. This property is read-only.

DSTagName(i as *Integer*) as *String*
The associated tag name for the index, i.

DSValue(i as *Integer*) as *Single*
The current value of the data.

DSSum(i as *Integer*) as *Double*
The current summation of the specified DSData array.

DSAverage(i as *Integer*) as *Double*
The current average of the specified DSData array.

SBTagCount(i as *Integer*) as *Integer*
The current number of tags associated with the sootblower group, i.

SBTagName(GroupIndex as *Integer*, TagIndex as *Integer*) as *String*
Returns the tag name for the given sootblower group and index within that group.

SBTagValue(GroupIndex as *Integer*, TagIndex as *Integer*) as *Variant*
Returns the value of the given sootblower group for the given tag index within that group.

Load as *Double*
Current unit load in MW.

Count as *Integer*
The total number of sootblower groups.

GroupState(i as *Integer*) as *Integer*
The current state of the sootblower group i.

GroupValue(i as *Integer*) as *Double*
The current recommendation value for the sootblower group i.

GroupTimeLastSet(i as *Integer*) as *Variant*
The last time the sootblower group i state was set.

GroupTimeLastCleared(i as *Integer*) as *Variant*
The last time the sootblower group i state was cleared.

OutValue(i as *Integer*) as *Double*
The output value of the sootblower group i.

OutFiltered(i as *Integer*) as *Double*
The filtered output value of the sootblower group i.

OutFilteredAbs(i as *Integer*) as *Double*
The absolute filtered output value of the sootblower group i.

OutMax(i as *Integer*) as *Double*
The maximum output value of the sootblower group i.

OutMin(i as *Integer*) as *Double*
The minimum output value of the sootblower group i.

OutEnv(i as *Integer*) as *Double*
The env output value of the sootblower group i.

Class: **SCISBSGroup**

Description

This class encapsulates individual sootblowers as a single group.

Methods

There are no methods for this class.

Properties

TagName(i as *Integer*) as *String*
The tagname of sootblower i.

TagValue(i as *Integer*) as *Variant*
The associated tag value of sootblower i.

TagNames as *Variant*
An array of tagnames associated with the sootblower group.

TagValues as *Variant*
An array of tag values associated with the sootblower group.

Count as *Integer*
Total number of sootblowers in the group.

GroupState as *Integer*
Current state of the sootblower group.

GroupValue as *Double*
Current value of the sootblower group.

TimeLastSet as *Variant*
Last time when the sootblower group state was set.

TimeLastCleared as *Variant*
Last time when the sootblower group state was cleared.

OutValue as *Double*
Output value of the sootblower group.

OutFiltered as *Double*
Filtered output value of the sootblower group.

OutFilteredAbs as *Double*
Filtered absolute output value of the sootblower group.

OutMax as *Double*
Maximum output value of the sootblower group.

OutMin as *Double*
Minimum output value of the sootblower group.

OutEnv as *Double*
Env output value of the sootblower group.

Class: **GroupDetail**

This class encapsulates detailed data about collective sootblower groups.

Methods

There are no methods for this class.

Properties

TagName(i as *Integer*, Group as *Integer*) as *String*
The tag name for sootblower i in Group.

Value(i as *Integer*, Group as *Integer*) as *String*
The tag value for sootblower i in Group.

APPENDIX C

ISBS INSTALLATION

Intelligent Sootblowing System
Installation
Version 2.1

GENERAL

The paper contains a general description of the Intelligent Sootblowing System software. The major components are shown in Figure 1.

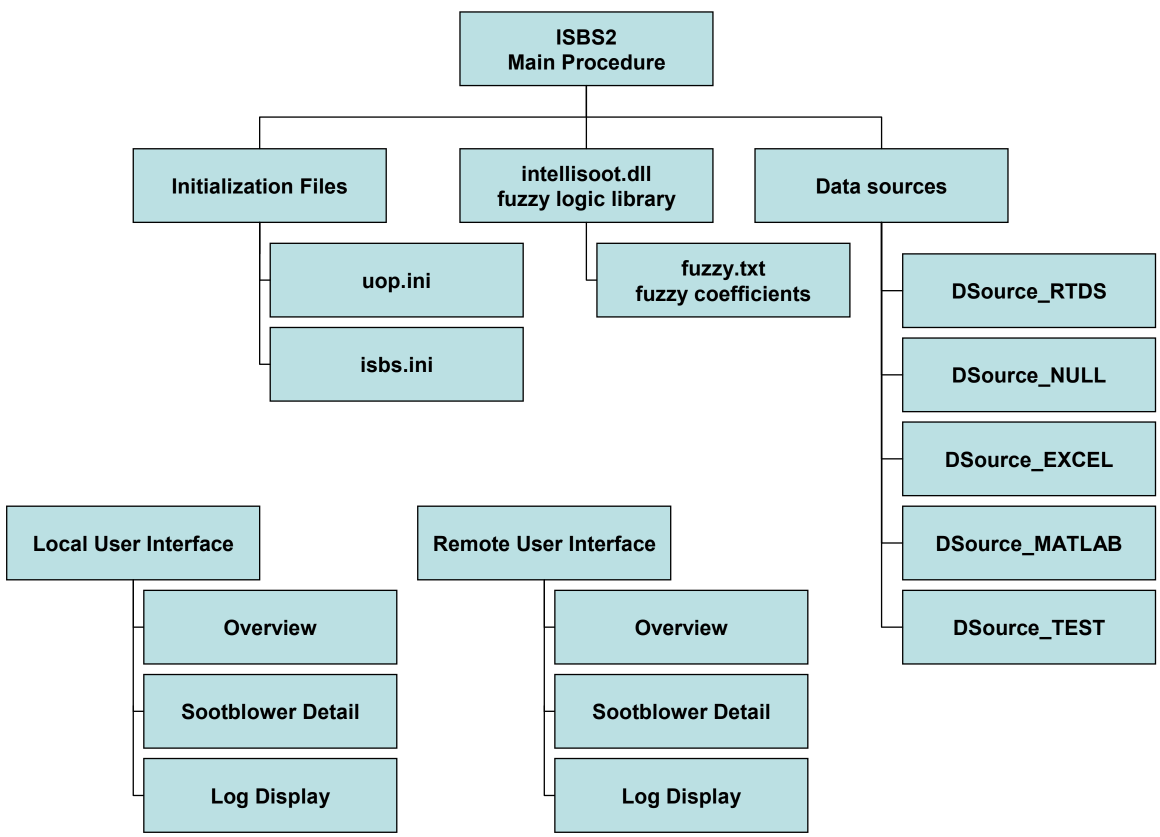


Figure 1. ISBS Overview

UOP INITIALIZATION FILE

```
[General]
sRTDSHost = HostComputerID
sLoadTag = "4CP001_I:MAIN_4.PNT_3"

[ISBS]
sIniFile = c:\rtds\isbs\isbs.ini
```

Figure 2. ISBS Entry in the UOP Initialization File

Description

Contains settings for various packages for the optimization programs. This file is located in the Windows root directory (C:\WINDOWS on most systems).

Layout

Section: General

sRTDSHost: Default RTDS host platform - where the RTDS resides. Should be a string. You should not need the domain name.

sLoadTag: Default tag name for the load parameter in the RTDS.

Section: ISBS

Section reserved for the ISBS system.

sIniFile: File to be used as initialization file for ISBS system.

ISBS INITIALIZATION FILE

```
[General]
sRTDSHost = HostComputerID
sLoadTag = "4CP001_I:MAIN_4.PNT_3"
sWorkingDirectory = c:\ISBS

iCycleTime = 20
dFilterConstant1 = 0.8
dFilterConstant2 = 0.8
dLevel1 = 0.7

iLogLevel = 5
sLogFile = c:\temp\isbs_log.txt
iLogToFile = 1
```

Figure 3. ISBS Initialization File

Description

Contains initialization settings for the ISBS system.

Layout

Section: General

sRTDSHost: Specific RTDS host platform - where the RTDS resides. If not defined, defaults to that in the UOP initialization file. Should be a string. You should not need the domain name. To use this driver, the RTDS COM object must be installed on the computer where ISBS is running. Special cases are provided for debugging:

Value	Description
NULL	Inputs are zero and outputs are discarded.
EXCEL	The ISBS program interfaces with EXCEL instead of the RTDS (not implemented yet).
MATLAB	The ISBS program interfaces with MATLAB instead of the RTDS. You must have MATLAB to run in this mode.
TEST	Constant test values.

sLoadTag: Specific tag to be used as the load index. If not defined, defaults to that in the UOP initialization file.

sWorkingDirectory: Specifies location of *fuzzy.txt* file.

iCycleTime: Interval (in seconds) program updates, including reading input data, processing, and outputting data. Suggested range is 20 to 120.

dFilterConstant1: Constant used to filter inputs. Filtered inputs are defined as:

$$X_f = X_f * dFilterConstant1 + (1-dFilterConstant1) * X$$

where X are the unfiltered values read from the data source and X_f are the filtered inputs. If *dFilterConstant1* = 0, there is no filtering. Permissible range: [0, 1].

dFilterConstant2: Constant used to filter output recommendations. Filtered outputs are defined as:

$$Y_f = Y_f * dFilterConstant2 + (1-dFilterConstant2) * Y$$

where Y are the unfiltered recommendations and Y_f are the filtered recommendations. The filtered recommendations are stored to the RTDS and displayed. If *dFilterConstant2* = 0.0, there is no filtering. Permissible range: [0, 1].

dLevel1: Sets threshold for recommendation to blow. Permissible range: [0, 1]. Typical value is 0.7. If 0.0, the recommendation is always to activate the sootblower group. As *dLevel1* is increased to 1.0, the recommendation to activate the sootblower group becomes less likely.

iLogLevel: Sets threshold for diagnostic messages to log windows and file. If message level is less than or equal to *iLevel1*, then the message is displayed. Otherwise, the message is ignored. In most instances, 0 (the default) is the correct setting, allowing only major status messages to be produced. If set to –1, all diagnostic messages are ignored. Permissible range: [-1, 255].

sLogFile: Sets the default log file for the program. The default is *c:\temp\isbs_log.txt*. The log file can be changed through the log window. If the file exists, the messages are appended to the end of the file. If the file does not exist, a file is created.

iLogToFile: If set to 0, does not log messages to file. If set to 1, logs messages to log file.

USER INTERFACE – MAIN CONSOLE OVERVIEW

Description

This is the primary interface for the user. This console must be started to start the ISBS engine. If this window is closed, the ISBS process is terminated unless the ISBS engine has been locked or there are other consoles (main or operator) attached to the ISBS engine. Increased functionality over the operator console includes the connections indicator, the ability to lock the ISBS engine process into memory preventing normal termination, and the ability to terminate the ISBS process regardless of connections to the ISBS engine.

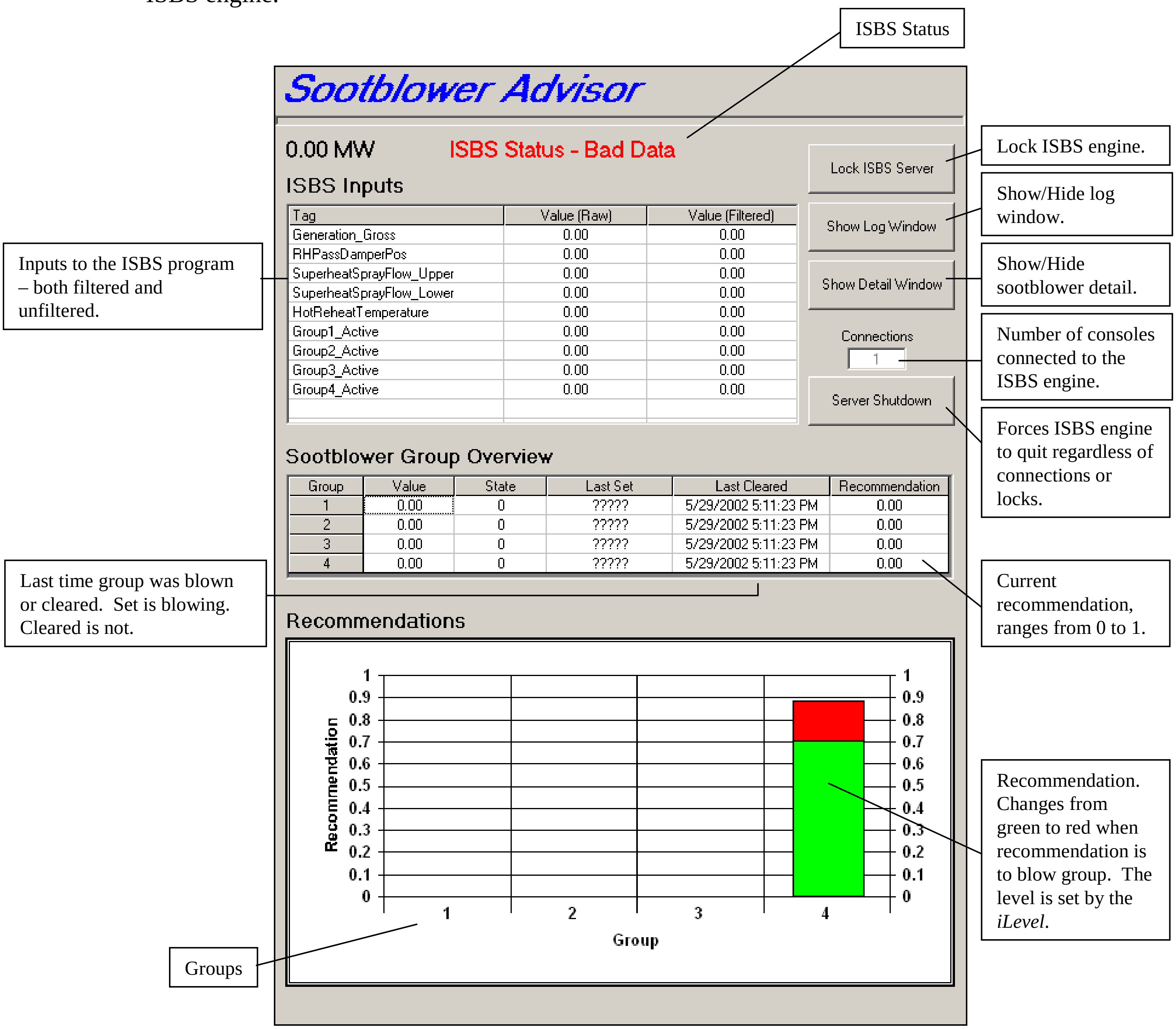


Figure 4. ISBS Main Console Overview

USER INTERFACE – OPERATOR CONSOLE OVERVIEW

Description

This is the primary interface for the operator console. This console can be used to start the ISBS engine. If this window is closed, the ISBS process is terminated unless the ISBS engine has been locked from the main console or there are other consoles (main or operator) attached to the engine.

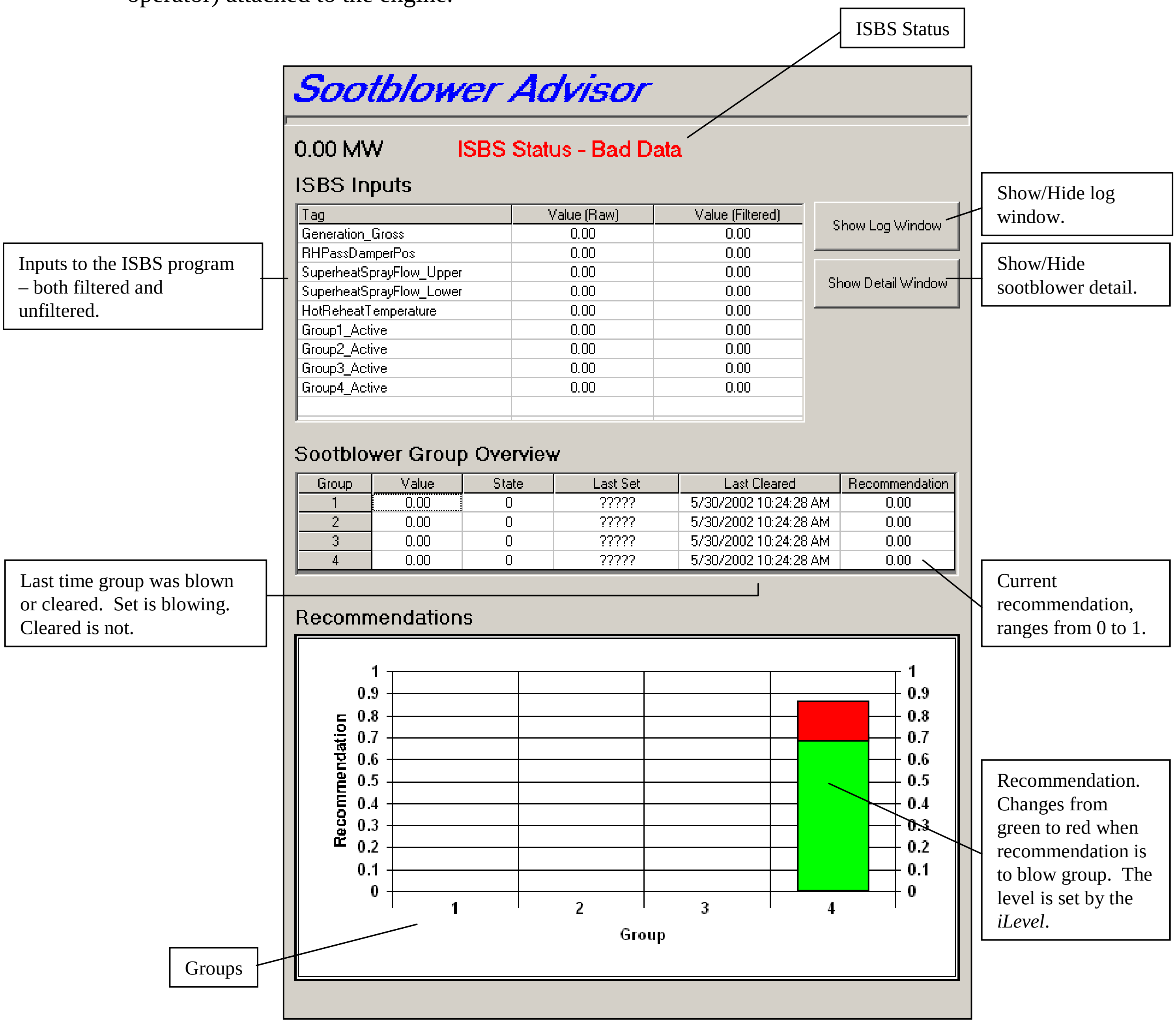


Figure 5. ISBS Main Console Overview

USER INTERFACE – SOOTBLOWER DETAIL

Description

This display shows the status of all sootblowers.

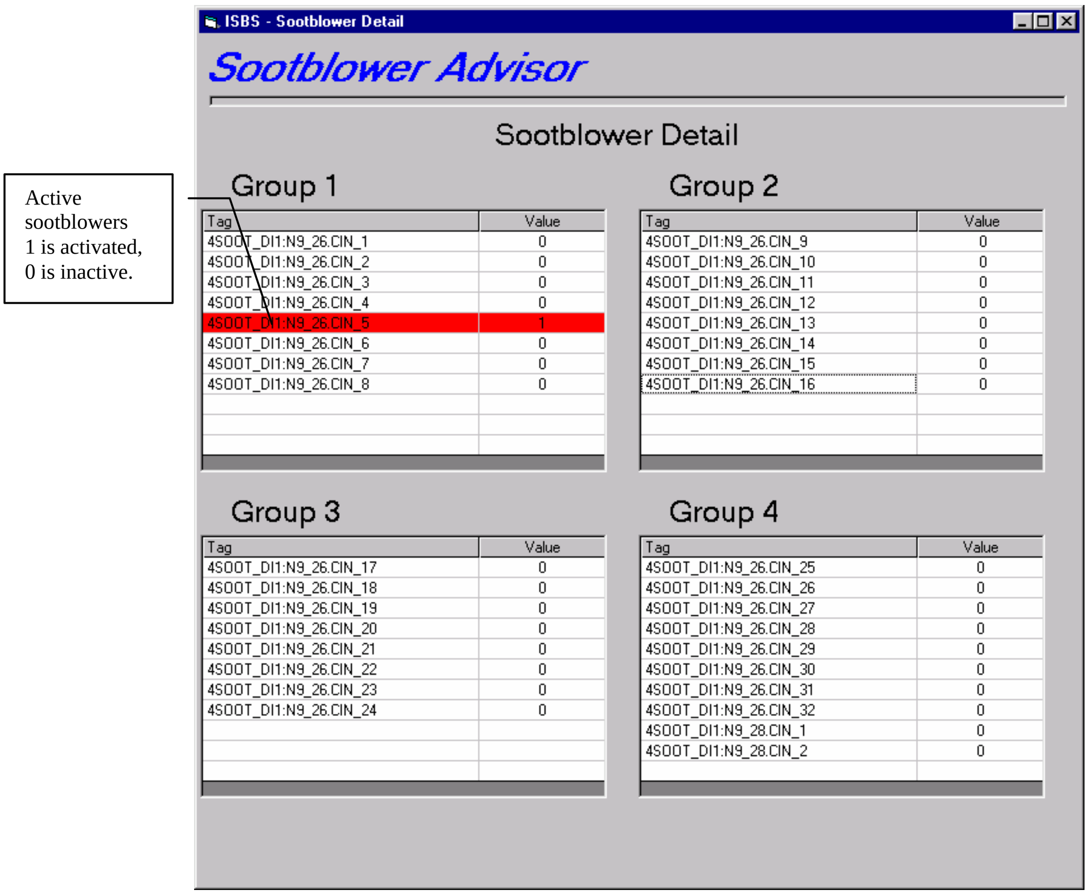


Figure 6. Sootblow Detail

USER INTERFACE – MESSAGE LOG WINDOW

Description

This interface is a scrolling display of diagnostic messages generated by the ISBS.

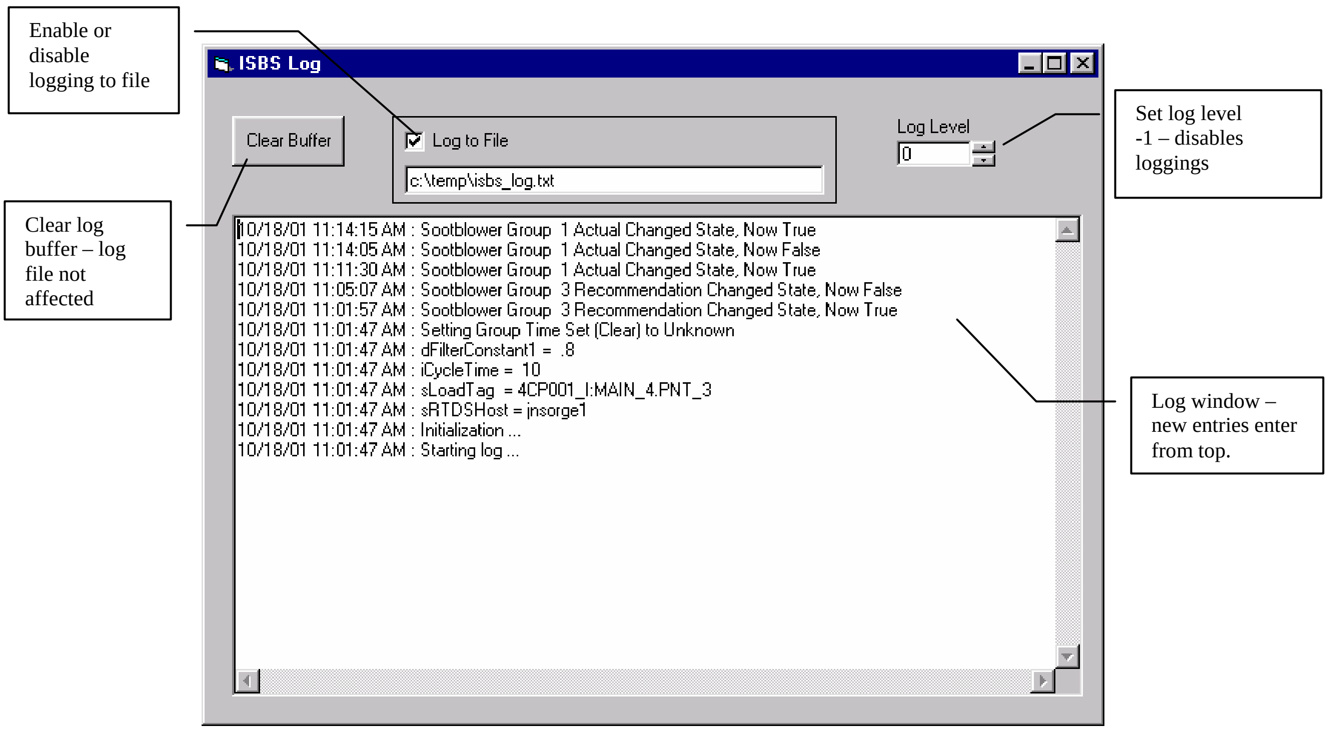


Figure 7. Log Display

INTELLISOOT.DLL AND FUZZY.TXT

Description

Intellisoot.dll is the dynamic load library that implements the fuzzy logic rule base for generating recommendations. The file *fuzzy.txt* contains coefficients for adjusting trapeziums defining the fixed fuzzy rules. Detailed information on the rule set can be found in the document *Intelligent Sootblowing System at Plant Hammond*.

DATA SOURCES

Description

The software is designed so that the data source can be changed. The driver *DSource_RTDS*, an interface to the RTDS, will typically be used. Other drivers have been developed for testing purposes including an interface to MATLAB and a null source interface.

REQUIRED FILES AND LOCATIONS

UOP.ini **-** UOP initialization file. It must be located in the Windows system directory.

ISBS.ini **-** ISBS initialization file. The fully qualified path is specified in UOP.ini.

ISBS_Log.txt **–** ISBS log file. The fully qualified path is specified in the ISBS initialization file or can be changed from the user panels.

intellisoot.dll **–** The ISBS fuzzy rule interpreter. It should be placed in the same directory as the ISBS executable.

fuzzy.txt **–** The fuzzy coefficients used by intellisoot.dll. This file should be in the same directory as the ISBS executable.

A possible directory structure would look like the following:

```
c:\
    \windows
        uop.ini
    \temp
        isbs_log.txt
    \rtds
        \isbs
            intellisoot.dll
            fuzzy.txt
            isbs.ini
```

APPENDIX D

UNIT OPTIMIZATION SOFTWARE

Unit Optimization Software

Version 2003.05.30

GENERAL

The paper contains a general description of the Unit Optimization software and its functionality. The major components are shown in Figure 1.

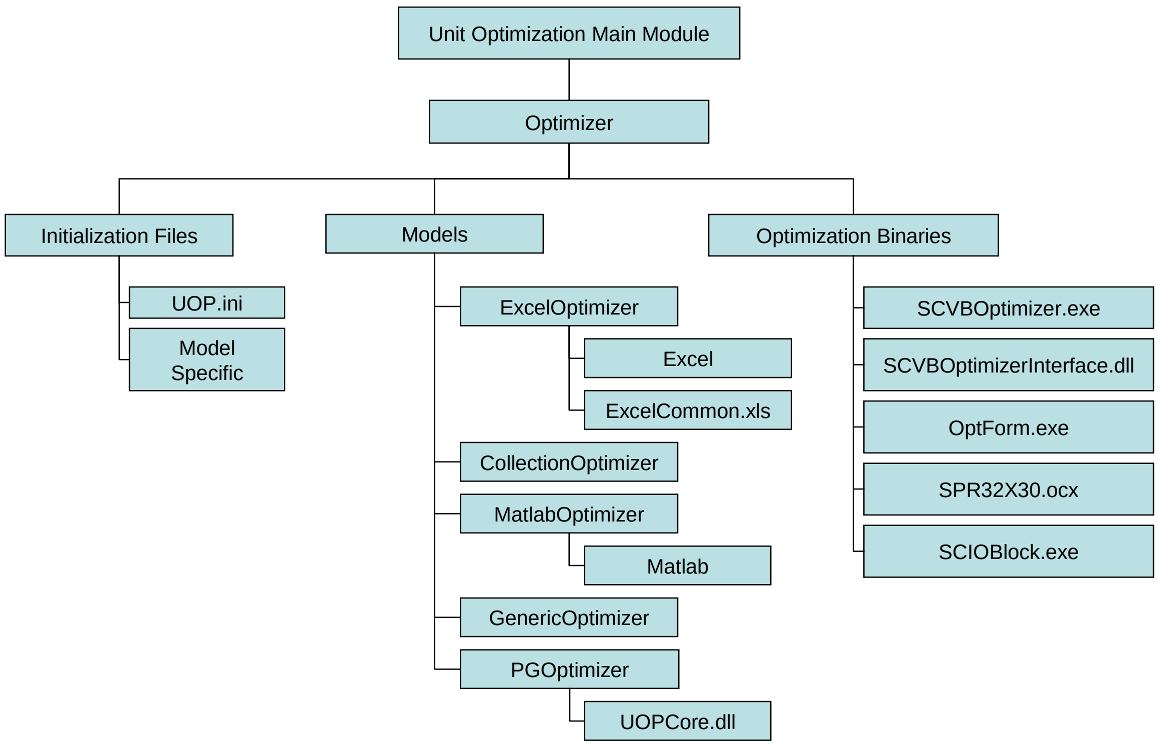


Figure 1. UOP Overview

INSTALLATION INSTRUCTIONS

The majority of the files included in the unit optimization software are COM compliant binaries requiring steps for system registration.

Installation of the unit optimization binaries requires the following steps:

- Copy the files to the desired directory. Ensure that the **UOPCore.dll** file is in the same directory as **SCVBOptimizer.exe**.
- Open a command line prompt and register the COM binaries as follows:
  SCVBOptimizer /regserver
  SCIOBlock /regserver
  OptForm /regserver
  regsvr32 SCVBOptimizerInterface.dll
  regsvr32 spr32x30.ocx
- Copy or create the **UOP.ini** file, and create a system environment variable named *UOPINI* and assign to it the fully qualified path and filename of the **UOP.ini** file.

UOP INITIALIZATION FILE

```
[General]
sRTDSHost = 127.0.0.1
sLoadTag = "4CP001_I:MAIN_4.PNT_3"

[ISBS]
IniFile = C:\UOP_Software\ISBS\isbs.ini

[OLEC]
OLECHOME = C:\UOP_Software\OLEC\lib

[ActiveOptimizers]
ExcelOptimizer1
NOX

[NOX]
DEBUG=1
COMID = "VBOptimizers.VBOptimizer"
DATA = "Excel|C:\UOP\Data\NOX.xls"

[EFF]
DEBUG=1
COMID = "VBOptimizers.VBOptimizer"
DATA = "Excel|C:\UOP\Data\EFF.xls"

[ExcelOptimizer1]
DEBUG=1
COMID = "VBOptimizers.VBOptimizer"
DATA = "Excel|C:\UOP\Data\Optimizer1.xls"
```

Figure 2. UOP Initialization File

Description

Contains settings for various packages of the optimization programs. This file is located in the directory assigned to the environment variable UOPINI.

Layout

Section: General

sRTDSHost: Default RTDS host platform - where the RTDS resides. Should be a string. You should not need the domain name. IP address can be substituted for machine name.

sLoadTag: Default tag name for the load parameter in the RTDS.

Section: ISBS
Contains information about the ISBS system. Please refer to the ISBS documentation.

Section: OLEC
Contains information about the online error correction system. Please refer to the OLEC documentation.

Section: ActiveOptimizers
Contains a list of the optimizers that are to be loaded upon the launch or reset of the unit optimization modules. Each optimizer listed is enumerated and loaded as a separate VBOptimizer object into memory by the root optimization process. The names correspond to proceeding sections that provide more detail for specific models. For example, the entry *ExcelOptimizer1* represents the model that can be described by the *ExcelOptimizer1* section found later in the initialization file. In the figure above, two models will be loaded: *ExcelOptimizer1* and *NOX*.

Section(s): Model Descriptors (ex., *ExcelOptimizer1*, *NOX*, *EFF*)
Each model descriptor sections contains information specific to the model types listed in the *ActiveOptimizers* section.

DEBUG: If the value is 1, available graphical user interfaces for the model are exposed so that the user can view the model data during calculations. If the value is 0, no interfaces are displayed. Graphic interfaces vary depending on model base type. The default is 0.

COMID: This is the COM object string identifier for the COM object that will be instantiated upon model creation. Typically, only **VBOptimizers.VBOptimizer** will be used. However, flexibility has been given to allow different COM binaries to be used as optimizers in the future (they must comply with the **SCVBOptimizerInterface** specification).

DATA: This is a string interpreted by the COM object created from *COMID* that describes the model. For VBOptimizers.VBOptimizer, this string takes the following form (note the two arguments are separated by a vertical hash):

> *<arg1>* | *<arg2>*
>
> *arg1*: This describes the type of model that is to be encapsulated by the VBOptimizer wrapper. It can have the following values: Excel, Generic, Matlab, PGCollection, or Collection.

arg2: This string describes additional data required by the model, and is model specific. For example, an *Excel* model would require *arg2* contain the fully qualified path and filename to the workbook that contains the model calculations.

SCVBOPTIMIZERS – MODELS

Description

The VBOptimizer acts as a wrapper object for one of five different types of models: Excel, Matlab, Collection, PGCollection, and Generic. For an optimizer object to be functional under the VBOptimizer wrapper, it must implement the SCVBOptimizerInterface. This ensures that future models developed will be consistent in their binary signature for COM-compliant optimization routines that use these objects. Please refer to the *UOP API documentation* for further information on the SCVBOptimizerInterface and SCVBOptimizer.

SCVBOptimizer Model Types

Excel
This type uses Microsoft's Excel as the optimization interface. Calculations can be entered into a spreadsheet and automated through the VBOptimizer interface. In order for a spreadsheet to be used in the VBOptimzer Excel model type, it must have two macro functions in the workbook. The *LocalRun* routine is called by the optimizer container when the model is run. This routine should contain any other calculations necessary to complete a model run (from the spreadsheet). The *LocalOptimize* routine is called when the VBOptimizer calls its optimization routine, and should contain any steps necessary to complete the optimization of the contained spreadsheet model. To use the Excel optimizer, the **ExcelCommon.xls** file must be installed in the directory where the **SCVBOptimizer.exe** file is located. This file is necessary when using the Excel Solver tool. During development, a significant problem was discovered when moving from Excel 97 to Excel XP and vise-versa. The solution required two versions of the **ExcelCommon.xls**. Depending on the version of Excel installed on the local computer (where **SCVBOptimizer.exe** resides), the appropriate file, **ExcelCommon97.xls** or **ExcelCommonXP.xls**, is renamed to **ExcelCommon.xls** for interoperating with Excel 97 or Excel XP, respectively.

> *arg2*: The fully qualified path and filename to the workbook that contains the model calculations.

Generic
This model is currently not used, but is intended to provide a template for future models that will be coded in Visual Basic, and contained within the SCVBOptimizer.exe binary.

> *arg2*: NA

Matlab
This model uses Mathsoft's Matlab numeric software as the main calculation engine. Similar to the Excel model type, a worksheet is set in a Matlab script and solved via automation from the SCVBOptimizer object.

arg2: The fully qualified path and filename to the Matlab worksheet that contains the model calculations.

Collection

This model may contain an agglomeration of several VBOptimizer objects. The intention of this type is to create a set of models that interact with each other to provide the optimum point among all models. This type may have an established algorithm that resolves conflicts among submodels. It can also be used as simply a container for multiple model sets. Collections may contain collections.

arg2: The name of the section that lists names of the optimizer models that will be created and placed in the collection. The names of the optimizer models correspond to other sections in the UOP.INI which describe the individual models.

PGCollection

This is a special version of the Collection model that uses the PowerGen optimization routine to resolve conflicting optimum points among the contained models. Using this model type requires the **UOPCore.dll**. Otherwise, it is similar to the Collection model.

arg2: The name of the section that lists names of the optimizer models that will be created and placed in the collection. The names of the optimizer models correspond to other sections in the UOP.INI which describe the individual models.

USER INTERFACE – SCVBOPTIMIZER

Description

This simple interface allows the user to view the current inputs, outputs, cost, and suboptimizers that are applicable to the current model being executed.

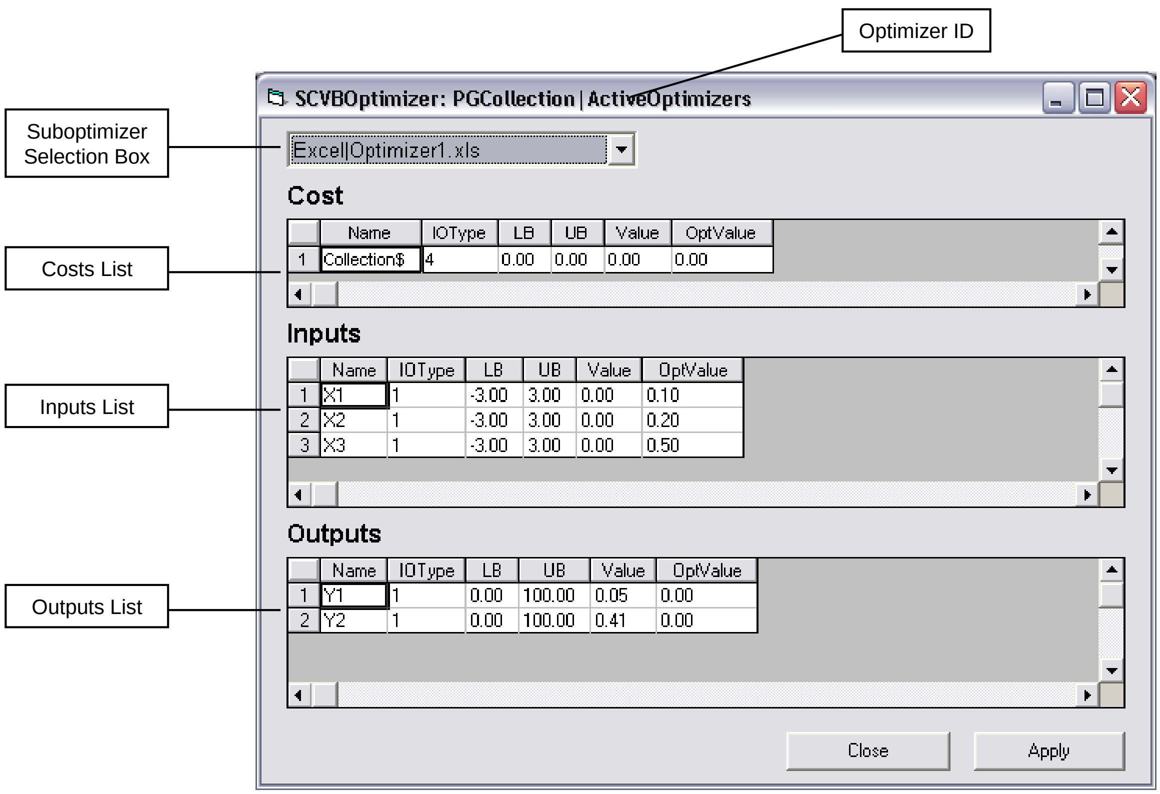


Figure 3. SCVBOptimizer User Interface

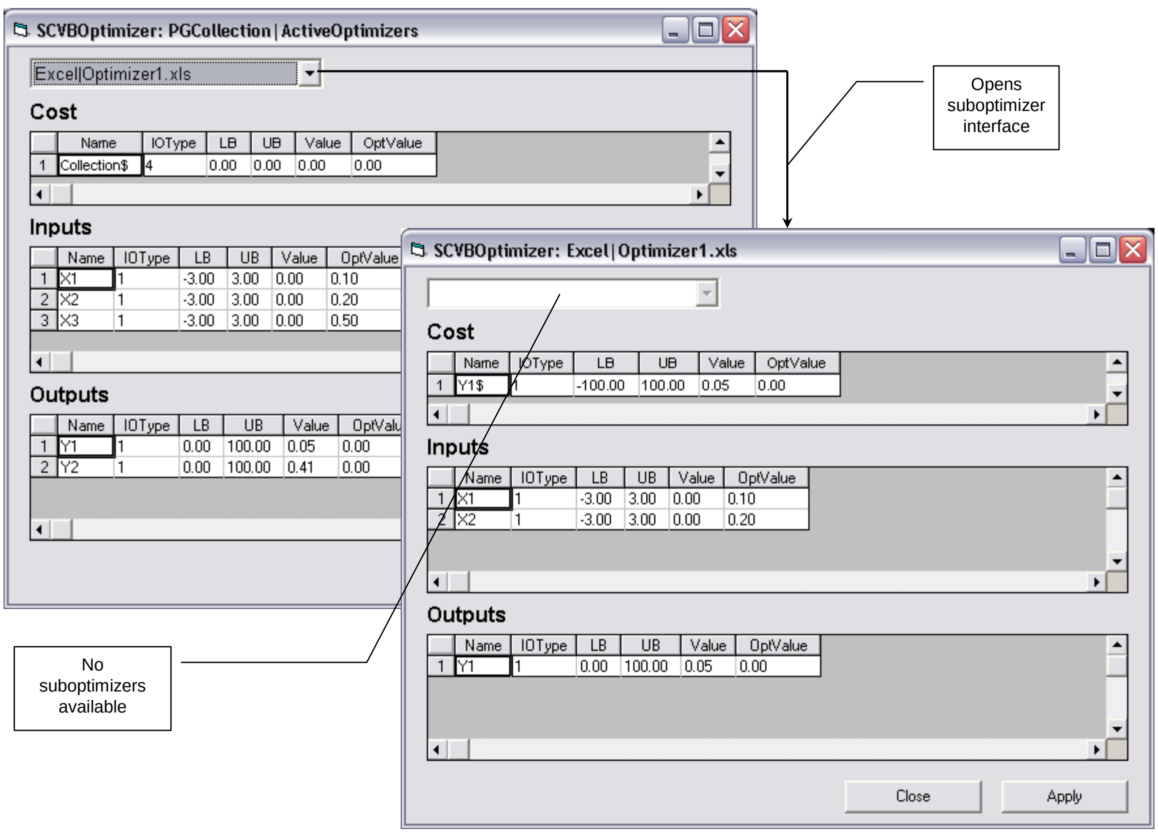


Figure 4. UOP User Interface

REQUIRED FILES AND LOCATIONS

UOP.ini **-** UOP initialization file. It must be located in the directory set by the UOPINI environment variable.

SCVBOptimizer.exe **–** The binary containing the out-of-process COM components and the different optimization models. It must be registered using the /regserver switch from the command line.

SCVBOptimizerInterface.dll **–** The binary containing the COM interface supported by the SCVBOptimizer and its contained models. This module may be used to enforce a standard interface for the submodels. It must be registered using the Windows system regsvr32.exe program.

SCIOBlock.exe **–** The binary containing the common input and output objects used to marshal data among optimizer processes.

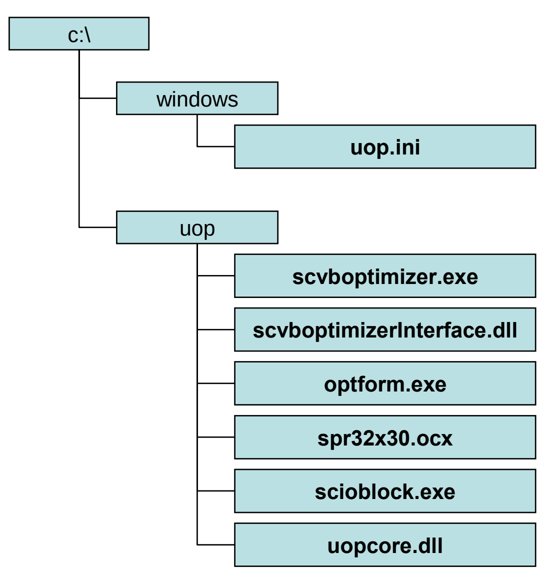
OptForm.exe **–** The binary containing the user interface for the SCVBOptimizer objects. This is also a COM out-of-process server that must be registered using the /regserver switch from the command line.

Spr32x30.ocx **–** The third-party spreadsheet grid control runtime implemented in the graphic user interface. It must be registered using the Windows system regsvr32.exe program.

ExcelCommon.xls **–** This spreadsheet is needed for running the Excel Solver add-in via the SCVBOptimizer objects. It is a copy of either **ExcelCommon97.xls** or **ExcelCommonXP.xls**, depending on the version of Excel residing on the workstation.

Other miscellaneous files **–** Other miscellaneous files include: Excel workbooks, text files (ini files), and MATLAB worksheets. These files are optimizer model specific.

A possible directory structure would look like the following:



APPENDIX E

SCVBOPTIMIZER

SCIOBlock API

v.2003.05.01

<u>SCInputs</u> (groups of Inputs)

CopyFrom(ByVal source As SCInputs**)**
Wholesale replacement of SCInputs, regardless of name or index, from source object to current object.

CopyTo(ByVal dest As SCInputs**)**
Wholesale replacement of SCInputs, regardless of name or index, from current object into dest object.

setSCInputs(ByVal source As SCInputs**)**
Replacement of SCInput objects that have matching names from source into current object.

getSCInputs(ByVal dest As SCInputs**)**
Replacement of SCInput objects that have matching names from current object into dest object.

setValuesByName(ByRef Names() as String, ByRef Values() as Double**)**
Sets the Value properties of the contained internal SCInput objects directly by locating the SCInput associated with each entry in the Names() array. If the entry in Names() is not located, it is ignored and operation continues to the next entry in the Names() array.

getValuesByName(ByRef Names() as String, ByRef Values() as Double**)**
Retrieves the Value properties of the contained internal SCInput objects directly by locating the SCInput associated with each entry in the Names() array. The values are stored and thus returned in the Values() array. If an SCInput cannot be found for the associated entry in the Names() array, the entry in the Values() array is not set, and operation continues to the next index in the Names() array.

getNamesAndValues(ByRef Names() as String, ByRef Values() as Double**)**
Retrieves all internal SCInput objects' Names and Values properties, and stores them in the Names() and Values() arrays. After operation is completed successfully, the Names() and Values() arrays' sizes should correspond directly to the number of SCInput objects.

Long **getInputIndex(**ByVal Name as String**)**
Returns the index of the internal SCInput object associated with Name.

Long **getCount()**
Returns number of contained SCInput objects.

SCOutput **getInputByName(**ByVal Name As String**)**
Returns the SCInput object associated with Name.

SCOutput **getInputByIndex(**ByVal index As Long**)**
Returns the SCInput object associated with the internal location index.

Long **setInputByIndex(**ByVal index As Long, ByVal o As SCInput**)**
Returns a number indicating how many SCInputs were set successfully in the location index (typically, 1 means success and 0 mean failure).

Long **setInputByName(**ByVal name As String, ByVal o As SCInput**)**
Returns a number indicating how many SCInputs were set successfully for the named SCInput (typically, 1 means success and 0 mean failure).

Long **Create(**ByVal totalOutputs As Long**)**
Creates totalOutputs empty internal SCInput objects.

Destroy()
Deletes the internal SCInput objects array.

Append(ByVal oInputs as SCInputs**)**
Adds the SCInput objects contained in SCInputs to the end of the internal SCInput object array.

Clean()
Combines the contained SCInput objects by name, appropriately changing the lower bound, upper bound, and value attribute of each non-unique (by name) SCInput.

Add(ByVal oInput as SCInput**)**
Adds the oInput SCInput object to the end of the internal SCInput object array.

Merge(ByVal source as SCInputs**)**
Appends the SCInputs to the internal SCInput object array, and subsequently calls **Clean()** to resolve duplicate entries.

<u>SCOutputs</u> (groups of outputs)

CopyFrom(ByVal source As SCOutputs**)**
Wholesale replacement of SCOutputs, regardless of name or index, from source object to current object.

CopyTo(ByVal dest As SCOutputs**)**

Wholesale replacement of SCOutputs, regardless of name or index, from current object into dest object.

setSCOutputs(ByVal source As SCOutputs**)**
Replacement of SCOutput objects that have matching names from source into current object.

getSCOutputs(ByVal dest As SCOutputs**)**
Replacement of SCOutput objects that have matching names from current object into dest object.

setValuesByName(ByRef Names() as String, ByRef Values() as Double**)**
Sets the Value properties of the contained internal SCOutput objects directly by locating the SCOutput associated with each entry in the Names() array. If the entry in Names() is not located, it is ignored and operation continues to the next entry in the Names() array.

getValuesByName(ByRef Names() as String, ByRef Values() as Double**)**
Retrieves the Value properties of the contained internal SCOutput objects directly by locating the SCOutput associated with each entry in the Names() array. The values are stored and thus returned in the Values() array. If an SCOutput cannot be found for the associated entry in the Names() array, the entry in the Values() array is not set, and operation continues to the next index in the Names() array.

getNamesAndValues(ByRef Names() as String, ByRef Values() as Double**)**
Retrieves all internal SCOutput objects' Names and Values properties, and stores them in the Names() and Values() arrays. After operation is completed successfully, the Names() and Values() arrays' sizes should correspond directly to the number of SCOutput objects.

Long **getOutputIndex(**ByVal Name as String**)**
Returns the index of the internal SCOutput object associated with Name.

Long **getCount()**
Returns number of contained SCOutput objects.

SCOutput **getOutputByName(**ByVal name As String**)**
Returns the SCOutput object associated with *name*.

SCOutput **getOutputByIndex(**ByVal index As Long**)**
Returns the SCOutput object associated with the internal location, *index*.

Long **setOutputByIndex(**ByVal index As Long, ByVal o As SCOutput**)**
Returns a number indicating how many SCOutputs were set successfully in the location index(typically, 1 means success and 0 mean failure).

Long **setOutputByName(**ByVal name As String, ByVal o As SCOutput**)**
Returns a number indicating how many SCOutputs were set successfully for the named SCOutput (typically, 1 means success and 0 mean failure).

Long **Create(**ByVal totalOutputs As Long**)**
Creates totalOutputs empty internal SCOutput objects.

Destroy()
Deletes the internal SCOutput objects array.

Append(ByVal oOutputs as SCOutputs**)**
Adds the SCOutput objects contained in SCOutputs to the end of the internal SCOutput object array.

Clean()
Combines the contained SCOutput objects by name, appropriately changing the lower bound, upper bound, and value attribute of each non-unique (by name) SCOutput

Add(ByVal oOutput as SCOutput**)**
Adds the oOutput SCOutput object to the end of the internal SCOutput object array.

Merge(ByVal source as SCOutputs**)**
Appends the SCOutputs to the internal SCOutput object array, and subsequently calls **Clean()** to resolve duplicate entries.

<u>SCOutput/SCInput/SCCost</u>

***Note: The SCInput API is presented here, but is easily applied to the SCOutput and SCCost APIs by replacing "SCInput" with "SCCost" or "SCOutput", as appropriate.*

getSCInput(ByVal dest As SCInput**)**
Fills the dest SCInput object with the current object's values.

setSCInput(ByVal source As SCInput**)**
Sets the SCInput object with the source SCInput's object values.

Property **Name()** As String
Sets/retrieves the name of the SCInput

Property **ioType()** As scIOType (read-only)
Sets/retrieves the ioType of the current object. See enum for values.

<u>scIOType</u>
ioNOTUSED = 0

ioCONTROL = 1
ioSTATE = 2
ioOUTPUT = 3
ioCOST = 4

Property **UB()** As Double
Sets/retrieves the upperbound value for the SCInput.

Property **LB()** As Double
Sets/retrieves the lowerbound value for the SCInput.

Property **OptValue()** As Double
Sets/retrieves the optimum value for the SCInput.

Property **Value()** As Double
Sets/retrieves the current value of the SCInput.

SCVBOptimizer API

v.2003.05.01

LoadData(ByVal datastring As String**)**
Customized for each SCOptimizer type.

Long **Function Run()**
Returns 0 if successful; non-zero otherwise.

Long **Optimize()**
Returns 0 if successful; non-zero otherwise.

Long **getReachedMin()**
Returns 1 if solution is at minimum; 0 otherwise.

setInputByName(ByVal name As String, ByVal oInput As SCInput**)**
Replaces properties of SCInput (id is name) with oInput properties.

setInputByIndex(ByVal index As Long, ByVal oInput As SCInput**)**
Replaces SCInput at the index position with oInput.

SCInput **getInputByName(**ByVal name As String**)**
Returns the SCInput associated with name.

SCInput **getInputByIndex(**ByVal index As Long**)**
Returns the SCInput associated with index.

setOutputByName(ByVal name As String, ByVal oOutput As SCOutput**)**
Replaces SCOutput (id is *name*) with values from *oOutput*.

setOutputByIndex(ByVal index As Long, ByVal oOutput As SCOutput**)**
Replaces SCOutput in the *index* slot with the values of *oOutput*.

SCOutput **getOutputByName(**ByVal name As String**)**
Returns an SCOutput with the name, *name*.

SCOutput **getOutputByIndex(**ByVal index As Long**)**
Returns an SCOutput associated with *index*.

SCCost **getCost()**
Returns a copy of the internal SCCost object.

SCOutputs **getOutputs()**
Returns a copy of the internal SCOutputs block.

SCInputs **getInputs()**
Returns a copy of the internal SCInputs block.

setInputs(ByVal oInputs As SCInputs**)**
Sets the internal SCInputs block to oInputs.

setOutputs(ByVal oOutputs As SCOutputs**)**
Sets the internal SCOutputs block to oOutputs

setCost(ByVal oCost as SCCost**)**
Sets the internal SCCost block to oCost

getAll(oInputs As SCInputs, oOutputs As SCOutputs, oCost As SCCost**)**
Retrieves a copy of the internal SCInputs block, SCOutputs block, and SCCost into oInputs, oOutputs, and oCost, respectively.

setAll(ByVal oInputs As SCInputs, ByVal oOutputs As SCOutputs, ByVal oCost As SCCost**)**
Sets the internal SCInputs block, SCOutputs block, and SCCost from oInputs, oOutputs, and oCost, respectively.

Long **getNoOfOutputs()**
Returns the number of outputs in the current optimizer.

Long **getNoOfInputs()**
Returns the number of inputs in the current optimizer.

Boolean **IsValid()**
Returns TRUE if the VBOptimizer is valid.

Boolean **IsVisbile()**
Returns TRUE if the VBOptimizer interface form is visible.

setDebugState(ByVal DebugState As Long**)**
Sets the debug level of the current optimizer (typically, 0=off, 1=on).

Long **getDebugState()**
Retrieves the current debug level of the current optimizer.

Show()
Displays the optimizer's user interface.

Hide()
Hides the optimizer's user interface.

String **getName()**

Retrieves the name of the current optimizer.

setName(ByVal name As String**)**

*This function is not implemented in the current version, but is reserved for possible future use. The optimizer name is, instead, the string passed to the optimizer by the **LoadData** function.*

CreateSubOpts(ByVal nOpts As Long**)**

Creates an internal array of optimizers, to be used as suboptimizers. Should be implemented in collection optimizers only.

DestroySubOpts()

Destroys the internal array of suboptimizers. Should be implemented in collection optimizers only.

Long **getSubOptCount()**

Returns the number of internal suboptimizers.

SCVBOptimizer **getSubOptByIndex(**ByVal index As Long**)**

Returns a reference to the suboptimizer located at the position *index* in the internal suboptimizer array.

SCVBOptimizer **getSubOptByName(**ByVal name As String**)**

Returns a reference to the suboptimizer labeled *name*.

setSubOptByIndex(ByVal index As Long, ByVal oSubOpt As SCVBOptimizer**)**

Sets a reference to the suboptimizer *oSubOpt* at location *index* in the internal suboptimizer array.

setSubOptByName(ByVal name As String, ByVal oSubOpt As SCVBOptimizer**)**

Sets a reference to the optimizer oSubOpt with the label *name* in the internal suboptimizer array.

Boolean **IsCollection()**

Returns TRUE for optimizers which contain suboptimizers; returns FALSE otherwise.

Long **getStatus()**

Returns 0 if ok; non-zero otherwise.

scOptimizerType **getType()**

Returns scOptimizerType (Long).

<u>scOptimizerType</u>

scNone = 0
scExcel = 1

scMatlab = 2
scGeneric = 3
scPowerGen = 4
scSynengco = 5
scCollection = 6

UpdateForm()

Updates the values in the display form for the current optimizer and all suboptimizers.

UpdateFromSubOpts()

Retrieves SCInputs, SCOutputs, and SCCost properties from the suboptimizers and replaces the objects of the internally stored SCIOBlocks. Valid only for optimizer collections.

UpdateSubOpts()

Copies the property values of the internally stored SCIOBlocks to the associated individual suboptimizers contained in the collection. Valid only for optimizer collections.

SetInputValues(ByRef Names() as String, ByRef Values() as Double**)**

Directly sets the Value property of each SCInput object in the contained optimizer.

Double **getCostValue()**

Returns the Value property of the SCCost object in the contained optimizer.

APPENDIX F

BOILER OPTIMIZER ACTIVE MODEL INFORMATION

```
ModelFileName = c:\gnocis\activemodel\ham_go8
ComboFileName = c:\gnocis\activemodel\U4_combo.ini
ConstFileName = c:\gnocis\activemodel\U4_const.ini
TagFileName = c:\gnocis\activemodel\U4_tag.ini
OutFileName = c:\gnocis\activemodel\U4_outputs.ini
Debug = 5
```

```
# File: Const.ini
#
# Version 1.0 - Original 5/21/2000 - JMF
#             - Hammond 4 Unit Optimization Project
#
#
# ----------------- CONTROL ---------------------

[WMILLAC]
SETPOINTINPUT = 1
BIAS% = 4FUELCONTROL:FDRDMD_BAL.OUT
BIAS%RANGE = 88000
GNOCISBIAS = 4GNOCIS:ROUTS.PNT_2
MINTAG = 4GNOCIS:FDRATRIM.LOLIM
MAXTAG = 4GNOCIS:FDRATRIM.HOLIM
CLAMPTAG = 4GNOCIS:CLAMPS.CO_3
LOCALREMOTETAG = 4FDRA:AFDR_STATION.MA
CUTOFF = 10000
OPERATINGMIN = 40000
OPERATINGMAX = 80000
#MOVEDELTA = 100
MOVECOST = 0.0002

[WMILLBC]
SETPOINTINPUT = 1
BIAS% = 4FUELCONTROL:FDRDMD_BAL.OUT
BIAS%RANGE = 88000
GNOCISBIAS = 4GNOCIS:ROUTS.PNT_3
MINTAG = 4GNOCIS:FDRBTRIM.LOLIM
MAXTAG = 4GNOCIS:FDRBTRIM.HOLIM
CLAMPTAG = 4GNOCIS:CLAMPS.CO_4
LOCALREMOTETAG = 4FDRB:BFDR_STATION.MA
CUTOFF = 10000
OPERATINGMIN = 40000
OPERATINGMAX = 80000
#MOVEDELTA = 100
MOVECOST = 0.0002

[WMILLCC]
SETPOINTINPUT = 1
BIAS% = 4FUELCONTROL:FDRDMD_BAL.OUT
BIAS%RANGE = 88000
GNOCISBIAS = 4GNOCIS:ROUTS.PNT_4
MINTAG = 4GNOCIS:FDRCTRIM.LOLIM
MAXTAG = 4GNOCIS:FDRCTRIM.HOLIM
CLAMPTAG = 4GNOCIS:CLAMPS.CO_5
LOCALREMOTETAG = 4FDRC:CFDR_STATION.MA
CUTOFF = 10000
OPERATINGMIN = 40000
OPERATINGMAX = 80000
#MOVEDELTA = 100
MOVECOST = 0.0002

[WMILLDC]
SETPOINTINPUT = 1
BIAS% = 4FUELCONTROL:FDRDMD_BAL.OUT
BIAS%RANGE = 88000
GNOCISBIAS = 4GNOCIS:ROUTS.PNT_5
MINTAG = 4GNOCIS:FDRDTRIM.LOLIM
```

```
MAXTAG = 4GNOCIS:FDRDTRIM.HOLIM
CLAMPTAG = 4GNOCIS:CLAMPS.CO_6
LOCALREMOTETAG = 4FDRD:DFDR_STATION.MA
CUTOFF = 10000
OPERATINGMIN = 40000
OPERATINGMAX = 80000
#MOVEDELTA = 100
MOVECOST = 0.0002

[WMILLEC]
SETPOINTINPUT = 1
BIAS% = 4FUELCONTROL:FDRDMD_BAL.OUT
BIAS%RANGE = 88000
GNOCISBIAS = 4GNOCIS:ROUTS.PNT_6
MINTAG = 4GNOCIS:FDRETRIM.LOLIM
MAXTAG = 4GNOCIS:FDRETRIM.HOLIM
CLAMPTAG = 4GNOCIS:CLAMPS.CO_7
LOCALREMOTETAG = 4FDRE:EFDR_STATION.MA
CUTOFF = 10000
OPERATINGMIN = 40000
OPERATINGMAX = 80000
#MOVEDELTA = 100
MOVECOST = 0.0002

[WMILLFC]
SETPOINTINPUT = 1
BIAS% = 4FUELCONTROL:FDRDMD_BAL.OUT
BIAS%RANGE = 88000
GNOCISBIAS = 4GNOCIS:ROUTS.PNT_7
MINTAG = 4GNOCIS:FDRFTRIM.LOLIM
MAXTAG = 4GNOCIS:FDRFTRIM.HOLIM
CLAMPTAG = 4GNOCIS:CLAMPS.CO_8
LOCALREMOTETAG = 4FDRF:FFDR_STATION.MA
CUTOFF = 10000
OPERATINGMIN = 40000
OPERATINGMAX = 80000
#MOVEDELTA = 100
MOVECOST = 0.0002

[YAOFAF1]
SETPOINTINPUT = 1
BIASOUT = 4OFAIRCDF1:CDF1STPT_SEL.OUT
GNOCISBIAS = 4GNOCIS:ROUTS1.PNT_1
MINTAG = 4GNOCIS:OFAF1TRIM.LOLIM
MAXTAG = 4GNOCIS:OFAF1TRIM.HOLIM
CLAMPTAG = 4GNOCIS:CLAMPS.CO_9
LOCALREMOTETAG = 4OFAIRCDF1:OFACDF1_PCTL.MA
#LIMITCLAMP = 1
MOVECOST = 10

[YAOFAR1]
SETPOINTINPUT = 1
BIASOUT = 4OFAIRCDR1:CDR1STPT_SEL.OUT
GNOCISBIAS = 4GNOCIS:ROUTS1.PNT_3
MINTAG = 4GNOCIS:OFAR1TRIM.LOLIM
MAXTAG = 4GNOCIS:OFAR1TRIM.HOLIM
CLAMPTAG = 4GNOCIS:CLAMPS.CO_11
LOCALREMOTETAG = 4OFAIRCDR1:OFACDR1_PCTL.MA
#LIMITCLAMP = 1
```

```
MOVECOST = 10

[YAOFAF2]
SETPOINTINPUT = 1
BIASOUT = 4OFAIRCDF2:CDF2STPT_SEL.OUT
GNOCISBIAS = 4GNOCIS:ROUTS1.PNT_2
MINTAG = 4GNOCIS:OFAF2TRIM.LOLIM
MAXTAG = 4GNOCIS:OFAF2TRIM.HOLIM
CLAMPTAG = 4GNOCIS:CLAMPS.CO_10
LOCALREMOTETAG = 4OFAIRCDF2:OFACDF2_PCTL.MA
#LIMITCLAMP = 1
MOVECOST = 10

[YAOFAR2]
SETPOINTINPUT = 1
BIASOUT = 4OFAIRCDR2:CDR2STPT_SEL.OUT
GNOCISBIAS = 4GNOCIS:ROUTS1.PNT_4
MINTAG = 4GNOCIS:OFAR2TRIM.LOLIM
MAXTAG = 4GNOCIS:OFAR2TRIM.HOLIM
CLAMPTAG = 4GNOCIS:CLAMPS.CO_12
LOCALREMOTETAG = 4OFAIRCDR2:OFACDR2_PCTL.MA
#LIMITCLAMP = 1
MOVECOST = 10

[AVG_O2]
SETPOINTINPUT = 1
#BIASOUT = 4O2CONTROL:O2SETPT_SEL2.OUT
BIASOUT = 4O2CONTROL:GNOCIS_BIAS.MEAS
GNOCISBIAS = 4GNOCIS:ROUTS.PNT_1
#OPERATORBIAS = 4O2CONTROL:O2SETPT_BIAS.BIAS
MINTAG = 4GNOCIS:O2TRIM.LOLIM
MAXTAG = 4GNOCIS:O2TRIM.HOLIM
CLAMPTAG = 4GNOCIS:CLAMPS.CO_1
LOCALREMOTETAG = 4O2CONTROL:O2_CTRLR.MA
#CLAMPABOVELOAD = 500
#LIMITCLAMP = 1
MOVEDELTA = .05
MOVECOST = 40000

[AVG_TSAAI]
CLAMP = 1

[AVG_TPAAI]
CLAMP = 1

[SCR_ON]
CLAMP = 1

[TMS_Setpoint]
CLAMP = 1

[THRH_Setpoint]
CLAMP = 1

[PMS_Setpoint]
CLAMP = 1

#
#------------------ OUTPUTS ---------------------
```

```
#
[NOX_LBMMBTU]
MINTAG = 4GNOCIS:NOXLIMS.LOLIM
MAXTAG = 4GNOCIS:NOXLIMS.HOLIM
USEFUZZY = 1
FUZZYCOEFF = 1000000
BIASADJUST = 1

[CIA]
MINTAG = 4GNOCIS:LOILIMTS.LOLIM
MAXTAG = 4GNOCIS:LOILIMTS.HOLIM
USEFUZZY = 1
FUZZYCOEFF = 1000000

[THRH]
#MINTAG = 4TURB:INRC_3
#MAXTAG = 4TURB:INRC_3
USEFUZZY = 1
FUZZYCOEFF = 0

[TMS]
#MINTAG = 4TURB:INRC_2
#MAXTAG = 4TURB:INRC_2
USEFUZZY = 1
FUZZYCOEFF = 0

[PMS]
#MINTAG = 4TURB:INRC_1
#MAXTAG = 4TURB:INRC_1
USEFUZZY = 1
FUZZYCOEFF = 0

[SH_SPRAY_FLOW_UPPER]
USEFUZZY = 1
FUZZYCOEFF = 0

[SH_SPRAY_FLOW_LOWER]
USEFUZZY = 1
FUZZYCOEFF = 0

[EFF]
MINTAG = 4GNOCIS:EFFLIMTS.LOLIM
MAXTAG = 4GNOCIS:EFFLIMTS.HOLIM
USEFUZZY = 1
#FUZZYWOMAX = 87.9
#FUZZYWOMIN = 87.9
FUZZYCOEFF = 1000000

[RH_DAMPER_POS]
USEFUZZY = 1
FUZZYCOEFF = 0

#
# ------------------- MISC ---------------------
#
LOADTAG = 4CP001_I:MAIN_4.PNT_3
NUMCONTROL = 17
NUMMODELOUT = 9
UPDATETAG = 4GNOCIS:LOGIC.BO01
```

```
CLOSEDLOOPTAG = 4GNOCIS:LOGIC.BI04
REMOVEBIASTAG = 4GNOCIS:LOGIC.BI06
MATRIXSIZE = 1
INTCPPORT = 2003
OUTTCPPORT1 = 2010
OUTTCPADDRESS1 = 127.0.0.1
OUTTCPPORT2 = 2021
OUTTCPADDRESS2 = 127.0.0.1
OUTTCPPORT3 = 2011
OUTTCPADDRESS3 = 127.0.0.1
OUTPUTMODEL = c:\gnocis\activemodel\ham_go_bias
```

4CP002_I:MAIN_10.PNT_1	MILL D COAL FLOW
4CP002_I:MAIN_11.PNT_1	MILL B COAL FLOW
4CP005_I:MAIN_5.PNT_1	DIVISION WALL INLET TEMPERATURE RH
4CP002_I:MAIN_10.PNT_2	MILL C COAL FLOW
4CP002_I:MAIN_11.PNT_2	MILL A COAL FLOW
4CP003_I:MAIN_6.PNT_2	OVERFIRE AIR DAMPER POSITION F1
4CP003_I:MAIN_7.PNT_2	OVERFIRE AIR DAMPER POSITION R1
4CP005_I:MAIN_4.PNT_2	DIVISION WALL INLET TEMPERATURE LH
4CP001_I:MAIN_1.PNT_3	MAIN STEAM PRESSURE
4CP002_I:MAIN_11.PNT_3	MILL F COAL FLOW
4CP005_I:MAIN_3.PNT_3	SUPERHEAT SPRAY FLOW (UPPER)
4CP005_I:MAIN_3.PNT_4	SUPERHEAT SPRAY FLOW (LOWER)
4CP003_I:MAIN_6.PNT_5	OVERFIRE AIR DAMPER POSITION F2
4CP003_I:MAIN_7.PNT_5	OVERFIRE AIR DAMPER POSITION R2
4CP001_I:MAIN_3.PNT_7	HOT REHEAT PRESSURE
4CP002_I:MAIN_3.PNT_7	MAIN STEAM TEMPERATURE
4CP005_I:MAIN_4.PNT_7	FINAL SUPERHEAT INLET TEMPERATURE
4CP002_I:MAIN_2.PNT_8	HOT REHEAT TEMPERATURE
4O2CONTROL:O2_MEASURE.RO01	EXCESS O2 LH AVERAGE (B?)
4O2CONTROL:O2_MEASURE.RO02	EXCESS O2 RH AVERAGE (A?)
4AIRHEATER:APA_HEATER.RO01	PRIMARY HEATER A GAS OUTLET TEMPERATURE
4AIRHEATER:APA_HEATER.RO03	PRIMARY HEATER A AIR INLET TEMPERATURE
4AIRHEATER:BPA_HEATER.RO01	PRIMARY HEATER B GAS OUTLET TEMPERATURE
4AIRHEATER:BPA_HEATER.RO03	PRIMARY HEATER B AIR INLET TEMPERATURE
4AIRHEATER:ASEC_AIRHTR.RO01	SECONDARY AIR HEATER A GAS OUTLET TEMPERATURE
4AIRHEATER:ASEC_AIRHTR.RO03	SECONDARY AIR HEATER A AIR INLET TEMPERATURE
4AIRHEATER:BSEC_AIRHTR.RO01	SECONDARY AIR HEATER B GAS OUTLET TEMPERATURE
4AIRHEATER:BSEC_AIRHTR.RO03	SECONDARY AIR HEATER B AIR INLET TEMPERATURE
4CP003_I:MAIN_4.PNT_6	CARBON-IN-ASH (SEKAM)
4CP002_I:MAIN_10.PNT_3	MILL E COAL FLOW
4CP001_I:MAIN_10.PNT_1	COMPLIANCE CEM CO2
4CP001_I:MAIN_10.PNT_2	COMPLIANCE CEM NOX
4CP005_I:MAIN_12.PNT_4	RHDAMPER POSITION
4UPSHSPRAY:FSHOUT_CTRLR.SPT	TMS_Setpoint
4SHPASS:PASDMP_CTRLR.SPT	THRH_Setpoint
4UMSBOILER:THTPSP_RATEL.OUT	PMS_Setpoint
4CP001_I:MAIN_4.PNT_3	CONST.INI LOADTAG
4GNOCIS:LOGIC.BI04	CONST.INI CLOSEDLOOPTAG
4GNOCIS:LOGIC.BI06	CONST.INI REMOVEBIASTAG
4FUELCONTROL:FDRDMD_BAL.OUT	CONST.INI BIAS% FEEDERS
4OFAIRCDF1:CDF1STPT_SEL.OUT	CONST.INI BIASOUT YAOFAF1
4OFAIRCDR1:CDR1STPT_SEL.OUT	CONST.INI BIASOUT YAOFAR1
4OFAIRCDF2:CDF2STPT_SEL.OUT	CONST.INI BIASOUT YAOFAF2
4OFAIRCDR2:CDR2STPT_SEL.OUT	CONST.INI BIASOUT YAOFAR2
4O2CONTROL:GNOCIS_BIAS.MEAS	CONST.INI BIASOUT AVG_O2
4GNOCIS:ROUTS.PNT_2	CONST.INI GNOCISBIAS FDR A
4GNOCIS:ROUTS.PNT_3	CONST.INI GNOCISBIAS FDR B
4GNOCIS:ROUTS.PNT_4	CONST.INI GNOCISBIAS FDR C
4GNOCIS:ROUTS.PNT_5	CONST.INI GNOCISBIAS FDR D
4GNOCIS:ROUTS.PNT_6	CONST.INI GNOCISBIAS FDR E
4GNOCIS:ROUTS.PNT_7	CONST.INI GNOCISBIAS FDR F
4GNOCIS:ROUTS1.PNT_1	CONST.INI GNOCISBIAS YAOFAF1
4GNOCIS:ROUTS1.PNT_3	CONST.INI GNOCISBIAS YAOFAR1
4GNOCIS:ROUTS1.PNT_2	CONST.INI GNOCISBIAS YAOFAF2
4GNOCIS:ROUTS1.PNT_4	CONST.INI GNOCISBIAS YAOFAR2
4GNOCIS:ROUTS.PNT_1	CONST.INI GNOCISBIAS AVG_O2
4O2CONTROL:O2SETPT_BIAS.BIAS	CONST.INI OPERATORBIAS AVG_O2
4O2CONTROL:O2_CTRLR.MA	CONST.INI LOCALREMOTETAG AVG_O2
4FDRA:AFDR_STATION.MA	CONST.INI LOCALREMOTETAG FDR A

```
4FDRB:BFDR_STATION.MA          CONST.INI LOCALREMOTETAG FDR B
4FDRC:CFDR_STATION.MA          CONST.INI LOCALREMOTETAG FDR C
4FDRD:DFDR_STATION.MA          CONST.INI LOCALREMOTETAG FDR D
4FDRE:EFDR_STATION.MA          CONST.INI LOCALREMOTETAG FDR E
4FDRF:FFDR_STATION.MA          CONST.INI LOCALREMOTETAG FDR F
4OFAIRCDF1:OFACDF1_PCTL.MA     CONST.INI LOCALREMOTETAG YAOFAF1
4OFAIRCDR1:OFACDR1_PCTL.MA     CONST.INI LOCALREMOTETAG YAOFAR1
4OFAIRCDF2:OFACDF2_PCTL.MA     CONST.INI LOCALREMOTETAG YAOFAF2
4OFAIRCDR2:OFACDR2_PCTL.MA     CONST.INI LOCALREMOTETAG YAOFAR2
4GNOCIS:LOGIC.BO01             CONST.INI UPDATETAG
4GNOCIS:FDRATRIM.LOLIM         CONST.INI MINTAG FDR A
4GNOCIS:FDRATRIM.HOLIM         CONST.INI MAXTAG FDR A
4GNOCIS:FDRBTRIM.LOLIM         CONST.INI MINTAG FDR B
4GNOCIS:FDRBTRIM.HOLIM         CONST.INI MAXTAG FDR B
4GNOCIS:FDRCTRIM.LOLIM         CONST.INI MINTAG FDR C
4GNOCIS:FDRCTRIM.HOLIM         CONST.INI MAXTAG FDR C
4GNOCIS:FDRDTRIM.LOLIM         CONST.INI MINTAG FDR D
4GNOCIS:FDRDTRIM.HOLIM         CONST.INI MAXTAG FDR D
4GNOCIS:FDRETRIM.LOLIM         CONST.INI MINTAG FDR E
4GNOCIS:FDRETRIM.HOLIM         CONST.INI MAXTAG FDR E
4GNOCIS:FDRFTRIM.LOLIM         CONST.INI MINTAG FDR F
4GNOCIS:FDRFTRIM.HOLIM         CONST.INI MAXTAG FDR F
4GNOCIS:OFAF1TRIM.LOLIM        CONST.INI MINTAG YAOFAF1
4GNOCIS:OFAF1TRIM.HOLIM        CONST.INI MAXTAG YAOFAF1
4GNOCIS:OFAR1TRIM.LOLIM        CONST.INI MINTAG YAOFAR1
4GNOCIS:OFAR1TRIM.HOLIM        CONST.INI MAXTAG YAOFAR1
4GNOCIS:OFAF2TRIM.LOLIM        CONST.INI MINTAG YAOFAF2
4GNOCIS:OFAF2TRIM.HOLIM        CONST.INI MAXTAG YAOFAF2
4GNOCIS:OFAR2TRIM.LOLIM        CONST.INI MINTAG YAOFAR2
4GNOCIS:OFAR2TRIM.HOLIM        CONST.INI MAXTAG YAOFAR2
4GNOCIS:O2TRIM.LOLIM           CONST.INI MINTAG AVG_O2
4GNOCIS:O2TRIM.HOLIM           CONST.INI MAXTAG AVG_O2
4GNOCIS:NOXLIMS.LOLIM          CONST.INI MINTAG NOX_LBMMBTU
4GNOCIS:NOXLIMS.HOLIM          CONST.INI MAXTAG NOX_LBMMBTU
4GNOCIS:LOILIMTS.LOLIM         CONST.INI MINTAG CIA
4GNOCIS:LOILIMTS.HOLIM         CONST.INI MAXTAG CIA
4GNOCIS:EFFLIMTS.LOLIM         CONST.INI MINTAG EFF
4GNOCIS:EFFLIMTS.HOLIM         CONST.INI MAXTAG EFF
4GNOCIS:CLAMPS.CO_3            CONST.INI CLAMPTAG FDR A
4GNOCIS:CLAMPS.CO_4            CONST.INI CLAMPTAG FDR B
4GNOCIS:CLAMPS.CO_5            CONST.INI CLAMPTAG FDR C
4GNOCIS:CLAMPS.CO_6            CONST.INI CLAMPTAG FDR D
4GNOCIS:CLAMPS.CO_7            CONST.INI CLAMPTAG FDR E
4GNOCIS:CLAMPS.CO_8            CONST.INI CLAMPTAG FDR F
4GNOCIS:CLAMPS.CO_9            CONST.INI CLAMPTAG YAOFAF1
4GNOCIS:CLAMPS.CO_11           CONST.INI CLAMPTAG YAOFAR1
4GNOCIS:CLAMPS.CO_10           CONST.INI CLAMPTAG YAOFAF2
4GNOCIS:CLAMPS.CO_12           CONST.INI CLAMPTAG YAOFAR2
4GNOCIS:CLAMPS.CO_1            CONST.INI CLAMPTAG AVG_O2
```

GC_WMILLAC
GC_WMILLBC
GC_WMILLCC
GC_WMILLDC
GC_WMILLEC
GC_WMILLFC
GC_YAOFAF1
GC_YAOFAR1
GC_YAOFAF2
GC_YAOFAR2
GC_AVG_O2
GC_AVG_TSAAI
GC_AVG_TPAAI
GC_SCR_ON
GC_TMS_Setpoint
GC_THRH_Setpoint
GC_PMS_Setpoint
GC_AVG_TSAGO
GC_AVG_TPAGO
GC_AVG_DIV_WALL_INLET_T
GC_HOT_REHEAT_PRESS
GC_SUPERHEAT_INLET_TEMP
GCP_AVG_TSAGO
GCP_AVG_TPAGO
GCP_AVG_DIV_WALL_INLET_T
GCP_HOT_REHEAT_PRESS
GCP_SUPERHEAT_INLET_TEMP
GCP_NOX_LBMMBTU
GCP_CIA
GCP_THRH
GCP_TMS
GCP_PMS
GCP_SH_SPRAY_FLOW_UPPER
GCP_SH_SPRAY_FLOW_LOWER
GCP_EFF
GCP_RH_DAMPER_POS
GB_WMILLAC
GB_WMILLBC
GB_WMILLCC
GB_WMILLDC
GB_WMILLEC
GB_WMILLFC
GB_YAOFAF1
GB_YAOFAR1
GB_YAOFAF2
GB_YAOFAR2
GB_AVG_O2
GB_AVG_TSAAI
GB_AVG_TPAAI
GB_SCR_ON
GB_TMS_Setpoint
GB_THRH_Setpoint
GB_PMS_Setpoint
GB_AVG_TSAGO
GB_AVG_TPAGO
GB_AVG_DIV_WALL_INLET_T
GB_HOT_REHEAT_PRESS
GB_SUPERHEAT_INLET_TEMP
GBP_AVG_TSAGO

```
GBP_AVG_TPAGO
GBP_AVG_DIV_WALL_INLET_T
GBP_HOT_REHEAT_PRESS
GBP_SUPERHEAT_INLET_TEMP
GBP_NOX_LBMMBTU
GBP_CIA
GBP_THRH
GBP_TMS
GBP_PMS
GBP_SH_SPRAY_FLOW_UPPER
GBP_SH_SPRAY_FLOW_LOWER
GBP_EFF
GBP_RH_DAMPER_POS
GP_AVG_TSAGO
GP_AVG_TPAGO
GP_AVG_DIV_WALL_INLET_T
GP_HOT_REHEAT_PRESS
GP_SUPERHEAT_INLET_TEMP
GP_NOX_LBMMBTU
GP_CIA
GP_THRH
GP_TMS
GP_PMS
GP_SH_SPRAY_FLOW_UPPER
GP_SH_SPRAY_FLOW_LOWER
GP_EFF
GP_RH_DAMPER_POS
GPW_AVG_TSAGO
GPW_AVG_TPAGO
GPW_AVG_DIV_WALL_INLET_T
GPW_HOT_REHEAT_PRESS
GPW_SUPERHEAT_INLET_TEMP
GPW_NOX_LBMMBTU
GPW_CIA
GPW_THRH
GPW_TMS
GPW_PMS
GPW_SH_SPRAY_FLOW_UPPER
GPW_SH_SPRAY_FLOW_LOWER
GPW_EFF
GPW_RH_DAMPER_POS
GS_STATUS
GCO_NOX_LBMMBTU
GCO_CIA
GCO_THRH
GCO_TMS
GCO_PMS
GCO_SH_SPRAY_FLOW_UPPER
GCO_SH_SPRAY_FLOW_LOWER
GCO_EFF
GCO_RH_DAMPER_POS
GCV_WMILLAC
GCV_WMILLBC
GCV_WMILLCC
GCV_WMILLDC
GCV_WMILLEC
GCV_WMILLFC
GCV_YAOFAF1
GCV_YAOFAR1
```

GCV_YAOFAF2
GCV_YAOFAR2
GCV_AVG_O2

Dataset: C:\User\GlobalOpt\RTDS\rawdata\jan_aug_2002B
Model: C:\User\GlobalOpt\RTDS\rawdata\Ham_GO7B
Time Interval:
Filter used: None.

Model Variables:
----

index# (C Language)	control/independent_name	Time Delay
0	!WMILLAC!	0
1	!WMILLBC!	0
2	!WMILLCC!	0
3	!WMILLDC!	0
4	!WMILLEC!	0
5	!WMILLFC!	0
6	!YAOFAF1!	0
7	!YAOFAR1!	0
8	!YAOFAF2!	0
9	!YAOFAR2!	0
10	!AVG_O2!	0
11	!AVG_TSAAI!	0
12	!AVG_TPAAI!	0
13	!SCR_ON!	0
14	!TMS_Setpoint!	0
15	!THRH_Setpoint!	0
16	!PMS_Setpoint!	0

index# (C Language)	initial_state/dependent_name	Time Delay
17	!AVG_TSAGO!	0
18	!AVG_TPAGO!	0
19	!AVG_DIV_WALL_INLET_T!	0
20	!HOT_REHEAT_PRESS!	0
21	!SUPERHEAT_INLET_TEMP!	0

index# (C Language)	predicted_state/dependent_name	Time Delay
22	!AVG_TSAGO!	0
23	!AVG_TPAGO!	0
24	!AVG_DIV_WALL_INLET_T!	0
25	!HOT_REHEAT_PRESS!	0
26	!SUPERHEAT_INLET_TEMP!	0

index# (C Language)	output_name	Time Delay
27	!NOX_LBMMBTU!	0
28	!CIA!	0
29	!THRH!	0
30	!TMS!	0
31	!PMS!	0
32	!SH_SPRAY_FLOW_UPPER!	0
33	!SH_SPRAY_FLOW_LOWER!	0
34	!EFF!	0
35	!RH_DAMPER_POS!	0

Raw Tags:

Note: Tag ids are language independent.

TagId	name	comment	type
1	TIME	DATE AND TIME	datetime
2	4CP002_I:MAIN_10.PNT_1	MILL D COAL FLOW	float
3	4CP002_I:MAIN_11.PNT_1	MILL B COAL FLOW	float
4	4CP005_I:MAIN_5.PNT_1	DIVISION WALL INLET TEMPERATURE RH	float
5	4CP002_I:MAIN_10.PNT_2	MILL C COAL FLOW	float
6	4CP002_I:MAIN_11.PNT_2	MILL A COAL FLOW	float
7	4CP003_I:MAIN_6.PNT_2	OVERFIRE AIR DAMPER POSITION F1	float
8	4CP003_I:MAIN_7.PNT_2	OVERFIRE AIR DAMPER POSITION R1	float
9	4CP005_I:MAIN_4.PNT_2	DIVISION WALL INLET TEMPERATURE LH	float
10	4CP001_I:MAIN_1.PNT_3	MAIN STEAM PRESSURE	float
11	4CP002_I:MAIN_11.PNT_3	MILL F COAL FLOW	float
12	4CP005_I:MAIN_3.PNT_3	SUPERHEAT SPRAY FLOW (UPPER)	float
13	4CP005_I:MAIN_3.PNT_4	SUPERHEAT SPRAY FLOW (LOWER)	float
14	4CP003_I:MAIN_6.PNT_5	OVERFIRE AIR DAMPER POSITION F2	float
15	4CP003_I:MAIN_7.PNT_5	OVERFIRE AIR DAMPER POSITION R2	float
16	4CP001_I:MAIN_3.PNT_7	HOT REHEAT PRESSURE	float
17	4CP002_I:MAIN_3.PNT_7	MAIN STEAM TEMPERATURE	float
18	4CP005_I:MAIN_4.PNT_7	FINAL SUPERHEAT INLET TEMPERATURE	float
19	4CP002_I:MAIN_2.PNT_8	HOT REHEAT TEMPERATURE	float
20	4O2CONTROL:O2_MEASURE.RO01	EXCESS O2 LH AVERAGE (B?)	float
21	4O2CONTROL:O2_MEASURE.RO02	EXCESS O2 RH AVERAGE (A?)	float
22	4AIRHEATER:APA_HEATER.RO01	PRIMARY HEATER A GAS OUTLET TEMPERATURE	float
23	4AIRHEATER:APA_HEATER.RO03	PRIMARY HEATER A AIR INLET TEMPERATURE	float
24	4AIRHEATER:BPA_HEATER.RO01	PRIMARY HEATER B GAS OUTLET TEMPERATURE	float
25	4AIRHEATER:BPA_HEATER.RO03	PRIMARY HEATER B AIR INLET TEMPERATURE	float
26	4AIRHEATER:ASEC_AIRHTR.RO01	SECONDARY AIR HEATER A GAS OUTLET TEMPERATURE	float
27	4AIRHEATER:ASEC_AIRHTR.RO03	SECONDARY AIR HEATER A AIR INLET TEMPERATURE	float
28	4AIRHEATER:BSEC_AIRHTR.RO01	SECONDARY AIR HEATER B GAS OUTLET TEMPERATURE	float
29	4AIRHEATER:BSEC_AIRHTR.RO03	SECONDARY AIR HEATER B AIR INLET TEMPERATURE	float
30	4CP003_I:MAIN_4.PNT_6	CARBON-IN-ASH (SEKAM)	float
31	4CP002_I:MAIN_10.PNT_3	MILL E COAL FLOW	float
32	4CP001_I:MAIN_10.PNT_1	COMPLIANCE CEM CO2	float
33	4CP001_I:MAIN_10.PNT_2	COMPLIANCE CEM NOX	float
34	4CP005_I:MAIN_12.PNT_4	RHDAMPER POSITION	float
35	4UPSHSPRAY:FSHOUT_CTRLR.SPT	TMS SETPOINT	float
36	4SHPASS:PASDMP_CTRLR.SPT	THRH SETPOINT	float
37	4UMSBOILER:THTPSP_RATEL.OUT	PMS SETPOINT	float

Trans. Tags:

Note: Tag ids are language independent.

TagId	name	comment	type
1	TIME	DATE AND TIME	datetime
2	4CP002_I:MAIN_10.PNT_1	MILL D COAL FLOW	float
3	4CP002_I:MAIN_11.PNT_1	MILL B COAL FLOW	float
4	4CP005_I:MAIN_5.PNT_1	DIVISION WALL INLET TEMPERATURE RH	float
5	4CP002_I:MAIN_10.PNT_2	MILL C COAL FLOW	float
6	4CP002_I:MAIN_11.PNT_2	MILL A COAL FLOW	float
7	4CP003_I:MAIN_6.PNT_2	OVERFIRE AIR DAMPER POSITION F1	float
8	4CP003_I:MAIN_7.PNT_2	OVERFIRE AIR DAMPER POSITION R1	float
9	4CP005_I:MAIN_4.PNT_2	DIVISION WALL INLET TEMPERATURE LH	float
10	4CP001_I:MAIN_1.PNT_3	MAIN STEAM PRESSURE	float
11	4CP002_I:MAIN_11.PNT_3	MILL F COAL FLOW	float
12	4CP005_I:MAIN_3.PNT_3	SUPERHEAT SPRAY FLOW (UPPER)	float
13	4CP005_I:MAIN_3.PNT_4	SUPERHEAT SPRAY FLOW (LOWER)	float
14	4CP003_I:MAIN_6.PNT_5	OVERFIRE AIR DAMPER POSITION F2	float
15	4CP003_I:MAIN_7.PNT_5	OVERFIRE AIR DAMPER POSITION R2	float
16	4CP001_I:MAIN_3.PNT_7	HOT REHEAT PRESSURE	float
17	4CP002_I:MAIN_3.PNT_7	MAIN STEAM TEMPERATURE	float
18	4CP005_I:MAIN_4.PNT_7	FINAL SUPERHEAT INLET TEMPERATURE	float
19	4CP002_I:MAIN_2.PNT_8	HOT REHEAT TEMPERATURE	float
20	4O2CONTROL:O2_MEASURE.RO01	EXCESS O2 LH AVERAGE (B?)	float
21	4O2CONTROL:O2_MEASURE.RO02	EXCESS O2 RH AVERAGE (A?)	float
22	4AIRHEATER:APA_HEATER.RO01	PRIMARY HEATER A GAS OUTLET TEMPERATURE	float
23	4AIRHEATER:APA_HEATER.RO03	PRIMARY HEATER A AIR INLET TEMPERATURE	float
24	4AIRHEATER:BPA_HEATER.RO01	PRIMARY HEATER B GAS OUTLET TEMPERATURE	float
25	4AIRHEATER:BPA_HEATER.RO03	PRIMARY HEATER B AIR INLET TEMPERATURE	float
26	4AIRHEATER:ASEC_AIRHTR.RO01	SECONDARY AIR HEATER A GAS OUTLET TEMPERATURE	float
27	4AIRHEATER:ASEC_AIRHTR.RO03	SECONDARY AIR HEATER A AIR INLET TEMPERATURE	float
28	4AIRHEATER:BSEC_AIRHTR.RO01	SECONDARY AIR HEATER B GAS OUTLET TEMPERATURE	float
29	4AIRHEATER:BSEC_AIRHTR.RO03	SECONDARY AIR HEATER B AIR INLET TEMPERATURE	float
30	4CP003_I:MAIN_4.PNT_6	CARBON-IN-ASH (SEKAM)	float
31	4CP002_I:MAIN_10.PNT_3	MILL E COAL FLOW	float
32	4CP001_I:MAIN_10.PNT_1	COMPLIANCE CEM CO2	float
33	4CP001_I:MAIN_10.PNT_2	COMPLIANCE CEM NOX	float
34	4CP005_I:MAIN_12.PNT_4	RHDAMPER POSITION	float
35	4UPSHSPRAY:FSHOUT_CTRLR.SPT	TMS SETPOINT	float
36	4UMSBOILER:THTPSP_RATEL.OUT	PMS SETPOINT	float
37	4SHPASS:PASDMP_CTRLR.SPT	THRH SETPOINT	float
38	WMILLAC		float
39	WMILLBC		float
40	WMILLCC		float
41	WMILLDC		float
42	WMILLEC		float
43	WMILLFC		float
44	YAOFAF1		float
45	YAOFAR1		float
46	YAOFAF2		float
47	YAOFAR2		float
48	AVG_O2		float
49	NOX_LBMMBTU		float
50	AVG_TSAGO		float
51	AVG_TPAGO		float
52	AVG_TSAAI		float
53	AVG_TPAAI		float
54	EFF_SENS_LOSS		float
55	CIA		float
56	EFF_COMB_LOSS		float
57	THRH		float
58	TMS		float
59	PMS		float
60	RH_LOSS		float
61	SH_LOSS		float
62	SH_SPRAY_FLOW_UPPER		float
63	SH_SPRAY_FLOW_LOWER		float
64	SPRAY_LOSS		float
65	EFF		float
66	AVG_DIV_WALL_INLET_T		float
67	HOT_REHEAT_PRESS		float
68	SUPERHEAT_INLET_TEMP		float
69	SCR_ON		float
70	RH_DAMPER_POS		float
71	TMS_Setpoint		float
72	THRH_Setpoint		float
73	PMS_Setpoint		float

```
Model Settings:
--------------

Control/Independent Settings:
----------------------------

      index: 0 name: !WMILLAC! tau: 0 type: control/independent
clamp type: no clamp
min hard con: 10.312500
max hard con: 86949.800000
max inc: 91286.500000
max dec: 91286.500000
priority: 1.000000
convergence: 869.395313
desired: 53394.800000
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 10.312500
fuzzy max: 86949.800000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
```

```
confidence int: 90.000000

      index: 1 name: !WMILLBC! tau: 0 type: control/independent
clamp type: no clamp
min hard con: -13.750000
max hard con: 88334.600000
max inc: 92765.800000
max dec: 92765.800000
priority: 1.000000
convergence: 883.483359
desired: 34794.300000
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -13.750000
fuzzy max: 88334.600000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000

      index: 2 name: !WMILLCC! tau: 0 type: control/independent
clamp type: no clamp
min hard con: -8.593800
max hard con: 88158.700000
max inc: 92575.700000
max dec: 92575.700000
priority: 1.000000
convergence: 881.672891
desired: 55686.100000
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -8.593800
fuzzy max: 88158.700000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000

      index: 3 name: !WMILLDC! tau: 0 type: control/independent
clamp type: no clamp
min hard con: -1756.560000
max hard con: 88951.600000
max inc: 95243.600000
max dec: 95243.600000
priority: 1.000000
convergence: 907.081802
desired: 49043.400000
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -1756.560000
fuzzy max: 88951.600000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000

      index: 4 name: !WMILLEC! tau: 0 type: control/independent
clamp type: no clamp
min hard con: -3396.350000
max hard con: 79283.100000
max inc: 86813.400000
max dec: 86813.400000
priority: 1.000000
convergence: 826.794211
desired: 25586.300000
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -3396.350000
fuzzy max: 79283.100000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000

      index: 5 name: !WMILLFC! tau: 0 type: control/independent
clamp type: no clamp
min hard con: -6.875000
max hard con: 88513.900000
max inc: 92946.800000
max dec: 92946.800000
priority: 1.000000
convergence: 885.207813
desired: 53599.700000
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -6.875000
fuzzy max: 88513.900000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000
```

```
      index: 6 name: !YAOFAF1! tau: 0 type: control/independent
clamp type: no clamp
min hard con: 1.257800
max hard con: 100.154000
max inc: 103.841000
max dec: 103.841000
priority: 1.000000
convergence: 0.988965
desired: 32.413500
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 1.257800
fuzzy max: 100.154000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000

      index: 7 name: !YAOFAR1! tau: 0 type: control/independent
clamp type: no clamp
min hard con: 0.168000
max hard con: 100.445000
max inc: 105.291000
max dec: 105.291000
priority: 1.000000
convergence: 1.002773
desired: 32.126200
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 0.168000
fuzzy max: 100.445000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000

      index: 8 name: !YAOFAF2! tau: 0 type: control/independent
clamp type: no clamp
min hard con: -1.998000
max hard con: 101.998000
max inc: 109.196000
max dec: 109.196000
priority: 1.000000
convergence: 1.039960
desired: 32.290000
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -1.998000
fuzzy max: 101.998000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000

      index: 9 name: !YAOFAR2! tau: 0 type: control/independent
clamp type: no clamp
min hard con: 0.234400
max hard con: 101.248000
max inc: 106.064000
max dec: 106.064000
priority: 1.000000
convergence: 1.010136
desired: 32.261300
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 0.234400
fuzzy max: 101.248000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000

      index: 10 name: !AVG_O2! tau: 0 type: control/independent
clamp type: no clamp
min hard con: 2.075250
max hard con: 7.089250
max inc: 5.264700
max dec: 5.264700
priority: 1.000000
convergence: 0.050140
desired: 4.395350
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 2.075250
fuzzy max: 7.089250
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000
```

```
      index: 11 name: !AVG_TSAAI! tau: 0 type: control/independent
clamp type: compute
min hard con: 40.605200
max hard con: 117.405000
max inc: 80.639600
max dec: 80.639600
priority: 1.000000
convergence: 0.767996
desired: 88.695800
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 40.605200
fuzzy max: 117.405000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000

      index: 12 name: !AVG_TPAAI! tau: 0 type: control/independent
clamp type: compute
min hard con: 52.355900
max hard con: 139.665000
max inc: 91.674100
max dec: 91.674100
priority: 1.000000
convergence: 0.873087
desired: 110.046000
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 52.355900
fuzzy max: 139.665000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000

      index: 13 name: !SCR_ON! tau: 0 type: control/independent
clamp type: compute
min hard con: 0.000000
max hard con: 1.000000
max inc: 1.050000
max dec: 1.050000
priority: 1.000000
convergence: 0.010000
desired: 0.574849
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 0.000000
fuzzy max: 1.000000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000

      index: 14 name: !TMS_Setpoint! tau: 0 type: control/independent
clamp type: compute
min hard con: 950.000000
max hard con: 1050.000000
max inc: 105.000000
max dec: 105.000000
priority: 1.000000
convergence: 1.000000
desired: 999.997000
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 950.000000
fuzzy max: 1050.000000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000

      index: 15 name: !THRH_Setpoint! tau: 0 type: control/independent
clamp type: compute
min hard con: 950.000000
max hard con: 1050.000000
max inc: 105.000000
max dec: 105.000000
priority: 1.000000
convergence: 1.000000
desired: 999.993000
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 950.000000
fuzzy max: 1050.000000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000

      index: 16 name: !PMS_Setpoint! tau: 0 type: control/independent
```

```
clamp type: compute
min hard con: 2315.000000
max hard con: 2415.000000
max inc: 105.000000
max dec: 105.000000
priority: 1.000000
convergence: 1.000000
desired: 2349.480000
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 2315.000000
fuzzy max: 2415.000000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000


Initial State/Dependent Settings:
---------------------------------

      index: 17 name: !AVG_TSAGO! tau: 0 type: initial state/dependent
clamp type: no clamp
min hard con: 270.710297
max hard con: 362.459229
max inc: 1000000000000000000000000000000.000000
max dec: 1000000000000000000000000000000.000000
priority: 1.000000
convergence: 0.917489
desired: 318.913767
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -1000000000000000000000000000000.000000
fuzzy max: 1000000000000000000000000000000.000000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 18 name: !AVG_TPAGO! tau: 0 type: initial state/dependent
clamp type: no clamp
min hard con: 183.363190
max hard con: 297.233795
max inc: 1000000000000000000000000000000.000000
max dec: 1000000000000000000000000000000.000000
priority: 1.000000
convergence: 1.138706
desired: 223.105481
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -1000000000000000000000000000000.000000
fuzzy max: 1000000000000000000000000000000.000000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 19 name: !AVG_DIV_WALL_INLET_T! tau: 0 type: initial state/dependent
clamp type: no clamp
min hard con: 667.757996
max hard con: 701.786743
max inc: 1000000000000000000000000000000.000000
max dec: 1000000000000000000000000000000.000000
priority: 1.000000
convergence: 0.340287
desired: 686.268309
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -1000000000000000000000000000000.000000
fuzzy max: 1000000000000000000000000000000.000000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 20 name: !HOT_REHEAT_PRESS! tau: 0 type: initial state/dependent
clamp type: no clamp
min hard con: 215.842407
max hard con: 626.294922
max inc: 1000000000000000000000000000000.000000
max dec: 1000000000000000000000000000000.000000
priority: 1.000000
convergence: 4.104525
desired: 414.759241
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -1000000000000000000000000000000.000000
fuzzy max: 1000000000000000000000000000000.000000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
```

```
confidence int: 90.000000

      index: 21 name: !SUPERHEAT_INLET_TEMP! tau: 0 type: initial state/dependent
clamp type: no clamp
min hard con: 813.939087
max hard con: 1012.822998
max inc: 1000000000000000000000000000000.000000
max dec: 1000000000000000000000000000000.000000
priority: 1.000000
convergence: 1.988839
desired: 879.076524
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -1000000000000000000000000000000.000000
fuzzy max: 1000000000000000000000000000000.000000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000


Predicted State/Dependent Settings:
-----------------------------------

      index: 22 name: !AVG_TSAGO! tau: 0 type: predicted state/dependent
clamp type: no clamp
max inc: 96.336378
max dec: 96.336378
priority: 0.000000
convergence: 0.917489
desired: 318.851551
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 270.710297
fuzzy max: 362.459229
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 23 name: !AVG_TPAGO! tau: 0 type: predicted state/dependent
clamp type: no clamp
max inc: 119.564136
max dec: 119.564136
priority: 0.000000
convergence: 1.138706
desired: 223.148481
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 183.363190
fuzzy max: 297.233795
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 24 name: !AVG_DIV_WALL_INLET_T! tau: 0 type: predicted state/dependent
clamp type: no clamp
max inc: 35.730185
max dec: 35.730185
priority: 0.000000
convergence: 0.340287
desired: 686.275855
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 667.757996
fuzzy max: 701.786743
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 25 name: !HOT_REHEAT_PRESS! tau: 0 type: predicted state/dependent
clamp type: no clamp
max inc: 430.975140
max dec: 430.975140
priority: 0.000000
convergence: 4.104525
desired: 414.599253
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 215.842407
fuzzy max: 626.294922
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 26 name: !SUPERHEAT_INLET_TEMP! tau: 0 type: predicted state/dependent
clamp type: no clamp
max inc: 208.828107
```

```
max dec: 208.828107
priority: 0.000000
convergence: 1.988839
desired: 878.986270
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 813.939087
fuzzy max: 1012.822998
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000


Output Settings:
---------------

      index: 27 name: !NOX_LBMMBTU! tau: 0 type: output
clamp type: no clamp
max inc: 0.821819
max dec: 0.821819
priority: 0.000000
convergence: 0.007827
desired: 0.208979
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 0.012058
fuzzy max: 0.794743
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 28 name: !CIA! tau: 0 type: output
clamp type: no clamp
max inc: 23.722657
max dec: 23.722657
priority: 0.000000
convergence: 0.225930
desired: 8.013780
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 0.114633
fuzzy max: 22.707640
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 29 name: !THRH! tau: 0 type: output
clamp type: no clamp
max inc: 111.874796
max dec: 111.874796
priority: 0.000000
convergence: 1.065474
desired: 989.509000
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 927.359924
fuzzy max: 1033.907349
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 30 name: !TMS! tau: 0 type: output
clamp type: no clamp
max inc: 111.718295
max dec: 111.718295
priority: 0.000000
convergence: 1.063984
desired: 998.375000
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 939.301575
fuzzy max: 1045.699951
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 31 name: !PMS! tau: 0 type: output
clamp type: no clamp
max inc: 353.165039
max dec: 353.165039
priority: 0.000000
convergence: 3.363477
desired: 2362.920000
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
```

```
cost: 0.000000
optimization method: none
fuzzy min: 2218.164063
fuzzy max: 2554.511719
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 32 name: !SH_SPRAY_FLOW_UPPER! tau: 0 type: output
clamp type: no clamp
max inc: 161660.034439
max dec: 161660.034439
priority: 0.000000
convergence: 1539.619376
desired: 90900.900000
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -964.843811
fuzzy max: 152997.093750
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 33 name: !SH_SPRAY_FLOW_LOWER! tau: 0 type: output
clamp type: no clamp
max inc: 145287.573111
max dec: 145287.573111
priority: 0.000000
convergence: 1383.691172
desired: 68304.600000
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -969.726624
fuzzy max: 137399.390625
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 34 name: !EFF! tau: 0 type: output
clamp type: no clamp
max inc: 3.876289
max dec: 3.876289
priority: 0.000000
convergence: 0.036917
desired: 86.087100
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 83.816849
fuzzy max: 87.508553
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 35 name: !RH_DAMPER_POS! tau: 0 type: output
clamp type: no clamp
max inc: 105.664454
max dec: 105.664454
priority: 0.000000
convergence: 1.006328
desired: 85.537700
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -0.818360
fuzzy max: 99.814453
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000



Global Optimizer Parameters:
----------------------------
autoscale off
timeout   0.000000

Combined Constraints:
---------------------------

Constraint No: 0 Type: Constraint (On)
   -0.005000 <= ((!WMILLAC[0]! + !WMILLBC[0]! + !WMILLCC[0]! + !WMILLDC[0]! + !WMILLEC[0]! + !WMILLFC[0]!)  - (!WMILLAC
[0]:original! + !WMILLBC[0]:original! + !WMILLCC[0]:original! + !WMILLDC[0]:original! + !WMILLEC[0]:original! + !WMILLFC
[0]:original!) )  / (!WMILLAC[0]:original! + !WMILLBC[0]:original! + !WMILLCC[0]:original! + !WMILLDC[0]:original! + !WMILLEC
[0]:original! + !WMILLFC[0]:original!)  <= 0.005000


Applied Transforms.
-------------------
```

```
transform { name "!WMILLAC!" expr "!4CP002_I:MAIN_11.PNT_2!" }
transform { name "!WMILLBC!" expr "!4CP002_I:MAIN_11.PNT_1!" }
transform { name "!WMILLCC!" expr "!4CP002_I:MAIN_10.PNT_2!" }
transform { name "!WMILLDC!" expr "!4CP002_I:MAIN_10.PNT_1!" }
transform { name "!WMILLEC!" expr "!4CP002_I:MAIN_10.PNT_3!" }
transform { name "!WMILLFC!" expr "!4CP002_I:MAIN_11.PNT_3!" }
transform { name "!YAOFAF1!" expr "!4CP003_I:MAIN_6.PNT_2!" }
transform { name "!YAOFAR1!" expr "!4CP003_I:MAIN_7.PNT_2!" }
transform { name "!YAOFAF2!" expr "!4CP003_I:MAIN_6.PNT_5!" }
transform { name "!YAOFAR2!" expr "!4CP003_I:MAIN_7.PNT_5!" }
transform { name "!AVG_O2!" expr "(!4O2CONTROL:O2_MEASURE.RO01! + !4O2CONTROL:O2_MEASURE.RO02!)  / 2.0" }
transform { name "!NOX_LBMMBTU!" expr "!4CP001_I:MAIN_10.PNT_2! * 100. * 1800. * 1.194E-7 / !4CP001_I:MAIN_10.PNT_1!" }
transform { name "!AVG_TSAGO!" expr "(!4AIRHEATER:ASEC_AIRHTR.RO01! + !4AIRHEATER:BSEC_AIRHTR.RO01!)  / 2.0" }
transform { name "!AVG_TPAGO!" expr "(!4AIRHEATER:APA_HEATER.RO01! + !4AIRHEATER:BPA_HEATER.RO01!)  / 2.0" }
transform { name "!AVG_TSAAI!" expr "(!4AIRHEATER:ASEC_AIRHTR.RO03! + !4AIRHEATER:BSEC_AIRHTR.RO03!)  / 2.0" }
transform { name "!AVG_TPAAI!" expr "(!4AIRHEATER:APA_HEATER.RO03! + !4AIRHEATER:BPA_HEATER.RO03!)  / 2.0" }
transform { name "!EFF_SENS_LOSS!" expr "(0.8 * (!AVG_TSAGO! - !AVG_TSAAI!)  + 0.2 * (!AVG_TPAGO! - !AVG_TPAAI!) )  * (.023 +
.00011 * (1. + !AVG_O2! / 0.9) ^2) " }
transform { name "!CIA!" expr "!4CP003_I:MAIN_4.PNT_6!" }
transform { name "!EFF_COMB_LOSS!" expr "0.1 * !CIA! * 14100 / 12000" }
transform { name "!THRH!" expr "!4CP002_I:MAIN_2.PNT_8!" }
transform { name "!TMS!" expr "!4CP002_I:MAIN_3.PNT_7!" }
transform { name "!PMS!" expr "!4CP001_I:MAIN_1.PNT_3!" }
transform { name "!RH_LOSS!" expr " $if(!THRH! < 1000., (1000. - !THRH!)  / 10 * .1, 0.) " }
transform { name "!SH_LOSS!" expr " $if(!TMS! < 1000., (1000. - !TMS!)  / 10 * .2, 0.) " }
transform { name "!SH_SPRAY_FLOW_UPPER!" expr "!4CP005_I:MAIN_3.PNT_3!" }
transform { name "!SH_SPRAY_FLOW_LOWER!" expr "!4CP005_I:MAIN_3.PNT_4!" }
transform { name "!SPRAY_LOSS!" expr "0.0048 * (!SH_SPRAY_FLOW_UPPER! + !SH_SPRAY_FLOW_LOWER!)  / 1000.0" }
transform { name "!EFF!" expr "100. - 6.5 - !EFF_SENS_LOSS! - !EFF_COMB_LOSS! - !RH_LOSS! - !SH_LOSS! - !SPRAY_LOSS!" }
transform { name "!AVG_DIV_WALL_INLET_T!" expr "(!4CP005_I:MAIN_5.PNT_1! + !4CP005_I:MAIN_4.PNT_2!)  / 2.0" }
transform { name "!HOT_REHEAT_PRESS!" expr "!4CP001_I:MAIN_3.PNT_7!" }
transform { name "!SUPERHEAT_INLET_TEMP!" expr "!4CP005_I:MAIN_4.PNT_7!" }
transform { name "!SCR_ON!" expr " $if(!NOX_LBMMBTU! < 0.2, 1., 0.) " }
transform { name "!RH_DAMPER_POS!" expr "!4CP005_I:MAIN_12.PNT_4!" }
transform { name "!TMS_Setpoint!" expr "!4UPSHSPRAY:FSHOUT_CTRLR.SPT!" }
transform { name "!THRH_Setpoint!" expr "!4SHPASS:PASDMP_CTRLR.SPT!" }
transform { name "!PMS_Setpoint!" expr "!4UMSBOILER:THTPSP_RATEL.OUT!" }
```

```
SVModelName = c:\gnocis\activemodel\ham4_sv_GO7
InPort = 2001
DataAddress = 127.0.0.1
DataPort = 2003
ArchAddress = 127.0.0.1
ArchPort = 2004
FlagAddress = 127.0.0.1
FlagPort = 2005
OutAddress = 127.0.0.1
OutPort = 2006
TagFile = c:\gnocis\activemodel\u4_tag.ini
OnOffTag = 4GNOCIS:LOGIC.BI04
Debug = 5
```

```
Port = 2004
TagFileName = c:\gnocis\activemodel\U4_Svtag.ini
InOut = in
Debug = 0
```

```
Port = 2005
TagFileName = c:\gnocis\activemodel\U4_Svouts.ini
InOut = in
Debug = 0
```

4CP002_I:MAIN_10.PNT_1	MILL D COAL FLOW
4CP002_I:MAIN_11.PNT_1	MILL B COAL FLOW
4CP005_I:MAIN_5.PNT_1	DIVISION WALL INLET TEMPERATURE RH
4CP002_I:MAIN_10.PNT_2	MILL C COAL FLOW
4CP002_I:MAIN_11.PNT_2	MILL A COAL FLOW
4CP003_I:MAIN_6.PNT_2	OVERFIRE AIR DAMPER POSITION F1
4CP003_I:MAIN_7.PNT_2	OVERFIRE AIR DAMPER POSITION R1
4CP005_I:MAIN_4.PNT_2	DIVISION WALL INLET TEMPERATURE LH
4CP001_I:MAIN_1.PNT_3	MAIN STEAM PRESSURE
4CP002_I:MAIN_11.PNT_3	MILL F COAL FLOW
4CP005_I:MAIN_3.PNT_3	SUPERHEAT SPRAY FLOW (UPPER)
4CP005_I:MAIN_3.PNT_4	SUPERHEAT SPRAY FLOW (LOWER)
4CP003_I:MAIN_6.PNT_5	OVERFIRE AIR DAMPER POSITION F2
4CP003_I:MAIN_7.PNT_5	OVERFIRE AIR DAMPER POSITION R2
4CP001_I:MAIN_3.PNT_7	HOT REHEAT PRESSURE
4CP002_I:MAIN_3.PNT_7	MAIN STEAM TEMPERATURE
4CP005_I:MAIN_4.PNT_7	FINAL SUPERHEAT INLET TEMPERATURE
4CP002_I:MAIN_2.PNT_8	HOT REHEAT TEMPERATURE
4O2CONTROL:O2_MEASURE.RO01	EXCESS O2 LH AVERAGE (B?)
4O2CONTROL:O2_MEASURE.RO02	EXCESS O2 RH AVERAGE (A?)
4AIRHEATER:APA_HEATER.RO01	PRIMARY HEATER A GAS OUTLET TEMPERATURE
4AIRHEATER:APA_HEATER.RO03	PRIMARY HEATER A AIR INLET TEMPERATURE
4AIRHEATER:BPA_HEATER.RO01	PRIMARY HEATER B GAS OUTLET TEMPERATURE
4AIRHEATER:BPA_HEATER.RO03	PRIMARY HEATER B AIR INLET TEMPERATURE
4AIRHEATER:ASEC_AIRHTR.RO01	SECONDARY AIR HEATER A GAS OUTLET TEMPERATURE
4AIRHEATER:ASEC_AIRHTR.RO03	SECONDARY AIR HEATER A AIR INLET TEMPERATURE
4AIRHEATER:BSEC_AIRHTR.RO01	SECONDARY AIR HEATER B GAS OUTLET TEMPERATURE
4AIRHEATER:BSEC_AIRHTR.RO03	SECONDARY AIR HEATER B AIR INLET TEMPERATURE
4CP003_I:MAIN_4.PNT_6	CARBON-IN-ASH (SEKAM)
4CP002_I:MAIN_10.PNT_3	MILL E COAL FLOW
4CP001_I:MAIN_10.PNT_1	COMPLIANCE CEM CO2
4CP001_I:MAIN_10.PNT_2	COMPLIANCE CEM NOX
4CP005_I:MAIN_12.PNT_4	RHDAMPER POSITION
4UPSHSPRAY:FSHOUT_CTRLR.SPT	TMS_Setpoint
4SHPASS:PASDMP_CTRLR.SPT	THRH_Setpoint
4UMSBOILER:THTPSP_RATEL.OUT	PMS_Setpoint
SENSORSUB	SENSOR SUBSTITUTED

```
4CP002_I:MAIN_10.PNT_1          MILL D COAL FLOW
4CP002_I:MAIN_11.PNT_1          MILL B COAL FLOW
4CP005_I:MAIN_5.PNT_1           DIVISION WALL INLET TEMPERATURE RH
4CP002_I:MAIN_10.PNT_2          MILL C COAL FLOW
4CP002_I:MAIN_11.PNT_2          MILL A COAL FLOW
4CP003_I:MAIN_6.PNT_2           OVERFIRE AIR DAMPER POSITION F1
4CP003_I:MAIN_7.PNT_2           OVERFIRE AIR DAMPER POSITION R1
4CP005_I:MAIN_4.PNT_2           DIVISION WALL INLET TEMPERATURE LH
4CP001_I:MAIN_1.PNT_3           MAIN STEAM PRESSURE
4CP002_I:MAIN_11.PNT_3          MILL F COAL FLOW
4CP005_I:MAIN_3.PNT_3           SUPERHEAT SPRAY FLOW (UPPER)
4CP005_I:MAIN_3.PNT_4           SUPERHEAT SPRAY FLOW (LOWER)
4CP003_I:MAIN_6.PNT_5           OVERFIRE AIR DAMPER POSITION F2
4CP003_I:MAIN_7.PNT_5           OVERFIRE AIR DAMPER POSITION R2
4CP001_I:MAIN_3.PNT_7           HOT REHEAT PRESSURE
4CP002_I:MAIN_3.PNT_7           MAIN STEAM TEMPERATURE
4CP005_I:MAIN_4.PNT_7           FINAL SUPERHEAT INLET TEMPERATURE
4CP002_I:MAIN_2.PNT_8           HOT REHEAT TEMPERATURE
4O2CONTROL:O2_MEASURE.RO01      EXCESS O2 LH AVERAGE (B?)
4O2CONTROL:O2_MEASURE.RO02      EXCESS O2 RH AVERAGE (A?)
4AIRHEATER:APA_HEATER.RO01      PRIMARY HEATER A GAS OUTLET TEMPERATURE
4AIRHEATER:APA_HEATER.RO03      PRIMARY HEATER A AIR INLET TEMPERATURE
4AIRHEATER:BPA_HEATER.RO01      PRIMARY HEATER B GAS OUTLET TEMPERATURE
4AIRHEATER:BPA_HEATER.RO03      PRIMARY HEATER B AIR INLET TEMPERATURE
4AIRHEATER:ASEC_AIRHTR.RO01     SECONDARY AIR HEATER A GAS OUTLET TEMPERATURE
4AIRHEATER:ASEC_AIRHTR.RO03     SECONDARY AIR HEATER A AIR INLET TEMPERATURE
4AIRHEATER:BSEC_AIRHTR.RO01     SECONDARY AIR HEATER B GAS OUTLET TEMPERATURE
4AIRHEATER:BSEC_AIRHTR.RO03     SECONDARY AIR HEATER B AIR INLET TEMPERATURE
4CP003_I:MAIN_4.PNT_6           CARBON-IN-ASH (SEKAM)
4CP002_I:MAIN_10.PNT_3          MILL E COAL FLOW
4CP001_I:MAIN_10.PNT_1          COMPLIANCE CEM CO2
4CP001_I:MAIN_10.PNT_2          COMPLIANCE CEM NOX
4CP005_I:MAIN_12.PNT_4          RHDAMPER POSITION
4UPSHSPRAY:FSHOUT_CTRLR.SPT     TMS_Setpoint
4SHPASS:PASDMP_CTRLR.SPT        THRH_Setpoint
4UMSBOILER:THTPSP_RATEL.OUT     PMS_Setpoint
```

Dataset: C:\User\GlobalOpt\GNOCIS\rawdata\jan_aug2002
Model: C:\User\GlobalOpt\RTDS\Ham4_sv_GO
Time Interval:
Filter used: None.

Model Variables:
----

index# (C Language)	input_name	Time Delay
0	!4CP002_I:MAIN_10.PNT_1!	0
1	!4CP002_I:MAIN_11.PNT_1!	0
2	!4CP005_I:MAIN_5.PNT_1!	0
3	!4CP002_I:MAIN_10.PNT_2!	0
4	!4CP002_I:MAIN_11.PNT_2!	0
5	!4CP003_I:MAIN_6.PNT_2!	0
6	!4CP003_I:MAIN_7.PNT_2!	0
7	!4CP005_I:MAIN_4.PNT_2!	0
8	!4CP001_I:MAIN_1.PNT_3!	0
9	!4CP002_I:MAIN_11.PNT_3!	0
10	!4CP005_I:MAIN_3.PNT_3!	0
11	!4CP005_I:MAIN_3.PNT_4!	0
12	!4CP003_I:MAIN_6.PNT_5!	0
13	!4CP003_I:MAIN_7.PNT_5!	0
14	!4CP001_I:MAIN_3.PNT_7!	0
15	!4CP002_I:MAIN_3.PNT_7!	0
16	!4CP005_I:MAIN_4.PNT_7!	0
17	!4CP002_I:MAIN_2.PNT_8!	0
18	!4O2CONTROL:O2_MEASURE.RO01!	0
19	!4O2CONTROL:O2_MEASURE.RO02!	0
20	!4AIRHEATER:APA_HEATER.RO01!	0
21	!4AIRHEATER:APA_HEATER.RO03!	0
22	!4AIRHEATER:BPA_HEATER.RO01!	0
23	!4AIRHEATER:BPA_HEATER.RO03!	0
24	!4AIRHEATER:ASEC_AIRHTR.RO01!	0
25	!4AIRHEATER:ASEC_AIRHTR.RO03!	0
26	!4AIRHEATER:BSEC_AIRHTR.RO01!	0
27	!4AIRHEATER:BSEC_AIRHTR.RO03!	0
28	!4CP003_I:MAIN_4.PNT_6!	0
29	!4CP002_I:MAIN_10.PNT_3!	0
30	!4CP001_I:MAIN_10.PNT_1!	0
31	!4CP001_I:MAIN_10.PNT_2!	0
32	!4CP005_I:MAIN_12.PNT_4!	0
33	!TMS_Setpoint!	0
34	!THRH_Setpoint!	0
35	!PMS_Setpoint!	0

index# (C Language)	output_name	Time Delay
36	!4CP002_I:MAIN_10.PNT_1!	0
37	!4CP002_I:MAIN_11.PNT_1!	0
38	!4CP005_I:MAIN_5.PNT_1!	0
39	!4CP002_I:MAIN_10.PNT_2!	0
40	!4CP002_I:MAIN_11.PNT_2!	0
41	!4CP003_I:MAIN_6.PNT_2!	0
42	!4CP003_I:MAIN_7.PNT_2!	0
43	!4CP005_I:MAIN_4.PNT_2!	0
44	!4CP001_I:MAIN_1.PNT_3!	0
45	!4CP002_I:MAIN_11.PNT_3!	0
46	!4CP005_I:MAIN_3.PNT_3!	0
47	!4CP005_I:MAIN_3.PNT_4!	0
48	!4CP003_I:MAIN_6.PNT_5!	0
49	!4CP003_I:MAIN_7.PNT_5!	0
50	!4CP001_I:MAIN_3.PNT_7!	0
51	!4CP002_I:MAIN_3.PNT_7!	0
52	!4CP005_I:MAIN_4.PNT_7!	0
53	!4CP002_I:MAIN_2.PNT_8!	0
54	!4O2CONTROL:O2_MEASURE.RO01!	0
55	!4O2CONTROL:O2_MEASURE.RO02!	0
56	!4AIRHEATER:APA_HEATER.RO01!	0
57	!4AIRHEATER:APA_HEATER.RO03!	0
58	!4AIRHEATER:BPA_HEATER.RO01!	0
59	!4AIRHEATER:BPA_HEATER.RO03!	0
60	!4AIRHEATER:ASEC_AIRHTR.RO01!	0
61	!4AIRHEATER:ASEC_AIRHTR.RO03!	0
62	!4AIRHEATER:BSEC_AIRHTR.RO01!	0
63	!4AIRHEATER:BSEC_AIRHTR.RO03!	0
64	!4CP003_I:MAIN_4.PNT_6!	0
65	!4CP002_I:MAIN_10.PNT_3!	0
66	!4CP001_I:MAIN_10.PNT_1!	0
67	!4CP001_I:MAIN_10.PNT_2!	0
68	!4CP005_I:MAIN_12.PNT_4!	0
69	!TMS_Setpoint!	0
70	!THRH_Setpoint!	0
71	!PMS_Setpoint!	0

Raw Tags:

Note: Tag ids are language independent.

TagId	name	comment	type
1	merged_time		datetime
2	4CP002_I:MAIN_10.PNT_1		float
3	4CP002_I:MAIN_11.PNT_1		float
4	4CP005_I:MAIN_5.PNT_1		float
5	4CP002_I:MAIN_10.PNT_2		float
6	4CP002_I:MAIN_11.PNT_2		float
7	4CP003_I:MAIN_6.PNT_2		float
8	4CP003_I:MAIN_7.PNT_2		float

9	4CP005_I:MAIN_4.PNT_2		float
10	4CP001_I:MAIN_1.PNT_3		float
11	4CP002_I:MAIN_11.PNT_3		float
12	4CP005_I:MAIN_3.PNT_3		float
13	4CP005_I:MAIN_3.PNT_4		float
14	4CP003_I:MAIN_6.PNT_5		float
15	4CP003_I:MAIN_7.PNT_5		float
16	4CP001_I:MAIN_3.PNT_7		float
17	4CP002_I:MAIN_3.PNT_7		float
18	4CP005_I:MAIN_4.PNT_7		float
19	4CP002_I:MAIN_2.PNT_8		float
20	4O2CONTROL:O2_MEASURE.RO01		float
21	4O2CONTROL:O2_MEASURE.RO02		float
22	4AIRHEATER:APA_HEATER.RO01		float
23	4AIRHEATER:APA_HEATER.RO03		float
24	4AIRHEATER:BPA_HEATER.RO01		float
25	4AIRHEATER:BPA_HEATER.RO03		float
26	4AIRHEATER:ASEC_AIRHTR.RO01		float
27	4AIRHEATER:ASEC_AIRHTR.RO03		float
28	4AIRHEATER:BSEC_AIRHTR.RO01		float
29	4AIRHEATER:BSEC_AIRHTR.RO03		float
30	4CP003_I:MAIN_4.PNT_6		float
31	4CP002_I:MAIN_10.PNT_3		float
32	4CP001_I:MAIN_10.PNT_1		float
33	4CP001_I:MAIN_10.PNT_2		float
34	4CP005_I:MAIN_12.PNT_4	RHDAMPER POSITION	float
35	TMS_Setpoint		float
36	THRH_Setpoint		float
37	PMS_Setpoint		float

Trans. Tags:

Note: Tag ids are language independent.

TagId	name	comment	type
1	merged_time		datetime
2	4CP002_I:MAIN_10.PNT_1		float
3	4CP002_I:MAIN_11.PNT_1		float
4	4CP005_I:MAIN_5.PNT_1		float
5	4CP002_I:MAIN_10.PNT_2		float
6	4CP002_I:MAIN_11.PNT_2		float
7	4CP003_I:MAIN_6.PNT_2		float
8	4CP003_I:MAIN_7.PNT_2		float
9	4CP005_I:MAIN_4.PNT_2		float
10	4CP001_I:MAIN_1.PNT_3		float
11	4CP002_I:MAIN_11.PNT_3		float
12	4CP005_I:MAIN_3.PNT_3		float
13	4CP005_I:MAIN_3.PNT_4		float
14	4CP003_I:MAIN_6.PNT_5		float
15	4CP003_I:MAIN_7.PNT_5		float
16	4CP001_I:MAIN_3.PNT_7		float
17	4CP002_I:MAIN_3.PNT_7		float
18	4CP005_I:MAIN_4.PNT_7		float
19	4CP002_I:MAIN_2.PNT_8		float
20	4O2CONTROL:O2_MEASURE.RO01		float
21	4O2CONTROL:O2_MEASURE.RO02		float
22	4AIRHEATER:APA_HEATER.RO01		float
23	4AIRHEATER:APA_HEATER.RO03		float
24	4AIRHEATER:BPA_HEATER.RO01		float
25	4AIRHEATER:BPA_HEATER.RO03		float
26	4AIRHEATER:ASEC_AIRHTR.RO01		float
27	4AIRHEATER:ASEC_AIRHTR.RO03		float
28	4AIRHEATER:BSEC_AIRHTR.RO01		float
29	4AIRHEATER:BSEC_AIRHTR.RO03		float
30	4CP003_I:MAIN_4.PNT_6		float
31	4CP002_I:MAIN_10.PNT_3		float
32	4CP001_I:MAIN_10.PNT_1		float
33	4CP001_I:MAIN_10.PNT_2		float
34	4CP005_I:MAIN_12.PNT_4	RHDAMPER POSITION	float
35	TMS_Setpoint		float
36	THRH_Setpoint		float
37	PMS_Setpoint		float

Model Settings:

Input Settings:
----

index: 0 name: !4CP002_I:MAIN_10.PNT_1! tau: 0 type: input
clamp type: no clamp
min hard con: -1756.562988
max hard con: 88951.617188
max inc: 95243.589185
max dec: 95243.589185
priority: 1.000000
convergence: 907.081802
desired: 52364.212299
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -1756.562988
fuzzy max: 88951.617188
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 1 name: !4CP002_I:MAIN_11.PNT_1! tau: 0 type: input
clamp type: no clamp
min hard con: -13.750000
max hard con: 88334.585938
max inc: 92765.752734
max dec: 92765.752734
priority: 1.000000
convergence: 883.483359
desired: 42454.790361
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -13.750000
fuzzy max: 88334.585938
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 2 name: !4CP005_I:MAIN_5.PNT_1! tau: 0 type: input
clamp type: no clamp
min hard con: 657.098328
max hard con: 703.980225
max inc: 49.225992
max dec: 49.225992
priority: 1.000000
convergence: 0.468819
desired: 687.868811
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 657.098328
fuzzy max: 703.980225
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 3 name: !4CP002_I:MAIN_10.PNT_2! tau: 0 type: input
clamp type: no clamp
min hard con: -8.593800
max hard con: 88158.695313
max inc: 92575.653568
max dec: 92575.653568
priority: 1.000000
convergence: 881.672891
desired: 57514.137485
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -8.593800
fuzzy max: 88158.695313
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 4 name: !4CP002_I:MAIN_11.PNT_2! tau: 0 type: input
clamp type: no clamp
min hard con: 10.312500
max hard con: 86949.843750
max inc: 91286.507813
max dec: 91286.507813
priority: 1.000000
convergence: 869.395313
desired: 55484.448221
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 10.312500
fuzzy max: 86949.843750
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 5 name: !4CP003_I:MAIN_6.PNT_2! tau: 0 type: input
clamp type: no clamp
min hard con: 1.257800
max hard con: 100.154297
max inc: 103.841322
max dec: 103.841322
priority: 1.000000
convergence: 0.988965
desired: 41.314078
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 1.257800
fuzzy max: 100.154297
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

```
      index: 6 name: !4CP003_I:MAIN_7.PNT_2! tau: 0 type: input
clamp type: no clamp
min hard con: 0.168000
max hard con: 100.445297
max inc: 105.291162
max dec: 105.291162
priority: 1.000000
convergence: 1.002773
desired: 41.050029
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 0.168000
fuzzy max: 100.445297
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 7 name: !4CP005_I:MAIN_4.PNT_2! tau: 0 type: input
clamp type: no clamp
min hard con: 667.721985
max hard con: 702.524719
max inc: 36.542871
max dec: 36.542871
priority: 1.000000
convergence: 0.348027
desired: 687.867800
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 667.721985
fuzzy max: 702.524719
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 8 name: !4CP001_I:MAIN_1.PNT_3! tau: 0 type: input
clamp type: no clamp
min hard con: 2218.164063
max hard con: 2703.378906
max inc: 509.475586
max dec: 509.475586
priority: 1.000000
convergence: 4.852148
desired: 2364.221678
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 2218.164063
fuzzy max: 2703.378906
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 9 name: !4CP002_I:MAIN_11.PNT_3! tau: 0 type: input
clamp type: no clamp
min hard con: -6.875000
max hard con: 88513.906250
max inc: 92946.820313
max dec: 92946.820313
priority: 1.000000
convergence: 885.207813
desired: 55573.820818
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -6.875000
fuzzy max: 88513.906250
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 10 name: !4CP005_I:MAIN_3.PNT_3! tau: 0 type: input
clamp type: no clamp
min hard con: -964.843811
max hard con: 152997.093750
max inc: 161660.034439
max dec: 161660.034439
priority: 1.000000
convergence: 1539.619376
desired: 85243.004552
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -964.843811
fuzzy max: 152997.093750
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
```

confidence int: 90.000000

index: 11 name: !4CP005_I:MAIN_3.PNT_4! tau: 0 type: input
clamp type: no clamp
min hard con: -969.726624
max hard con: 137399.390625
max inc: 145287.573111
max dec: 145287.573111
priority: 1.000000
convergence: 1383.691172
desired: 74700.184469
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -969.726624
fuzzy max: 137399.390625
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 12 name: !4CP003_I:MAIN_6.PNT_5! tau: 0 type: input
clamp type: no clamp
min hard con: -1.998000
max hard con: 101.998001
max inc: 109.195801
max dec: 109.195801
priority: 1.000000
convergence: 1.039960
desired: 41.126684
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -1.998000
fuzzy max: 101.998001
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 13 name: !4CP003_I:MAIN_7.PNT_5! tau: 0 type: input
clamp type: no clamp
min hard con: 0.234400
max hard con: 101.248001
max inc: 106.064281
max dec: 106.064281
priority: 1.000000
convergence: 1.010136
desired: 41.106574
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 0.234400
fuzzy max: 101.248001
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 14 name: !4CP001_I:MAIN_3.PNT_7! tau: 0 type: input
clamp type: no clamp
min hard con: 215.842407
max hard con: 626.294922
max inc: 430.975140
max dec: 430.975140
priority: 1.000000
convergence: 4.104525
desired: 456.385122
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 215.842407
fuzzy max: 626.294922
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 15 name: !4CP002_I:MAIN_3.PNT_7! tau: 0 type: input
clamp type: no clamp
min hard con: 939.301575
max hard con: 1045.699951
max inc: 111.718295
max dec: 111.718295
priority: 1.000000
convergence: 1.063984
desired: 998.094123
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 939.301575
fuzzy max: 1045.699951
fuzzy coeff: 1.000000

fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 16 name: !4CP005_I:MAIN_4.PNT_7! tau: 0 type: input
clamp type: no clamp
min hard con: 813.939087
max hard con: 1012.822998
max inc: 208.828107
max dec: 208.828107
priority: 1.000000
convergence: 1.988839
desired: 867.661512
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 813.939087
fuzzy max: 1012.822998
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 17 name: !4CP002_I:MAIN_2.PNT_8! tau: 0 type: input
clamp type: no clamp
min hard con: 927.359924
max hard con: 1033.907349
max inc: 111.874796
max dec: 111.874796
priority: 1.000000
convergence: 1.065474
desired: 993.021201
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 927.359924
fuzzy max: 1033.907349
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 18 name: !4O2CONTROL:O2_MEASURE.RO01! tau: 0 type: input
clamp type: no clamp
min hard con: 1.941900
max hard con: 6.905900
max inc: 5.212200
max dec: 5.212200
priority: 1.000000
convergence: 0.049640
desired: 3.966649
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 1.941900
fuzzy max: 6.905900
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 19 name: !4O2CONTROL:O2_MEASURE.RO02! tau: 0 type: input
clamp type: no clamp
min hard con: 2.089300
max hard con: 7.386800
max inc: 5.562375
max dec: 5.562375
priority: 1.000000
convergence: 0.052975
desired: 4.318681
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 2.089300
fuzzy max: 7.386800
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 20 name: !4AIRHEATER:APA_HEATER.RO01! tau: 0 type: input
clamp type: no clamp
min hard con: 173.996597
max hard con: 300.912811
max inc: 133.262025
max dec: 133.262025
priority: 1.000000
convergence: 1.269162
desired: 222.861629
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 173.996597
fuzzy max: 300.912811

fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 21 name: !4AIRHEATER:APA_HEATER.RO03! tau: 0 type: input
clamp type: no clamp
min hard con: 31.327200
max hard con: 143.694199
max inc: 117.985349
max dec: 117.985349
priority: 1.000000
convergence: 1.123670
desired: 109.215884
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 31.327200
fuzzy max: 143.694199
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 22 name: !4AIRHEATER:BPA_HEATER.RO01! tau: 0 type: input
clamp type: no clamp
min hard con: 181.229996
max hard con: 301.112793
max inc: 125.876937
max dec: 125.876937
priority: 1.000000
convergence: 1.198828
desired: 229.097393
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 181.229996
fuzzy max: 301.112793
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 23 name: !4AIRHEATER:BPA_HEATER.RO03! tau: 0 type: input
clamp type: no clamp
min hard con: 67.062599
max hard con: 136.165207
max inc: 72.557738
max dec: 72.557738
priority: 1.000000
convergence: 0.691026
desired: 110.674492
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 67.062599
fuzzy max: 136.165207
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 24 name: !4AIRHEATER:ASEC_AIRHTR.RO01! tau: 0 type: input
clamp type: no clamp
min hard con: 264.406097
max hard con: 357.407990
max inc: 97.651987
max dec: 97.651987
priority: 1.000000
convergence: 0.930019
desired: 318.599467
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 264.406097
fuzzy max: 357.407990
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 25 name: !4AIRHEATER:ASEC_AIRHTR.RO03! tau: 0 type: input
clamp type: no clamp
min hard con: 41.783001
max hard con: 116.539597
max inc: 78.494425
max dec: 78.494425
priority: 1.000000
convergence: 0.747566
desired: 88.381956
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 41.783001

```
fuzzy max: 116.539597
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 26 name: !4AIRHEATER:BSEC_AIRHTR.RO01! tau: 0 type: input
clamp type: no clamp
min hard con: 262.216492
max hard con: 373.260590
max inc: 116.596303
max dec: 116.596303
priority: 1.000000
convergence: 1.110441
desired: 330.805009
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 262.216492
fuzzy max: 373.260590
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 27 name: !4AIRHEATER:BSEC_AIRHTR.RO03! tau: 0 type: input
clamp type: no clamp
min hard con: 39.116901
max hard con: 118.339600
max inc: 83.183833
max dec: 83.183833
priority: 1.000000
convergence: 0.792227
desired: 89.246055
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 39.116901
fuzzy max: 118.339600
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 28 name: !4CP003_I:MAIN_4.PNT_6! tau: 0 type: input
clamp type: no clamp
min hard con: 0.114633
max hard con: 22.707640
max inc: 23.722657
max dec: 23.722657
priority: 1.000000
convergence: 0.225930
desired: 7.442103
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 0.114633
fuzzy max: 22.707640
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 29 name: !4CP002_I:MAIN_10.PNT_3! tau: 0 type: input
clamp type: no clamp
min hard con: -3396.350830
max hard con: 79283.070313
max inc: 86813.392200
max dec: 86813.392200
priority: 1.000000
convergence: 826.794211
desired: 34883.333938
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -3396.350830
fuzzy max: 79283.070313
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 30 name: !4CP001_I:MAIN_10.PNT_1! tau: 0 type: input
clamp type: no clamp
min hard con: 5.478600
max hard con: 13.034600
max inc: 7.933800
max dec: 7.933800
priority: 1.000000
convergence: 0.075560
desired: 11.579476
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
```

fuzzy min: 5.478600
fuzzy max: 13.034600
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 31 name: !4CP001_I:MAIN_10.PNT_2! tau: 0 type: input
clamp type: no clamp
min hard con: 5.351600
max hard con: 310.429688
max inc: 320.331992
max dec: 320.331992
priority: 1.000000
convergence: 3.050781
desired: 109.045022
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 5.351600
fuzzy max: 310.429688
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 32 name: !4CP005_I:MAIN_12.PNT_4! tau: 0 type: input
clamp type: no clamp
min hard con: -0.818360
max hard con: 99.814453
max inc: 105.664454
max dec: 105.664454
priority: 1.000000
convergence: 1.006328
desired: 76.224008
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -0.818360
fuzzy max: 99.814453
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 33 name: !TMS_Setpoint! tau: 0 type: input
clamp type: no clamp
min hard con: 950.000000
max hard con: 1050.000000
max inc: 105.000000
max dec: 105.000000
priority: 1.000000
convergence: 1.000000
desired: 1000.026695
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 950.000000
fuzzy max: 1050.000000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 34 name: !THRH_Setpoint! tau: 0 type: input
clamp type: no clamp
min hard con: 950.000000
max hard con: 1050.000000
max inc: 105.000000
max dec: 105.000000
priority: 1.000000
convergence: 1.000000
desired: 999.924280
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 950.000000
fuzzy max: 1050.000000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 35 name: !PMS_Setpoint! tau: 0 type: input
clamp type: no clamp
min hard con: 2315.000000
max hard con: 2415.000000
max inc: 105.000000
max dec: 105.000000
priority: 1.000000
convergence: 1.000000
desired: 2364.940099
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000

optimization method: none
fuzzy min: 2315.000000
fuzzy max: 2415.000000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

Output Settings:
----

index: 36 name: !4CP002_I:MAIN_10.PNT_1! tau: 0 type: output
clamp type: no clamp
max inc: 95243.589185
max dec: 95243.589185
priority: 0.000000
convergence: 907.081802
desired: 52364.212299
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: -1756.562988
fuzzy max: 88951.617188
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 37 name: !4CP002_I:MAIN_11.PNT_1! tau: 0 type: output
clamp type: no clamp
max inc: 92765.752734
max dec: 92765.752734
priority: 0.000000
convergence: 883.483359
desired: 42454.790361
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: -13.750000
fuzzy max: 88334.585938
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 38 name: !4CP005_I:MAIN_5.PNT_1! tau: 0 type: output
clamp type: no clamp
max inc: 49.225992
max dec: 49.225992
priority: 0.000000
convergence: 0.468819
desired: 687.868811
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 657.098328
fuzzy max: 703.980225
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 39 name: !4CP002_I:MAIN_10.PNT_2! tau: 0 type: output
clamp type: no clamp
max inc: 92575.653568
max dec: 92575.653568
priority: 0.000000
convergence: 881.672891
desired: 57514.137485
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: -8.593800
fuzzy max: 88158.695313
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 40 name: !4CP002_I:MAIN_11.PNT_2! tau: 0 type: output
clamp type: no clamp
max inc: 91286.507813
max dec: 91286.507813
priority: 0.000000
convergence: 869.395313
desired: 55484.448221
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 10.312500
fuzzy max: 86949.843750
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 41 name: !4CP003_I:MAIN_6.PNT_2! tau: 0 type: output
clamp type: no clamp
max inc: 103.841322
max dec: 103.841322
priority: 0.000000
convergence: 0.988965
desired: 41.314078
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 1.257800
fuzzy max: 100.154297
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 42 name: !4CP003_I:MAIN_7.PNT_2! tau: 0 type: output
clamp type: no clamp
max inc: 105.291162
max dec: 105.291162
priority: 0.000000
convergence: 1.002773
desired: 41.050029
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 0.168000
fuzzy max: 100.445297
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 43 name: !4CP005_I:MAIN_4.PNT_2! tau: 0 type: output
clamp type: no clamp
max inc: 36.542871
max dec: 36.542871
priority: 0.000000
convergence: 0.348027
desired: 687.867800
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 667.721985
fuzzy max: 702.524719
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 44 name: !4CP001_I:MAIN_1.PNT_3! tau: 0 type: output
clamp type: no clamp
max inc: 509.475586
max dec: 509.475586
priority: 0.000000
convergence: 4.852148
desired: 2364.221678
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 2218.164063
fuzzy max: 2703.378906
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 45 name: !4CP002_I:MAIN_11.PNT_3! tau: 0 type: output
clamp type: no clamp
max inc: 92946.820313
max dec: 92946.820313
priority: 0.000000
convergence: 885.207813
desired: 55573.820818
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: -6.875000
fuzzy max: 88513.906250
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 46 name: !4CP005_I:MAIN_3.PNT_3! tau: 0 type: output
clamp type: no clamp
max inc: 161660.034439
max dec: 161660.034439
priority: 0.000000
convergence: 1539.619376
desired: 85243.004552
min desired: 0.000000

max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: -964.843811
fuzzy max: 152997.093750
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 47 name: !4CP005_I:MAIN_3.PNT_4! tau: 0 type: output
clamp type: no clamp
max inc: 145287.573111
max dec: 145287.573111
priority: 0.000000
convergence: 1383.691172
desired: 74700.184469
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: -969.726624
fuzzy max: 137399.390625
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 48 name: !4CP003_I:MAIN_6.PNT_5! tau: 0 type: output
clamp type: no clamp
max inc: 109.195801
max dec: 109.195801
priority: 0.000000
convergence: 1.039960
desired: 41.126684
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: -1.998000
fuzzy max: 101.998001
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 49 name: !4CP003_I:MAIN_7.PNT_5! tau: 0 type: output
clamp type: no clamp
max inc: 106.064281
max dec: 106.064281
priority: 0.000000
convergence: 1.010136
desired: 41.106574
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 0.234400
fuzzy max: 101.248001
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 50 name: !4CP001_I:MAIN_3.PNT_7! tau: 0 type: output
clamp type: no clamp
max inc: 430.975140
max dec: 430.975140
priority: 0.000000
convergence: 4.104525
desired: 456.385122
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 215.842407
fuzzy max: 626.294922
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 51 name: !4CP002_I:MAIN_3.PNT_7! tau: 0 type: output
clamp type: no clamp
max inc: 111.718295
max dec: 111.718295
priority: 0.000000
convergence: 1.063984
desired: 998.094123
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 939.301575
fuzzy max: 1045.699951
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 52 name: !4CP005_I:MAIN_4.PNT_7! tau: 0 type: output
clamp type: no clamp
max inc: 208.828107
max dec: 208.828107
priority: 0.000000
convergence: 1.988839
desired: 867.661512
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 813.939087
fuzzy max: 1012.822998
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 53 name: !4CP002_I:MAIN_2.PNT_8! tau: 0 type: output
clamp type: no clamp
max inc: 111.874796
max dec: 111.874796
priority: 0.000000
convergence: 1.065474
desired: 993.021201
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 927.359924
fuzzy max: 1033.907349
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 54 name: !4O2CONTROL:O2_MEASURE.RO01! tau: 0 type: output
clamp type: no clamp
max inc: 5.212200
max dec: 5.212200
priority: 0.000000
convergence: 0.049640
desired: 3.966649
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 1.941900
fuzzy max: 6.905900
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 55 name: !4O2CONTROL:O2_MEASURE.RO02! tau: 0 type: output
clamp type: no clamp
max inc: 5.562375
max dec: 5.562375
priority: 0.000000
convergence: 0.052975
desired: 4.318681
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 2.089300
fuzzy max: 7.386800
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 56 name: !4AIRHEATER:APA_HEATER.RO01! tau: 0 type: output
clamp type: no clamp
max inc: 133.262025
max dec: 133.262025
priority: 0.000000
convergence: 1.269162
desired: 222.861629
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 173.996597
fuzzy max: 300.912811
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 57 name: !4AIRHEATER:APA_HEATER.RO03! tau: 0 type: output
clamp type: no clamp
max inc: 117.985349
max dec: 117.985349
priority: 0.000000
convergence: 1.123670
desired: 109.215884
min desired: 0.000000

max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 31.327200
fuzzy max: 143.694199
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 58 name: !4AIRHEATER:BPA_HEATER.RO01! tau: 0 type: output
clamp type: no clamp
max inc: 125.876937
max dec: 125.876937
priority: 0.000000
convergence: 1.198828
desired: 229.097393
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 181.229996
fuzzy max: 301.112793
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 59 name: !4AIRHEATER:BPA_HEATER.RO03! tau: 0 type: output
clamp type: no clamp
max inc: 72.557738
max dec: 72.557738
priority: 0.000000
convergence: 0.691026
desired: 110.674492
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 67.062599
fuzzy max: 136.165207
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 60 name: !4AIRHEATER:ASEC_AIRHTR.RO01! tau: 0 type: output
clamp type: no clamp
max inc: 97.651987
max dec: 97.651987
priority: 0.000000
convergence: 0.930019
desired: 318.599467
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 264.406097
fuzzy max: 357.407990
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 61 name: !4AIRHEATER:ASEC_AIRHTR.RO03! tau: 0 type: output
clamp type: no clamp
max inc: 78.494425
max dec: 78.494425
priority: 0.000000
convergence: 0.747566
desired: 88.381956
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 41.783001
fuzzy max: 116.539597
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 62 name: !4AIRHEATER:BSEC_AIRHTR.RO01! tau: 0 type: output
clamp type: no clamp
max inc: 116.596303
max dec: 116.596303
priority: 0.000000
convergence: 1.110441
desired: 330.805009
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 262.216492
fuzzy max: 373.260590
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 63 name: !4AIRHEATER:BSEC_AIRHTR.RO03! tau: 0 type: output
clamp type: no clamp
max inc: 83.183833
max dec: 83.183833
priority: 0.000000
convergence: 0.792227
desired: 89.246055
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 39.116901
fuzzy max: 118.339600
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 64 name: !4CP003_I:MAIN_4.PNT_6! tau: 0 type: output
clamp type: no clamp
max inc: 23.722657
max dec: 23.722657
priority: 0.000000
convergence: 0.225930
desired: 7.442103
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 0.114633
fuzzy max: 22.707640
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 65 name: !4CP002_I:MAIN_10.PNT_3! tau: 0 type: output
clamp type: no clamp
max inc: 86813.392200
max dec: 86813.392200
priority: 0.000000
convergence: 826.794211
desired: 34883.333938
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: -3396.350830
fuzzy max: 79283.070313
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 66 name: !4CP001_I:MAIN_10.PNT_1! tau: 0 type: output
clamp type: no clamp
max inc: 7.933800
max dec: 7.933800
priority: 0.000000
convergence: 0.075560
desired: 11.579476
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 5.478600
fuzzy max: 13.034600
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 67 name: !4CP001_I:MAIN_10.PNT_2! tau: 0 type: output
clamp type: no clamp
max inc: 320.331992
max dec: 320.331992
priority: 0.000000
convergence: 3.050781
desired: 109.045022
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 5.351600
fuzzy max: 310.429688
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

index: 68 name: !4CP005_I:MAIN_12.PNT_4! tau: 0 type: output
clamp type: no clamp
max inc: 105.664454
max dec: 105.664454
priority: 0.000000
convergence: 1.006328
desired: 76.224008
min desired: 0.000000

```
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: -0.818360
fuzzy max: 99.814453
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 69 name: !TMS_Setpoint! tau: 0 type: output
clamp type: no clamp
max inc: 105.000000
max dec: 105.000000
priority: 0.000000
convergence: 1.000000
desired: 1000.026695
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 950.000000
fuzzy max: 1050.000000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 70 name: !THRH_Setpoint! tau: 0 type: output
clamp type: no clamp
max inc: 105.000000
max dec: 105.000000
priority: 0.000000
convergence: 1.000000
desired: 999.924280
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 950.000000
fuzzy max: 1050.000000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000

      index: 71 name: !PMS_Setpoint! tau: 0 type: output
clamp type: no clamp
max inc: 105.000000
max dec: 105.000000
priority: 0.000000
convergence: 1.000000
desired: 2364.940099
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: use-desired
fuzzy min: 2315.000000
fuzzy max: 2415.000000
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000



Global Optimizer Parameters:
----------------------------
autoscale off
timeout   0.000000

Combined Constraints:
---------------------------

none


Applied Transforms.
-------------------
```

```
[GENERAL]

[NOX]
OutputName=NOX_LBMMBTU
Enabled=1
UseModelBias=1
ManualBias=0.0
LowBiasLimit=-0.4
HighBiasLimit=0.05
ModelPath=c:\OnLineErrorCorrection\LIB\GBCorrectModelLoader.dll
ModelData=c:\OnLineErrorCorrection\DATA\GBC1.ini|BC_NOX

[TMS]
OutputName=TMS
Enabled=1
UseModelBias=1
ManualBias=0.0
LowBiasLimit=-5
HighBiasLimit=5
ModelPath=c:\OnLineErrorCorrection\LIB\GBCorrectModelLoader.dll
ModelData=c:\OnLineErrorCorrection\DATA\GBC1.ini|BC_TMS

[PMS]
OutputName=PMS
Enabled=1
UseModelBias=1
ManualBias=0.0
LowBiasLimit=-25
HighBiasLimit=25
ModelPath=c:\OnLineErrorCorrection\LIB\GBCorrectModelLoader.dll
ModelData=c:\OnLineErrorCorrection\DATA\GBC1.ini|BC_PMS

[THRH]
OutputName=THRH
Enabled=1
UseModelBias=1
ManualBias=0.0
LowBiasLimit=-5
HighBiasLimit=5
ModelPath=c:\OnLineErrorCorrection\LIB\GBCorrectModelLoader.dll
ModelData=c:\OnLineErrorCorrection\DATA\GBC1.ini|BC_THRH
```

```
[NULLMODEL]
ModelType = c:\uop_software\olec\LIB\nullmodel

[RunAvgY10]
ModelType = c:\uop_software\olec\LIB\RunAvgY
iSY = 10

[RunAvgY30]
ModelType = c:\uop_software\olec\LIB\RunAvgY
iSY = 30

[BiasAdjust]
ModelType = c:\uop_software\olec\LIB\BiasAdjust

[ConstantModel1]
ModelType = c:\uop_software\olec\LIB\ConstantModel
dConstant = 0.123

[BC_NOX]
ModelType = c:\uop_software\olec\LIB\tlm1
DefaultTagList = c:\uop_software\olec\DATA\DefaultTagList.txt
SubModel = c:\uop_software\olec\DATA\GBC1.ini|RunAvgY10
References = NOX

[BC_EFF]
ModelType = c:\uop_software\olec\LIB\tlm1
DefaultTagList = c:\uop_software\olec\DATA\DefaultTagList.txt
SubModel = c:\uop_software\olec\DATA\GBC1.ini|RunAvgY30
References = EFF

[BC_TMS]
ModelType = c:\uop_software\olec\LIB\tlm1
DefaultTagList = c:\uop_software\olec\DATA\DefaultTagList.txt
SubModel = c:\uop_software\olec\DATA\GBC1.ini|RunAvgY30
References = TMS

[BC_PMS]
ModelType = c:\uop_software\olec\LIB\tlm1
DefaultTagList = c:\uop_software\olec\DATA\DefaultTagList.txt
SubModel = c:\uop_software\olec\DATA\GBC1.ini|RunAvgY30
References = PMS

[BC_THRH]
ModelType = c:\uop_software\olec\LIB\tlm1
DefaultTagList = c:\uop_software\olec\DATA\DefaultTagList.txt
SubModel = c:\uop_software\olec\DATA\GBC1.ini|RunAvgY30
References = THRH

;=====================================================================================
========
;
; Input / Output Block Object
;
;       Currently these must be in the same ini file as models that refer to them
;       Currently these are pretty dumb,only doing  limit checking & then scaling
;       Would like to add a parser so simple preprocessing can be done
;
;       Source - The tag name
;       ValidLowerBound - If below this level, set error flag and don't use,
```

```
defaults -inf
;       ValidUpperBound - If above this level, set error flag and don't use,
defaults +inf
;       ScaleLower  - Used to scale source to 0.0 to 1.0, default is 0.0
;       ScaleUpper - Used to scale source to 0.0 to 1.0 , default is 0.0
;                -> if (sl ~= su)  y = (x - sl) / (su - sl) ; else y = x (i.e. no
scaling)
;
;==================================================================================
========

[NOX]
Source = NOX_LBMMBTU
ValidLowerBound = 0.01
ValidUpperBound = 0.65

[EFF]
Source = EFF
ValidLowerBound = 85
ValidUpperBound = 90

[TMS]
Source = TMS
ValidLowerBound = 900
ValidUpperBound = 1100

[PMS]
Source = PMS
ValidLowerBound = 1500
ValidUpperBound = 3000

[THRH]
Source = THRH
ValidLowerBound = 900
ValidUpperBound = 1100
```

APPENDIX G

TURBINE OPTIMIZER ACTIVE MODEL INFORMATION

```
ModelFileName = c:\gnocis\activemodel\ham4_2k_turb2
ComboFileName = c:\gnocis\activemodel\U4_combo.ini
ConstFileName = c:\gnocis\activemodel\U4_const.ini
TagFileName = c:\gnocis\activemodel\U4_tag.ini
OutFileName = c:\gnocis\activemodel\U4_outputs.ini
Debug = 5
```

```
# File: Const.ini
#
# Version 1.0 - Original 12/01/2000 - JMF
#             - Hammond 4 Unit Optimization Project
#
#
# ----------------- CONTROL ---------------------

[PMS]
SETPOINTINPUT = 1
BIASOUT = 4UMSBOILER:THTPSP_RATEL.OUT
GNOCISBIAS = PMSGNOCISBIAS
MINTAG = 4TURB:INMIN_1
MAXTAG = 4TURB:INMAX_1
CLAMPTAG = 4TURB:CLAMPED1
#LOCALREMOTETAG =
#MOVEDELTA = 100
#MOVECOST = 0.01

[TMS]
SETPOINTINPUT = 1
BIASOUT = 4UPSHSPRAY:FSHOUT_CTRLR.SPT
GNOCISBIAS = TMSGNOCISBIAS
MINTAG = 4TURB:INMIN_2
MAXTAG = 4TURB:INMAX_2
CLAMPTAG = 4TURB:CLAMPED2
#LOCALREMOTETAG =
#MOVEDELTA = 100
#MOVECOST = 0.01

[THRH]
SETPOINTINPUT = 1
BIASOUT = 4SHPASS:PASDMP_CTRLR.SPT
GNOCISBIAS = THRHGNOCISBIAS
MINTAG = 4TURB:INMIN_3
MAXTAG = 4TURB:INMAX_3
CLAMPTAG = 4TURB:CLAMPED3
#LOCALREMOTETAG =
#MOVEDELTA = 100
#MOVECOST = 0.01

[GROSS_MW]
CLAMP = 1

#
#------------------ OUTPUTS ---------------------
#
[TOT_PERCENT_HRDEV]
MINTAG = 4TURB:INMIN_5
MAXTAG = 4TURB:INMAX_5
USEFUZZY = 1
FUZZYCOEFF = 10000000

#
# ------------------- MISC ----------------------
#
LOADTAG = 4CP001_I:MAIN_4.PNT_3
NUMCONTROL = 4
NUMMODELOUT = 1
```

```
UPDATETAG = UPDATETAG
CLOSEDLOOPTAG = CLOSEDLOOP
#REMOVEBIASTAG =
MATRIXSIZE = 1
INTCPPORT = 2003
OUTTCPPORT1 = 2020
OUTTCPADDRESS1 = 148.199.229.116
OUTTCPPORT2 = 2021
OUTTCPADDRESS2 = 148.199.229.74
OUTTCPPORT3 = 2011
OUTTCPADDRESS3 = 148.199.229.74
OUTPUTMODEL = c:\gnocis\activemodel\ham4_2k_biasturb2
```

4CP001_I:MAIN_1.PNT_3	MAIN STEAM PRESSURE
4CP002_I:MAIN_3.PNT_7	MAIN STEAM TEMPERATURE
4CP002_I:MAIN_2.PNT_8	HOT REHEAT TEMPERATURE
4CP001_I:MAIN_4.PNT_3	GENERATION (GROSS)
4UMSBOILER:THTPSP_RATEL.OUT	PMS SETPOINT
4UPSHSPRAY:FSHOUT_CTRLR.SPT	TMS SETPOINT
4SHPASS:PASDMP_CTRLR.SPT	THRH SETPOINT
PMSGNOCISBIAS	
TMSGNOCISBIAS	
THRHGNOCISBIAS	
UPDATETAG	
CLOSEDLOOPTAG	
4TURB:INMAX_1	MAIN STEAM PRESSURE MAX
4TURB:INMAX_2	MAIN STEAM TEMPERATURE MAX
4TURB:INMAX_3	HOT REHEAT TEMPERATURE MAX
4TURB:INMAX_5	TOTAL PERCENT HEAT RECOVERY MAX
4TURB:INMIN_1	MAIN STEAM PRESSURE MIN
4TURB:INMIN_2	MAIN STEAM TEMPERATURE MIN
4TURB:INMIN_3	HOT REHEAT TEMPERATURE MIN
4TURB:INMIN_5	TOTAL PERCENT HEAT RECOVERY MIN
4TURB:CLAMPED1	MAIN STEAM PRESSURE CLAMP
4TURB:CLAMPED2	MAIN STEAM TEMPERATURE CLAMP
4TURB:CLAMPED3	HOT REHEAT TEMPERATURE CLAMP

```
GC_PMS                   4TURB:INRC_1
GC_TMS                   4TURB:INRC_2
GC_THRH                  4TURB:INRC_3
GC_GROSS_MW              4TURB:INRC_4
GCP_TOT_PERCENT_HRDEV    4TURB:OUTP
GB_PMS                   4TURB:INBC_1
GB_TMS                   4TURB:INBC_2
GB_THRH                  4TURB:INBC_3
GB_GROSS_MW              4TURB:INBC_4
GBP_TOT_PERCENT_HRDEV    4TURB:OUTB
GP_TOT_PERCENT_HRDEV     4TURB:OUTM
GCO_TOT_PERCENT_HRDEV    4TURB:OUTI
GCV_PMS                  4TURB:INRB_1
GCV_TMS                  4TURB:INRB_2
GCV_THRH                 4TURB:INRB_3
GPW_TOT_PERCENT_HRDEV    NULL
GS_STATUS                NULL
```

```
Dataset: /home/gnocis/hammond/ham2000/Ham4_apr_oct2000_turb
Model: /home/gnocis/hammond/ham2000/ham4_2k_turb2
Time Interval:
Filter used: None.

Model Variables:
----------------

     index# (C Language)                       input_name   Time Delay
     -------------------                       ----------   ----------
                       0                            !PMS!            0
                       1                            !TMS!            0
                       2                           !THRH!            0
                       3                       !GROSS_MW!            0

     index# (C Language)                      output_name   Time Delay
     -------------------                      -----------   ----------
                       4             !TOT_PERCENT_HRDEV!            0
Raw Tags:
---------
Note: Tag ids are language independent.

   TagId                            name                            comment     type
   -----                            ----                            -------     ----
       1                            TIME                      DATE AND TIME datetime
       2            4CP001_I:MAIN_1.PNT_3               MAIN STEAM PRESSURE    float
       3            4CP002_I:MAIN_3.PNT_7            MAIN STEAM TEMPERATURE    float
       4            4CP002_I:MAIN_2.PNT_8            HOT REHEAT TEMPERATURE    float
       5            4CP001_I:MAIN_4.PNT_3                GENERATION (GROSS)    float

Trans. Tags:
------------
Note: Tag ids are language independent.

   TagId                            name                            comment     type
   -----                            ----                            -------     ----
       1                            TIME                      DATE AND TIME datetime
       2            4CP001_I:MAIN_1.PNT_3               MAIN STEAM PRESSURE    float
       3            4CP002_I:MAIN_3.PNT_7            MAIN STEAM TEMPERATURE    float
       4            4CP002_I:MAIN_2.PNT_8            HOT REHEAT TEMPERATURE    float
       5            4CP001_I:MAIN_4.PNT_3                GENERATION (GROSS)    float
       6                             PMS                                       float
       7                             TMS                                       float
       8                            THRH                                       float
       9                        GROSS_MW                                       float
      10                    LOAD_PERCENT                                       float
      11                         HRDEV_1                                       float
      12                         HRDEV_2                                       float
      13                         HRDEV_3                                       float
      14              TOT_PERCENT_HRDEV                                        float
Model Settings:
--------------

Input Settings:
---------------

      index: 0 name: PMS tau: 0 type: input
clamp type: no clamp
min hard con: 2257.050049
max hard con: 2466.090088
max inc: 219.494995
max dec: 219.494995
priority: 1.000000
convergence: 2.090430
desired: 2364.479980
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 2257.050049
fuzzy max: 2466.090088
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000

      index: 1 name: TMS tau: 0 type: input
clamp type: no clamp
min hard con: 969.010986
max hard con: 1025.400024
max inc: 59.209599
max dec: 59.209599
priority: 1.000000
convergence: 0.563901
desired: 996.924988
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 969.010986
fuzzy max: 1025.400024
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000

      index: 2 name: THRH tau: 0 type: input
clamp type: no clamp
min hard con: 911.114990
max hard con: 1038.109985
max inc: 133.345001
```

```
max dec: 133.345001
priority: 1.000000
convergence: 1.269950
desired: 989.265991
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 911.114990
fuzzy max: 1038.109985
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000

      index: 3 name: GROSS_MW tau: 0 type: input
clamp type: compute
min hard con: 180.750000
max hard con: 529.130005
max inc: 365.799011
max dec: 365.799011
priority: 1.000000
convergence: 3.483800
desired: 415.239014
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: 180.750000
fuzzy max: 529.130005
fuzzy coeff: 1.000000
fuzzy used: fuzzy-disabled
confidence int: 90.000000

Output Settings:
---------------

      index: 4 name: TOT_PERCENT_HRDEV tau: 0 type: output
clamp type: no clamp
max inc: 2.906210
max dec: 2.906210
priority: 0.000000
convergence: 0.027678
desired: 0.443600
min desired: 0.000000
max desired: 0.000000
cost coeff: 1.000000
cost: 0.000000
optimization method: none
fuzzy min: -0.702700
fuzzy max: 2.065100
fuzzy coeff: 1.000000
fuzzy used: fuzzy-enabled
confidence int: 90.000000



Global Optimizer Parameters:
----------------------------
autoscale off
timeout   0.000000

Combined Constraints:
---------------------------

none


Applied Transforms.
-------------------

transform { name "!PMS!" expr "!4CP001_I:MAIN_1.PNT_3!" }
transform { name "!TMS!" expr "!4CP002_I:MAIN_3.PNT_7!" }
transform { name "!THRH!" expr "!4CP002_I:MAIN_2.PNT_8!" }
transform { name "!GROSS_MW!" expr "!4CP001_I:MAIN_4.PNT_3!" }
transform { name "!LOAD_PERCENT!" expr "!GROSS_MW! / 485.0 * 100.0" }
transform { name "!HRDEV_1!" expr " $if(!LOAD_PERCENT! < 75.0, !PMS! * (.0516 * !LOAD_PERCENT! - 5.87)  / 1000. + 14.75 - .1246 *
!LOAD_PERCENT!, !PMS! * (.0094 * !LOAD_PERCENT! - 2.705)  / 1000. + 6.534 - .0227 * !LOAD_PERCENT!) " }
transform { name "!HRDEV_2!" expr " $if(!LOAD_PERCENT! < 75.0, !TMS! * ( - .006 * !LOAD_PERCENT! - 1.25)  / 100. + .06 * !
LOAD_PERCENT! + 12.5,  - .017 * !TMS! + 17.0) " }
transform { name "!HRDEV_3!" expr " $if(!LOAD_PERCENT! < 75.0, !THRH! * (.003 * !LOAD_PERCENT! - 1.575 )  / 100.0 + 15.75 - .03 *
!LOAD_PERCENT!,  - !THRH! * .0135 + 13.5) " }
transform { name "!TOT_PERCENT_HRDEV!" expr "!HRDEV_1! + !HRDEV_2! + !HRDEV_3!" }
```

APPENDIX H

GBCORRECT MODEL TYPES

Model Name	Batcher1
Library Name	Batcher1.dll
Source Directory	Batcher1
Description	Batches inputs into sub-models during updates. During runs, no batching is performed.
Parameters	
ModelType	%OLEC_LIB_DIR%/Batcher1
SubModel	Sub-model to batch inputs in to INI file | INI section
BatchPeriod	Period (in number of calls) to store data before sending data to sub-models.
Example Ini File	[NOXC1_BATCHER] ModelType = %OLEC_LIB_DIR%/BATCHER1 SubModel = "%OLEC_DATA_DIR%/ModelTest.ini|NOXC1" BatchPeriod = 30
Notes	

Model Name	BiasAdjust
Library Name	BiasAdjust.dll
Source Directory	BiasAdjust
Description	Returns the last error as the current error. Since the error tends to be highly auto-correlated for small delays, this may be useful, but the other filter models are probably better even for this case.
Parameters	
ModelType	%OLEC_LIB_DIR%/ BiasAdjust
Example Ini File	[BiasAdjust] ModelType = %OLEC_LIB_DIR%/BiasAdjust
Notes	

Model Name	ConstantModel
Library Name	constantmodel.dll
Source Directory	ConstantModel
Description	Model that returns a constant. Used primarily for testing.
Parameters	
ModelType	%OLEC_LIB_DIR%/ConstantModel
dConstant	Constant to be used (Default = 0.0)
Example Ini File	[ConstantModel1] ModelType = %OLEC_LIB_DIR%/ConstantModel dConstant = 0.123
Notes	

Model Name	DRBF1
Library Name	DRBF1.dll
Source Directory	DRBF1_Using_MATCOM
Description	Model based on an adaptive radial basis function neural network.
Parameters	
ModelType	%OLEC_LIB_DIR%/ drbf1
Sigma0	Width of basis function (Default=8.0)
epsmax	Largest scale of interest (Default=2.0)
epsmin	Smallest scale of interest (Default=0.2)
emin	Error criteria (Default=0.02)
R	Measurement noise variance (Default=1.0)
k	Overlap parameter (Default=0.87)
P0	Uncertainty parameter (Default=1.0)
Q0	Scalar that determines random step size (Default=0.02)
lambda	Decay constant (Default=0.977)
n	Adaptation step size (Default=0.1)
UpdateMode	Determines center update algorithm (Default=3) 1 – Extended Kalman filter (memory and calculation intensive) 2 – Euler 1 (less memory, less accurate) 3 – Euler 2 (still less memory)
MaxCenters	The maximum number of centers that can be added (Default=30)
Example Ini File	[BiasAdjust] ModelType = %OLEC_LIB_DIR%/drbf1
Notes	This model does not directly support initialization through ini files. Parameters must be set using DRBF2, GenericModel models, or TLM1.

Model Name	DRBF2
Library Name	DRBF2.dll
Source Directory	DRBF2_Using_CModelContainer
Description	Model based on an adaptive radial basis function neural network.
Parameters	
ModelType	%OLEC_LIB_DIR%/ drbf1
Sigma0	Width of basis function (Default=8.0)
epsmax	Largest scale of interest (Default=2.0)
epsmin	Smallest scale of interest (Default=0.2)
emin	Error criteria (Default=0.02)
R	Measurement noise variance (Default=1.0)
k	Overlap parameter (Default=0.87)
P0	Uncertainty parameter (Default=1.0)
Q0	Scalar that determines random step size (Default=0.02)
lambda	Decay constant (Default=0.977)
n	Adaptation step size (Default=0.1)
UpdateMode	Determines center update algorithm (Default=3) 1 – Extended Kalman filter (memory and calculation intensive) 2 – Euler 1 (less memory, less accurate) 3 – Euler 2 (still less memory)
MaxCenters	The maximum number of centers that can be added (Default=30)
ModelFilename	Default save model filename.
Example Ini File	[DRBF2] ModelType = %OLEC_LIB_DIR%/ DRBF2 Sigma0 = 8.0 epsmax = 2.0 epsmin = 0.02 emin = 0.05 R = 1.0 k = 0.87 P0 = 1.0 Q0 = 0.02 lambda = 0.977 n = 0.01 emin = 0.02 UpdateMode = 3 MaxCenters = 50 ModelFilename = c:/temp/NOXC1.MAT
Notes	This model using DRBF1 as the calculation model.

Model Name	GBCorrectModelLoader
Library Name	GBCorrectModelLoader.dll
Source Directory	GBCorrectModelLoader
Description	Can be used to load models from GBCorrect.
Parameters	<< not used >>
ModelType	
Example Ini File	<< not used >>
Notes	Specify the underlying model in the LoadModel argument INI_File|INI_Section

Model Name	MLP1
Library Name	MLP1.dll
Source Directory	MLP1_Using_MATCOM
Description	Adaptive Multiple-Layer Perceptron (2 layer) neural network.
Parameters	
ModelType	%OLEC_LIB_DIR%/ MLP1
iNAF	Number of activation functions (Default=5*NumInputs^2)
dLR	Learning rate (Default=0.2)
ModelFilename	Default save model filename
Example Ini File	[DRBF2] ModelType = %OLEC_LIB_DIR%/ DRBF2 dLR = 0.01 ModelFilename = c:/temp/NOXC1.MAT
Notes	

Model Name	RunAvg
Library Name	RunAvg.dll
Source Directory	RunAvg
Description	During model updates, collects the inputs (X and Y) over a specified period passing the average to a submodel. During runs, does not filter.
Parameters	
ModelTyp	%OLEC_LIB_DIR%/ RunAvg
SubModel	Sub-model to batch inputs in to INI file | INI section
AvgPeriod	Averaging period (Default=30)
Example Ini File	[RunAvg_BiasAdjust] ModelType = %OLEC_LIB_DIR%/RunAvg SubModel = "%OLEC_DATA_DIR%/ModelTest.ini|BiasAdjust" AvgPeriod = 30
Notes	

Model Name	RunAvgY
Library Name	RunAvgY.dll
Source Directory	RunAvgY
Description	Update: Collects the inputs over a specified period returning the average. Run: Returns the last average from update
Parameters	
ModelTyp	%OLEC_LIB_DIR%/ RunAvgY
iSY	Averaging period (Default=30)
Example Ini File	[RunAvgY1] ModelType = %OLEC_LIB_DIR%/RunAvgY iSY = 25
Notes	

Model Name	TLM1
Library Name	TLM1.dll
Source Directory	TLM1
Description	Serves as the front end to other models, allowing setting of inputs by name and implementing error checking on these inputs.
Parameters	
ModelTyp	%OLEC_LIB_DIR%/ TLM1
SubModel	Sub-model to batch inputs in to INI file | INI section
DefaultTagList	Tag list to use if one is not already loaded
Inputs	List of inputs by name (input blocks), all must be good to update
References	List of references by name (input blocks), all must be good to update
SubModelSetParameter	Can set sub-model parameters by this field
SubModelSetParameter1	Can set sub-model parameters by this field
SubModelSetParameter2	Can set sub-model parameters by this field
SubModelSetParameter3	Can set sub-model parameters by this field
SubModelSetParameter4	Can set sub-model parameters by this field
Outputs	List of outputs by name. For GBCorrect, not used.
ErrorTrapLevel	Error trap level (Default=2). Returns NaNs for errors above this severity.
Example Ini File	[NOXC2] ModelType = "%OLEC_LIB_DIR%/tlm1 DefaultTagList = "%OLEC_DATA_DIR%/DefaultTagList.txt" SubModel = "%OLEC_DATA_DIR%/ModelTest.ini|DRBF2" Inputs = WMILLAC, O2 References = NOX_Actual SubModelSetParameter = “MaxCenters=10” [O2] Source = AVG_O2 ValidLowerBound = 2.0 ValidUpperBound = 6.0 ScaleUpper = 0.0 ScaleLower = 10.0
Notes	