Tuning PEPSE® Boiler Models Using Compiled Algorithms

James E. Blakeley

HALLIBURTON NUS Environmental Corporation

TUNING PEPSE® BOILER MODELS USING COMPILED ALGORITHMS

Jeb Blakeley
HALLIBURTON NUS Environmental Corp.

ABSTRACT

A detailed PEPSE model was built for a Babcock and Wilcox boiler rated for 2,160,000 lb/hr maximum continuous steam flow. This model was tuned using vendor performance data and a compiled algorithm that determined the correct combination of fouling factors and heat transfer coefficient multipliers required to match the vendor's data. This application of a compiled algorithm is fast, easy to use, and greatly enhances the analyst's ability to perform multiple stacked case runs to select the best parameters for a particular model. The compiled algorithm and PEPSE controls and operations are described along with lessons learned in the implementation of the boiler tuning method.

INTRODUCTION

A detailed PEPSE boiler model was constructed to study the repowering of an existing unit with a Babcock and Wilcox oil and gas fired drum boiler. The completed boiler model was integrated with a turbine cycle model and a heat recovery steam generator to study repowering options. This paper focuses on the use of compiled algorithms to tune the boiler model heat transfer coefficient multipliers and fouling factors to match the stage heat transfer with the vendor performance data.

Figure 1 presents a simplified diagram of the boiler. A summary of the major boiler features shows that at the rated steam conditions of 2475 psig, 1053°F main steam, and 1003 °F reheat; the maximum continuous rated steam flow is 2,160,000 lb/hr. It is a natural circulation, pressure fired boiler with gas recirculation, primary, secondary, and tertiary superheaters, a reheater, a bare tube economizer, and a membrane wall furnace. The boiler is separated into three sections by two division walls.

The process of developing the boiler model included constructing a simplified model to verify boundary conditions at two loads. The gas recirculation fan was included in this model to properly account for all energy sources into the boiler. The overall efficiency and the individual heat losses were matched to within 0.003%.

The verified boundary conditions were transferred to a detailed PEPSE boiler model. The calculated heat transfer in each convective stage of the detailed model was then tuned such that the steam temperatures at the exit of each stage matched the values from the vendor's literature. PEPSE controls and a compiled algorithm were used to aid this process.

Compiled algorithms are new to Version 57. They are similar to the PEPSE operation feature in that both allow the user to write custom equations that will be solved by PEPSE. However, the compiled algorithm is more versatile. One compiled algorithm can include

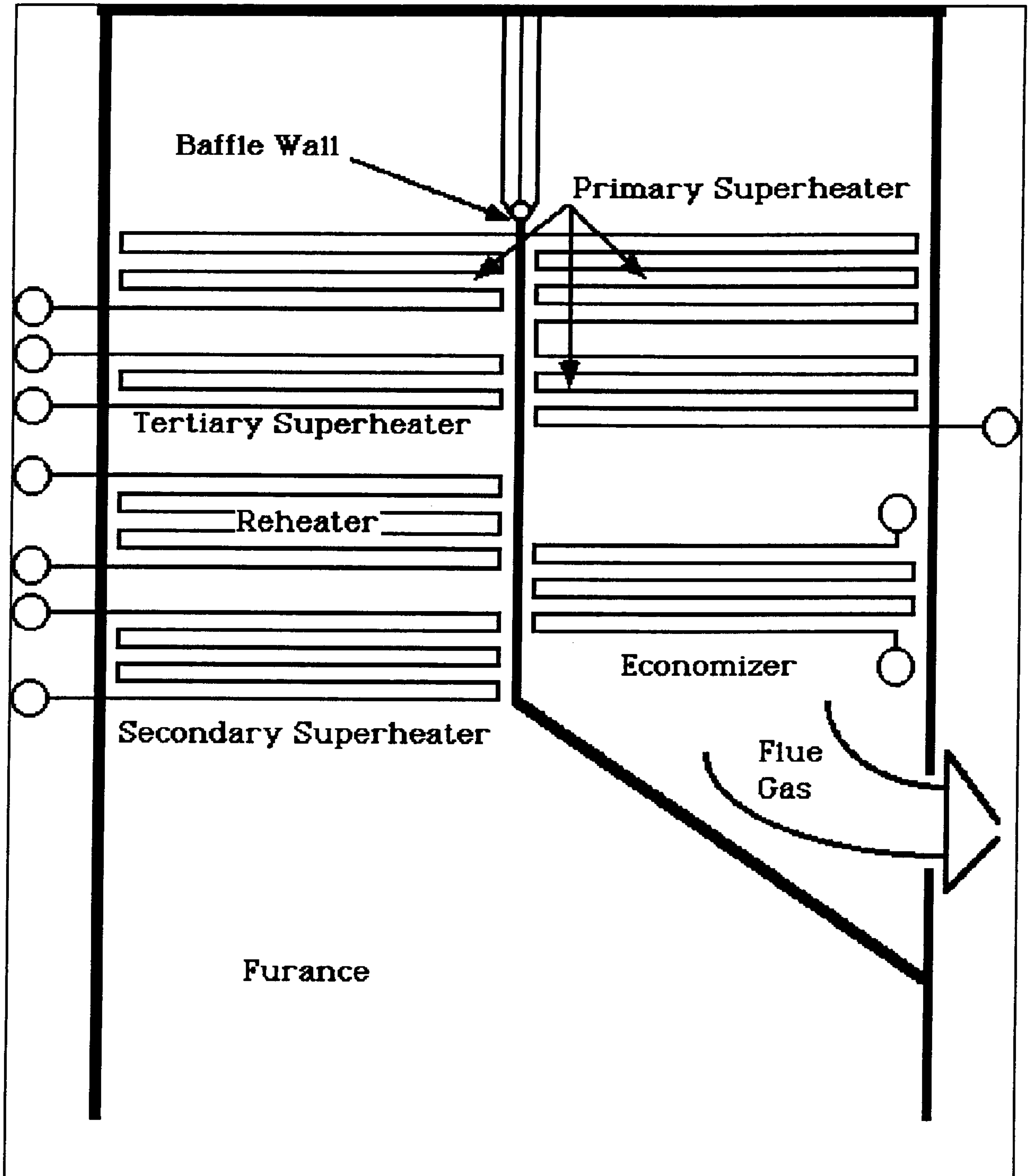

Figure 1. Boiler Cross Section.

many complex equations involving many input variables. A compiled algorith can have IF-THEN-ELSE constructs, and several output variables can be produced by one compiled algorithm.

The boiler model described in this paper was built using PEPSE version 57 with the graphical man-machine interface (MMI).

SOLUTION METHOD

Tuning the heat transfer is accomplished by picking a fouling resistance and a heat transfer coefficient multiplier that produce the desired steam exit temperature. In PEPSE, the variable used to represent fouling resistance is called RHTM and is the 33rd word on the 70YYYS cards. It can be found in the MMI on the third page of the <u>(B) Component Data/(C) Heat Exchanger/Type 28 Convective Stage</u> menu. RHTM is an additive term in the heat transfer resistance of the tube wall and physically represents a combination of tube corrosion and slag buildup.

The heat transfer coefficient multiplier is called HTTIRH and is the 21st word on the 70YYYS card for component Type 28. It can be found in the MMI on the fourth page of the <u>(B) Component Data/(C) Heat Exchanger/Type 28 Convective Stage</u> menu. HTTIRH is input as a negative number. This indicates to PEPSE that it is a multiplier on the internally calculated overall heat transfer coefficient for a convective stage. The overall heat transfer coefficient includes the convective term between the flue gas and tube wall, the conduction through the wall, the in-line fouling resistance, and the convective term between the wall and working fluid inside the tube.

After finding the fouling and heat transfer coefficient multiplier values that produce a desired result at one load, the process is repeated for another load. Experience has shown that a pair of these values chosen to satisfy one load will not, in general, satisfy the heat transfer conditions of a second load.

A method that works in most instances, and which was used in this analysis, is to solve four problems. In the first case, a load was selected and each fouling resistance, RHTM, was set to 0.0. Controls were used to solve for the required heat transfer multiplier, HTTIRH, for each convective stage in the boiler. A second case at the same load but with a different fouling resistance (0.002 is recommended) yielded another HTTIRH for each stage. These two points defined a line for each stage, described by one equation with two unknowns.

This process was repeated at a second load to give a second equation in the same two unknowns. A compiled algorithm was used to determine the solution to this set of equations for each convective stage. Graphically, this method is demonstrated by simply plotting two lines and determining the intersection of the lines as shown in Figure 2.

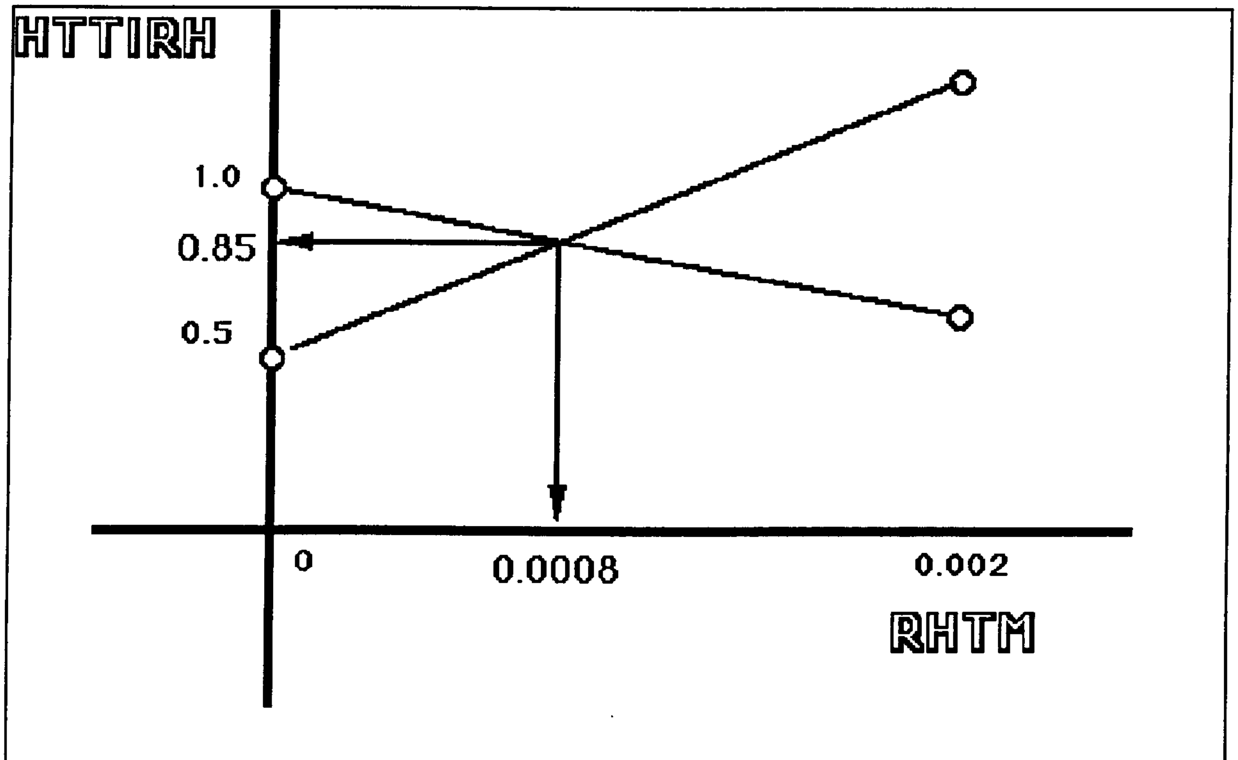

Figure 2. Graphical representation of compiled algorithm.

These calculations were performed for all stages with only one PEPSE run by executing four stacked cases. At the end of each case, all RHTM and HTTIRH values were saved in different operational variables. This is the only way the results of the first three cases could be made available to the compiled algorithm at the end of the fourth case.

At the end of the fourth case, a compiled algorithm was executed to find the intersection point, RHTM and HTTIRH, for each stage. These pairs of values were saved in operational variables. After all the tuning was finished, these values were transferred to the final boiler model. The final boiler model did not need the compiled algorithm to correctly calculate the convective stage heat transfer.

COMPILED ALGORITHM DESCRIPTION

The compiled algorithm was set up to accept four pairs of values from operational variables which describe two straight lines. The algorithm used straightforward algebra to determine the intersection of the two lines and return the solution pair in operational variables. This algorithm was called once for each stage in the boiler model. A listing of the algorithm is shown in Figure 3.

The 'IF' test is used to prevent a divide-by-zero error. If that situation were to occur, the calculations would be skipped, and no results would be returned for the X and Y values of the intersection. RM1 and RM2 are scratch variables that are the slopes of the two lines. In the unlikely condition that RM1 is equal to RM2, the calculation would fail.

IMPLEMENTATION OF THE COMPILED ALGORITHM

Implementing a compiled algorithm is different than implementing any of the other PEPSE features. A compiled algorithm is actually a stand-alone executable computer program that must be created by the user with tools provided in PEPSE Version 57. Chapter 18 in Volume I of the PEPSE Manual describes the process in detail.

```
INTERSECT
$
$ This algorithm calculates the intersection of two lines that
$ are defined by four points. The mapping from PEPSE variables
$ to compiled algorithm variables is as follows.
$
$        X1       X value for first point of first line
$        X2       Y value for first point of first line
$
$        X3       X value for second point of first line
$        X4       Y value for second point of first line
$
$        X5       X value for first point of second line
$        X6       Y value for first point of second line
$
$        X7       X value for second point of second line
$        X8       Y value for second point of second line
$
$ Results:
$        X9       X value of the intersection
$        X10      Y value of the intersection
$
IF
  ( (X3 .NE. X1) .AND. (X7 .NE. X5) .AND. (X4 .NE. X2) .AND.
     (X8 .NE. X6) ) THEN
        RM1 = ((X4 - X2) / (X3 - X1));
        RM2 = ((X8 - X6) / (X7 - X5));
        X9 =  (X6 - X2 + (RM1 * X1) - (RM2 * X5)) /
              (RM1 - RM2);
            X10 = RM1 * (X9 - X1) + X2
ENDIF
```

Figure 3. Listing of compiled algorithm called INTERSECT.

Briefly, a listing of the algorithm to be solved was typed into a text editor and saved as a file called ALGDEF.DAT in the \PEPSE\CA\ directory on a personal computer. This file was converted to FORTRAN source code by typing the command "ALGEN <CR>" from the DOS prompt. The executable file, COMALG.EXE, was created by typing the command "BUILD <CR>" from the DOS prompt. COMALG.EXE was copied to the directory from which PEPSE was executed. Since the MMI was used to execute PEPSE, this was the same directory that held the model files. (The same processes are available on mainframe versions of PEPSE, but the protocol may differ somewhat from that used on the PC.)

In addition to getting an executable file in the correct directory, implementation of a compiled algorithm requires the user to make data available to the compiled algorithm. This was done through the use of PEPSE's operational variables.

As stated earlier, the HTTIRH and RHTM values from each case were saved. They were copied into operational variables (OPVB) using the 88NNN0 cards or the <u>(E) Special Features/(E) Operations</u> menu in the MMI. Since the HTTIRH was used as a negative number in the calculations, it was multiplied by -1, the value of OPVB 1, before saving to an operational variable. OPVB 150 through OPVB 157 were used for the inputs from the four cases for the first stage. OPVB 158 and OPVB 159 were used for the results from the first stage. The second stage used the 160s, the third stage used the 170s, and so forth. The intent was to choose operational variables that would make this compiled algorithm portable from one boiler model to another. Figure 4 shows the operations used for the first case. Similar sets of operations were defined for cases two through four.

Additional compiled algorithm input includes the mapping of OPVBs to the Xn values in the compiled algorithm. This can be done on the 10NN0SS cards or in the <u>(E) Special Features/(K) Compiled Algorithm</u> menu of the MMI, and is shown in Figure 5.

During the execution of PEPSE, OPVB 150 gets set equal to X1, OPVB 151 gets set equal to X2, and so forth. Variables whose values get sent to the compiled algorithm and variables that received values calculated in the compiled algorithm are included on these cards. OPVB 159 is an example of a variable that receives an 'output' from the compiled algorithm calculation. It is set to the value of X10 since it is the tenth variable on the 10NN0SS cards.

```
* HTTIRH First Stage
881500  HTTIRH   90      MUL     OPVB     1     OPVB 150
* FOUL First Stage
881510  RHTM     90      EQL     OPVB    151
* HTTIRH Second Stage
881600  HTTIRH  110      MUL     OPVB     1     OPVB 160
* FOUL Second Stage
881610  RHTM    110      EQL     OPVB    161
* HTTIRH Third Stage
881700  HTTIRH  140      MUL     OPVB     1     OPVB 170
* FOUL Third Stage
881710  RHTM    140      EQL     OPVB    171
* HTTIRH Fourth Stage
881800  HTTIRH  150      MUL     OPVB     1     OPVB 180
* FOUL Fourth Stage
881810  RHTM    150      EQL     OPVB    181
* HTTIRH Fifth Stage
881900  HTTIRH  210      MUL     OPVB     1     OPVB 190
* FOUL Fifth Stage
881910  RHTM    210      EQL     OPVB    191
```

Figure 4. Example of Operation Definition cards for the CA.

```
*  Find resulting HTTIRH and RHTM for stage 1
1001000    0    0   INTERSECT
1001001 OPVB 150 OPVB 151 OPVB 152 OPVB 153 OPVB 154
1001002 OPVB 155 OPVB 156 OPVB 157 OPVB 158 OPVB 159
```

Figure 5. Example of CA Variable Name and Data Location Data Cards.

RESULTS

Good results were obtained for stages which had adequate geometry descriptions and intermediate temperatures. For example, Figure 6 shows the two lines that were calculated for the secondary superheater. Table 1 shows the stages, the fouling resistance values, and the heat transfer coefficient multipliers obtained from this method.

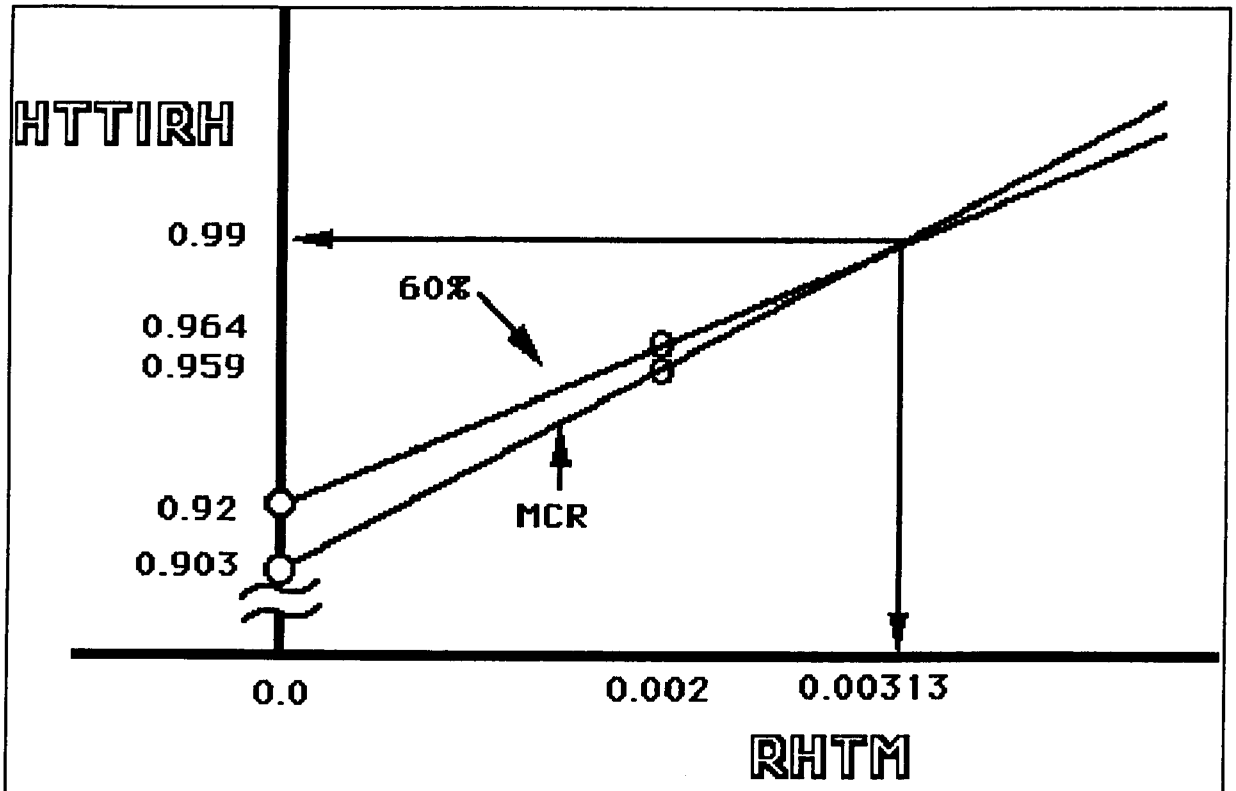

FIGURE 6. Results for the Secondary Superheater.

STAGE	HTTIRH	RHTM
Economizer	1.026	0.0025
Primary S.H.	Schedule	0.0020
Secondary S.H.	1.061	0.0063
Tertiary S.H.	1.665	0.0188
Reheater	0.898	-0.0086

Table 1. Boiler tuning results for the convective stages.

The HTTIRH and RHTM had to be scheduled for the primary superheater stage because the solution to the two equations produced a negative HTTIRH and RHTM. Reasonable values were produced by each of the four cases. However, the two lines were nearly parallel and the intersection was in the third quadrant of the X-Y plane. Some possible reasons for this result include: the primary superheater had nearly 50% of the convective stage area, it had baffle wall tubes running between the first and second stages, and it had no intermediate temperatures to use for tuning.

On a 486, 33 Mhz computer, the four stacked cases required approximately 30 minutes to complete execution.

CONCLUSIONS

The importance of matching the known boiler performance data (vendor performance sheet, test data, or other selected data) with the PEPSE model is an integral step in validating the calculations. A compiled algorithm, coupled with stacked cases as described in this paper allows the effect of boundary conditions, control system variables, and other parameters to be analyzed easily, quickly, and correctly.

The compiled algorithm is portable between boiler models, and can be used as a custom enhancement to the PEPSE code. Use of this method puts the load on the computer, not the engineer.