CHAPTER
FIVE

METHODS OF FINDING A SOLUTION

Some difficulties were experienced in finding a solving scheme that could handle the models presented in Chapters 3 and 4. The main difficulty lay in that some of the numerical schemes chosen would not converge at every time step. Therefore other solving means were attempted until a solving method could be found that solved at every time step. Three different solving methods were attempted in solving the NCHE model:

1) backsolving with TRNSYS 14's new solver,
2) an internal regula falsi method in conjunction with TRNSYS 14's new solver (RFS1),
3) and an internal regula falsi method in conjunction with TRNSYS 14's original solver, successive substitution (RFSS).

Sections 5.1, 5.2 and 5.3 describe in considerable detail the different solving methods employed and the problems encountered with these solving methods. These sections are written assuming the

reader is familiar with TRNSYS. A detailed understanding of the solution schemes used in modeling a NCHE loop are not required for the implementation of TYPE 67. Consequently it is not necessary for some readers to read Sections 5.1, 5.2 and 5.3. However as a significant amount of time was spent analyzing the solution schemes presented in this chapter, it was considered of value to report these findings. The simple model was run with all three solving schemes. Comparative results are presented in Section 5.4. Section 5.3 presents a description of the solving method used for the models presented.

5.1 Solution Using TRNSYS 14 Backsolver

TRNSYS 14 has two solvers, the original solver, which uses successive substitution, and the new solver, which uses Powell's method (Powell 1970).

5.1.1 TRNSYS 14 New Solver

The new solver has the capacity to solve not only for outputs, but for inputs as well. Backsolving refers to the case in which one or more outputs are specified and the corresponding inputs are determined. When backsolving, the new solver will be referred to as the backsolver.

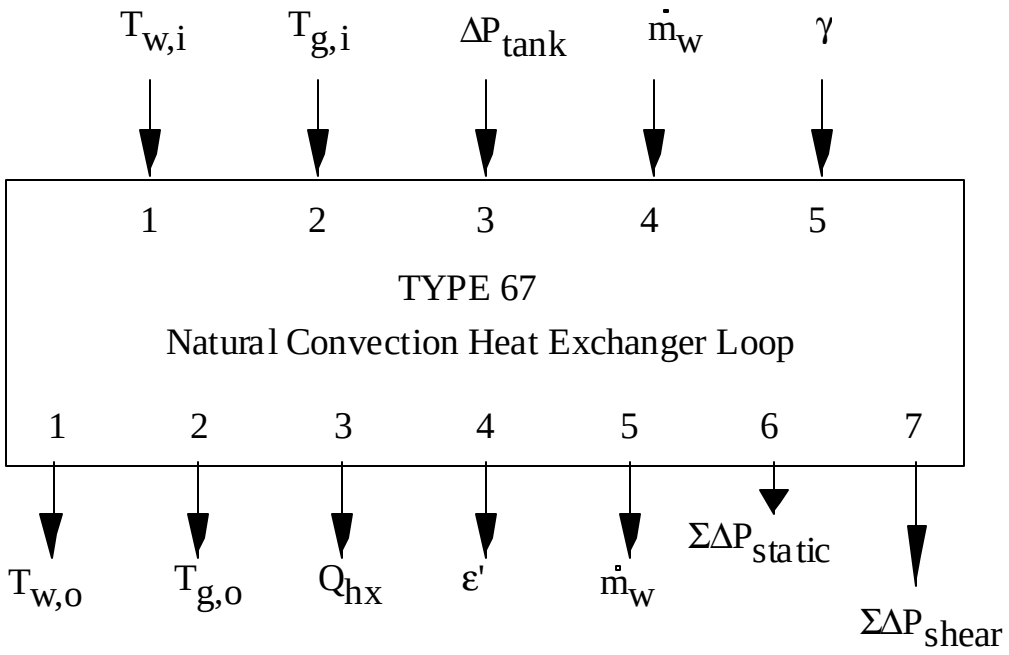

Figure 5.1.1 TRNSYS information diagram used for backsolving.

5.1.2 Backsolving Using TYPE 67

TRNSYS TYPE 67, shown in Figure 5.1.1, was written for backsolving with the new solver. Five inputs are required, including the heat exchanger water flow rate of the previous time step or iteration. The water flow rate is used in finding the outlet temperatures, which in turn are used to find the pressure drops and gains around the loop. The total static and shear pressure differences are output from the type, and are set equal in the deck in an equations card. TRNSYS then iterates until it finds the solution for the water flow rate, and the rest of the variables in that block.

5.1.3 Non-Backsolving Time Steps

The backsolver found solutions about 99% of the time. For some situations involving, for example, thin pipe diameters in the water loop or very high glycol flow rates, the backsolver had an especially difficult time, sometimes generating up to 500 error messages in 8670 time steps. TRNSYS indicates that a backwards solution is not possible when either the number of iterations surpasses the number on the LIMITS card, or when the solution diverges. When TRNSYS cannot backsolve, TRNSYS uses the user specified water flow rate guess value to forward solve. Using the simple model, non-backsolving timesteps (NBTSs) were for most situations kept to within 25.

By varying the user specified water flow rate guess value, the number of NBTSs change, and the times of the NBTSs change as well. The solution at each timestep usually is carried forward as the initial guess value for the next timestep. When there was no water flow the previous time step, one would expect TRNSYS to deliver 0.0 as the initial water flow rate guess value for the next timestep. However, the backsolver cannot carry 0.0 forward as the initial guess value, as this could create difficulties in Powell's numerical solving scheme. Instead, when the previous water flow rate solution is 0.0, TRNSYS will instead insert the user specified water flow rate guess value into TYPE 67 as the initial guess value for the next iteration. For this reason the user specified water flow rate guess value has much to do with which and how many time steps TRNSYS could not backsolve.

Fortunately the effect of the NBTSs is small as long as there are not too many of them. Water flow rate guess values were varied on an otherwise identical TRNSYS deck in order to generate different amounts of nonbacksolving time steps. Yearly runs were made, and the results, shown below in Figure 5.1.2, illustrate how small numbers of non-backsolving time steps have little effect on the solar fraction results.

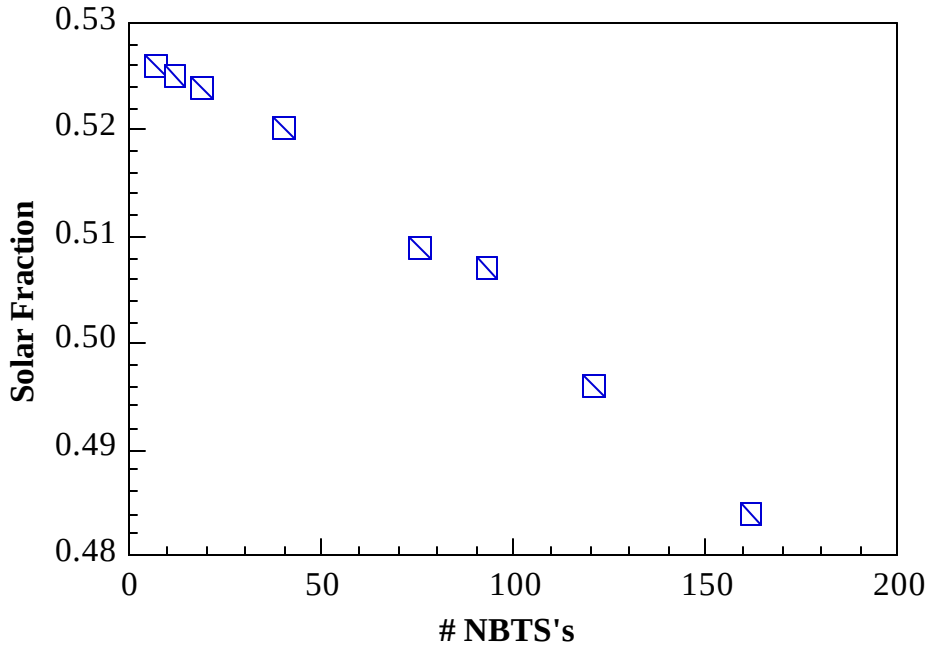

Figure 5.1.2 Effect of nonbacksolving time steps (NBTSs) on solar fraction solution.

Figure 5.1.2 shows that for 40 NBTSs or less the greatest percentage difference between any two solar fraction values was less than 1.2%, barely greater than the simulation energy balance percent difference of 0.96%. For 76 NBTSs or more the greatest percent difference was less than 3.4%, and for 93 warnings or less, the greatest percent difference was less than 3.9%. As these results apply to one set of parameters, one situation, the effect of NBTSs on results may be greater or less for other situations. Still, Figure 5.1.2 does give a good indication of the effect of NBTSs on the results.

Figures 5.1.3,4 show a typical NBTS. At time step 707, either the solution has diverged, or TRNSYS was unable to converge within 500 iterations. When TRNSYS is unable to backsolve, it forward solves with the user specified guess value for water flow rate. The guess value used in this simulation was 1 kg/hr, which translates to 0.00028 kg/s, hence the flow rate at time step 707 drops

to nearly zero. The effect of the NBTS on the tank water temperature is difficult to discern in Figure 5.1.4. Figures 5.1.5,6, however, give a clearer indication of the effect of one hour of a questionable water flow rate on the tank temperature.

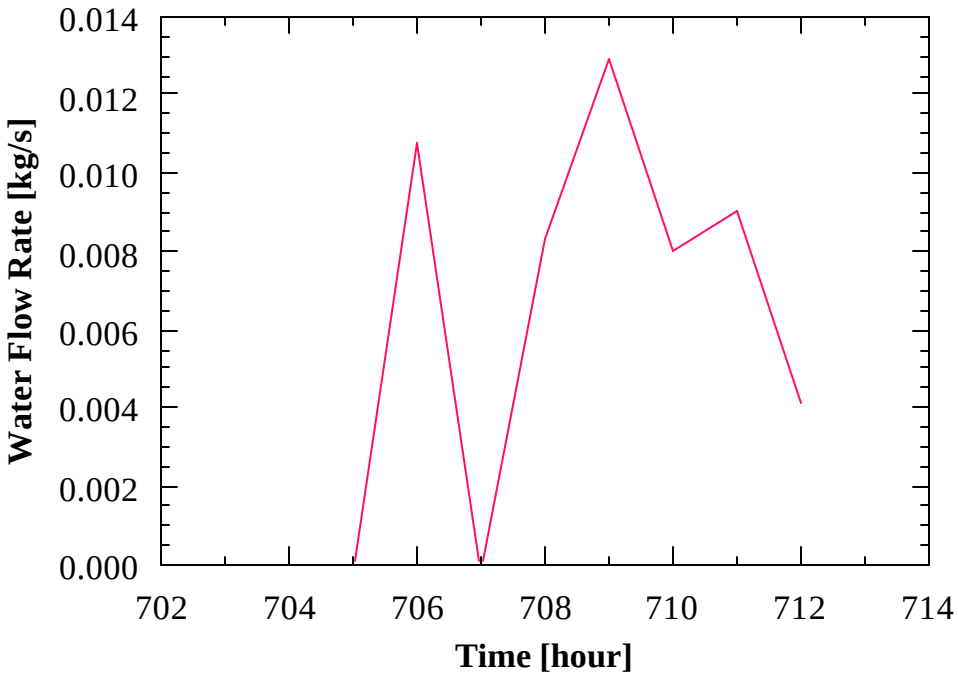

Figure 5.1.3 Example of NBTSs effect on water flow rate.

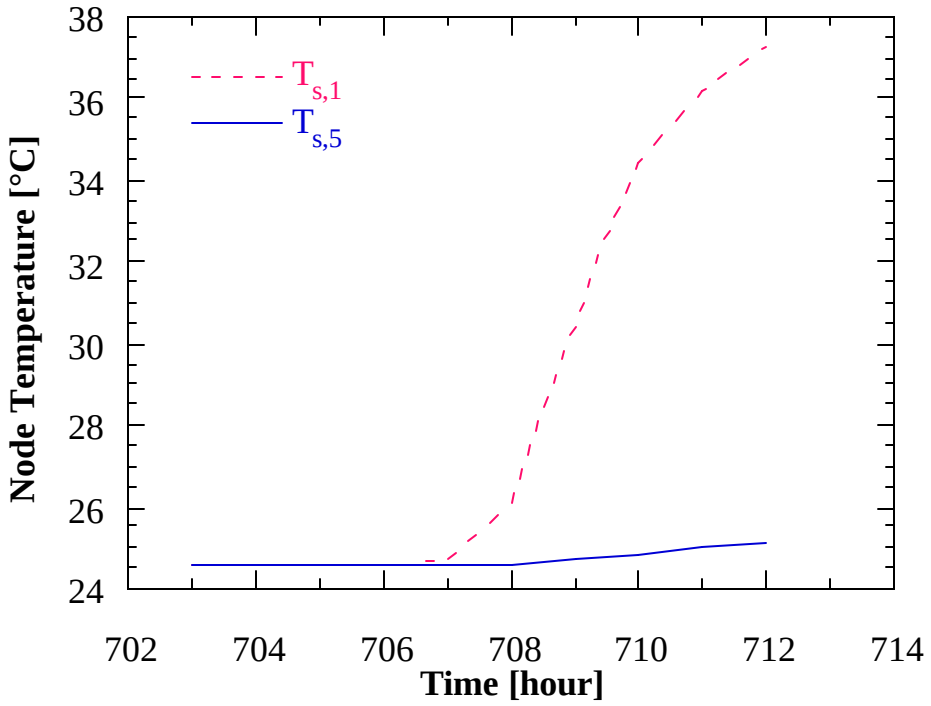

Figure 5.1.4 Example of NBTSs small effect on tank temperatures. $T_{s,1}$ and $T_{s,5}$ represent the top and bottom tank node temperatures for a 5 node tank.

5.1.4 The Effect of Incorrect Water Flow Rates on Tank Temperature

Incorrect water flow rates occur for two reasons, at NBTSs and when the water flow rate drastically increases for a timestep with the commencement of flow. In Figure 5.1.5 the water flow rate suddenly increases at time step 973. The water flow rate registers past 0.09 kg/s, however Figure 5.1.6 shows the effect of the water flow rate at this time step is small. The temperature of the top node, $T_{s,1}$, of the 5 node tank increases by 0.2°C, while the bottom node, $T_{s,1}$, registers no change at all.

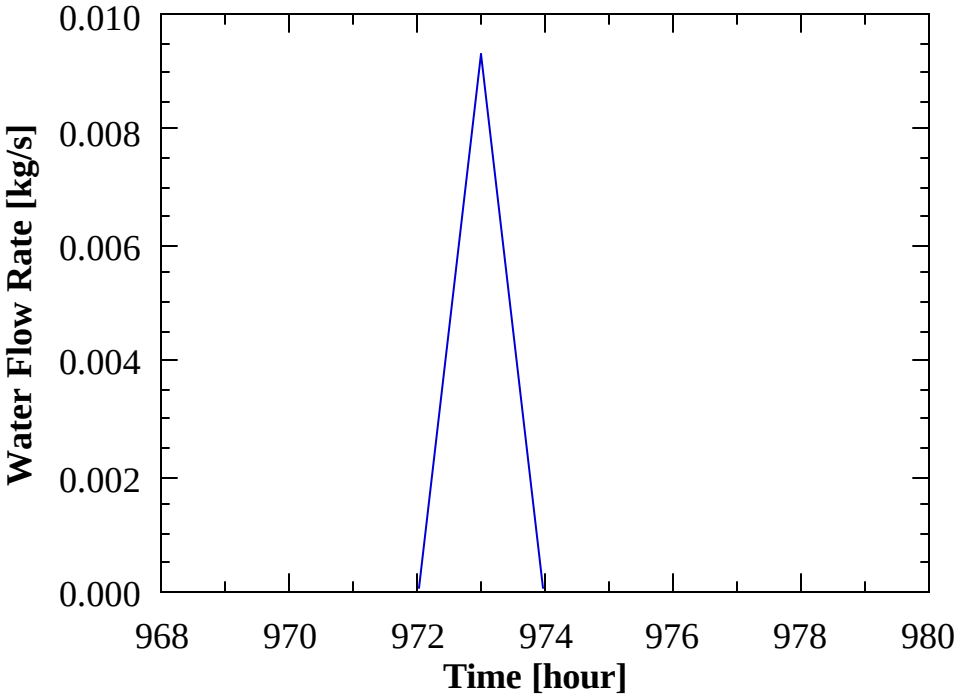

Figure 5.1.5 Example of suddenly increasing water flow rate.

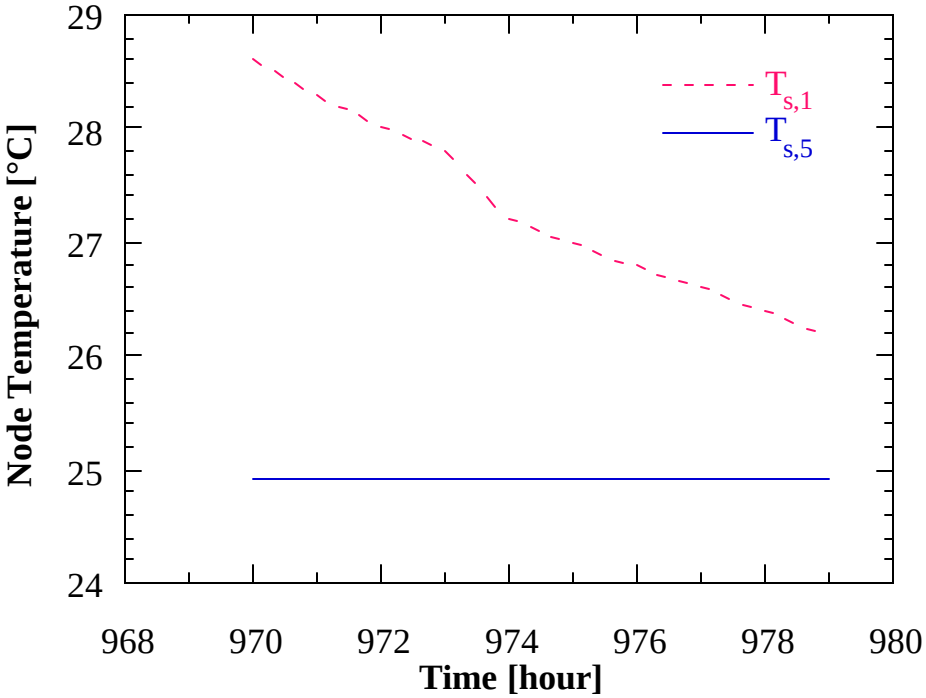

Figure 5.1.6 A sudden increase in water flow rate leads to small changes in tank temperatures.

5.1.5 The Effect of Simulation Time Step on Backsolving

Frequently for one hour intervals a NBTS occurs at the time step following the commencement of water flow. This happens when at the initial time step of water flow, there is a marked increase in the water flow rate from no flow to some high flow rate.

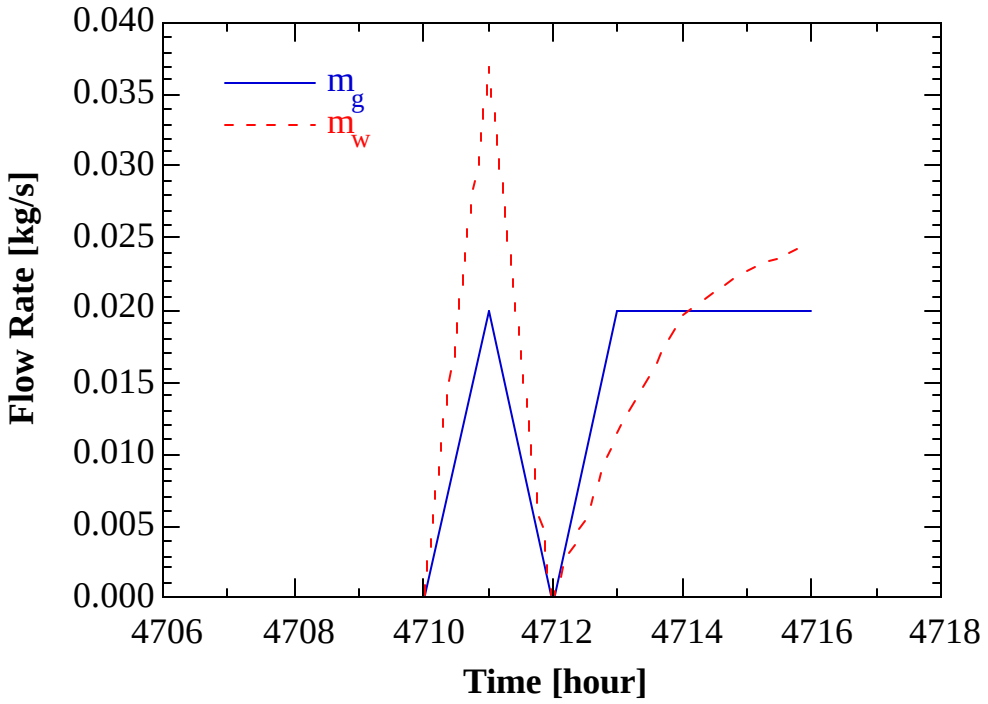

Figure 5.1.7 Sudden increases in water flow rate are often followed by NBTSs.

Figure 5.1.7 offers an illustration of this process. At time step 4711 the controller turns on (as indicated by m_g = 0.02 kg/s), and the water flow rate then suddenly increases to a high level, only to be followed by a NBTS at time step 4712. When TRNSYS cannot backsolve, TRNSYS will

forward solve using the user specified water flow rate guess value, which is this case was 1 kg/hr or nearly no flow. This accounts for the drop in water flow rate at 4712.

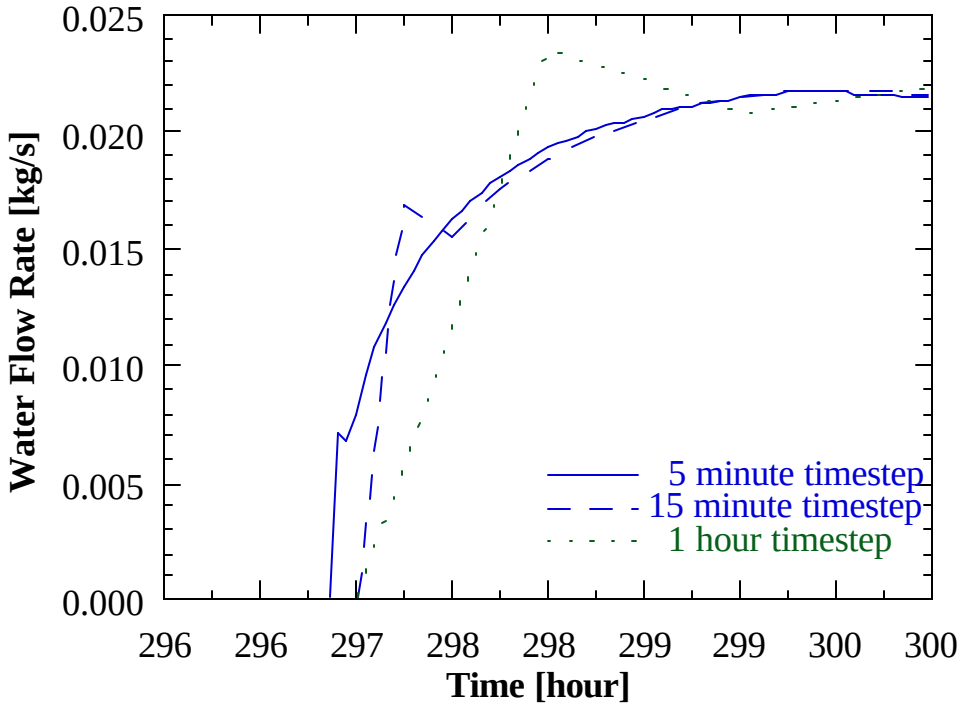

Figure 5.1.8 Water flow rate for three different time steps.

TRNSYS simulations are presented from the same deck using 3 different time steps, 1 hour, 15 minutes, and 3 minutes. At time step 298 Figure 5.1.8 shows a spike in water flow rate for the 1 hour interval curve. The solution for the water flow rate for 1 hour intervals at time step 298 is nowhere near the other solutions. All three time steps register a spiking behavior at the commencement of water flow. As the time step is reduced, the spike becomes more subdued, until it is barely obvious for the 3 minute time step case. Consequently, TRNSYS is able to recover easier from the inflated water flow rate solution and solve at the next time step. For one hour time steps it is believed that large spikes in the water flow rate cause the NBTSs that follow.

5.1.6 The Ultimate Effect of NBTSs

Even though the effect of individual NBTSs is small, there are conditions in which the number of NBTSs is large, and the accumulated effect upon the results is damaging. If TYPE 67 employed the backsolving method, the model would be limited in its applications, and its credibility would be questionable. Therefore it was necessary to find a new way to solve the model.

One possibility would be to amend the new solver's code. As different water flow rate guess values yielded non-backsolving time steps at different time steps, using a combination of two or three water flow rate guess values would virtually eliminate the problem. The pseudocode below illustrates how this could be done:

```
If too many iterations or if solutions diverge =>
        read new guess value
        remember last time step outputs
        begin iterating again
```

Reworking the code might involve saving and updating several variables in an array every solved time step. This too would have to be worked into the code. However reworking the backsolver code could prove messy and time-consuming. A safer and more timely route was to write an internal regula falsi solver.

5.2 Solution Using Internal Regula Falsi with TRNSYS's New Solver (RFS1)

An internal regula falsi solver was written into TYPE 67, so that given a set of inputs, the water flow rate could be found within the type. The outputs then would be sent to TRNSYS's new solver, which would then generate a new set of inputs for TYPE 67, and so on. The TRNSYS information flow diagram is shown in Figure 5.2.1.

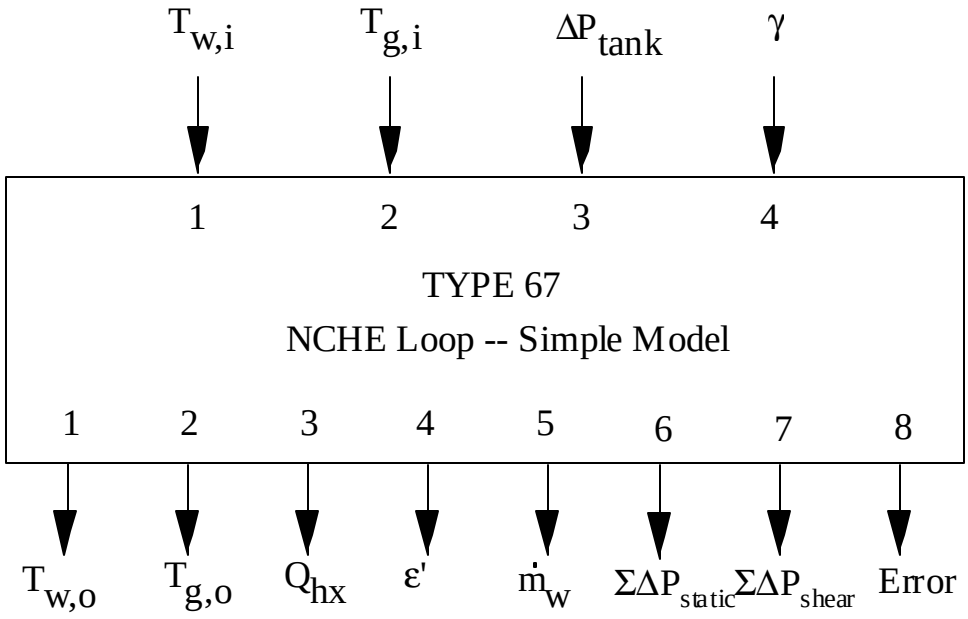

Figure 5.2.1 TRNSYS diagram for model with internal regula falsi solver and TRNSYS new solver. This same diagram also applies for the model that combines the internal regula falsi solver and successive substitution.

5.2.1 Solving with Regula Falsi

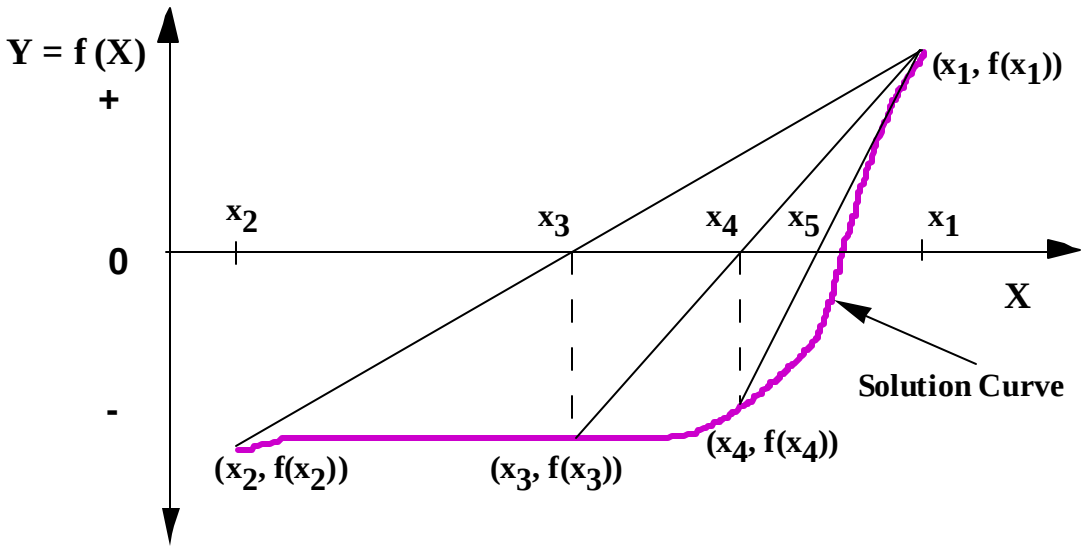

Figure 5.2.2 Solving with regula falsi.

Regula falsi is a simple root finding method. Consider the solution curve y=f(x) in Figure 5.2.2. The root of the curve, or solution, corresponds to the x and y coordinates where y=0. At other x values, y is nonzero and represents the error. The solution procedure is as follows:

1) Given a function, for example, y=f(x),

2) High and low limits are set for the independent variable at x_1 and x_2, such that the corresponding f(x) values are positive and negative.

3) These limits, x_1 and x_2, are inserted into the function, y=f(x), and the y values, $f(x_1)$ and $f(x_2)$, are found. In this way points $(x_1,f(x_1))$ and $(x_2,f(x_2))$ in Figure 5.2.2 have been defined.

4) A chord is drawn between points $(x_1,f(x_1))$ and $(x_2,f(x_2))$. The point where the chord crosses the axis, x_3, is obtained from the relation:

$$x_3 = \frac{x_1 f(x_2) - x_2 f(x_1)}{f(x_2) - f(x_1)} \qquad (5.2.1)$$

5) The x_3 value is inserted into the function, thereby establishing the point $(x_3, f(x_3))$.

6) If the point $(x_3, f(x_3))$ corresponds to a negative y value, then $(x_3, f(x_3))$ becomes the new low point, as it does in this case. However, if $(x_3, f(x_3))$ corresponded to a positive y value, then $(x_3, f(x_3))$ would become the new high value.

7) Again a chord is drawn between the new low and the high value, i.e. points $(x_3, f(x_3))$ and $(x_1, f(x_1))$.

8) x_4 is established at the intersection of the chord and the x axis using the equation:

$$x_4 = \frac{x_1 f(x_3) - x_3 f(x_1)}{f(x_3) - f(x_1)} \qquad (5.2.2)$$

9) x_4 is then inserted into the function, and the corresponding y value, $f(x_4)$ is then found.

10) The process continues until the y value found is close enough to the x axis so that it lies within a specified tolerance.

In TYPE 67, x corresponds to the water flow rate and y corresponds to the error, which is defined as the difference between ΔP_{shear} and ΔP_{static}. The equation, y = f (x), corresponds to the series of equations that make up the NCHE model.

5.2.2 Results Using the RFS1 Method

Of the three solving methods chosen, the internal regula falsi method in conjunction with the new solver is by far the slowest and worst of the three methods. Although the new solver is forward solving, not backsolving, there were many time steps when TRNSYS could not find a solution, many more than in using the backsolving method. Results using the RFS1 method nearly matched those of the backsolver and RFSS methods at 15 minute time steps, except when no solution was possible (see Section 5.4).

When using the new solver, a number of equations are solved simultaneously. One of these equations, that which corresponds to the water flow rate, is solved independently in TYPE 67. This equation is inextricably linked to the others, and should be in the same solving block, therefore solving at the same time. As the equation corresponding to the water flow rate is solved independently, this may render Powell's method, used in TRNSYS's new solver scheme, incapable of solving the system of equations. Due to the preponderance of non-solving time steps, another solution method was tried.

5.3 Solution Using Internal Regula Falsi with TRNSYS's Successive Substitution (RFSS)

5.3.1 Successive Substitution

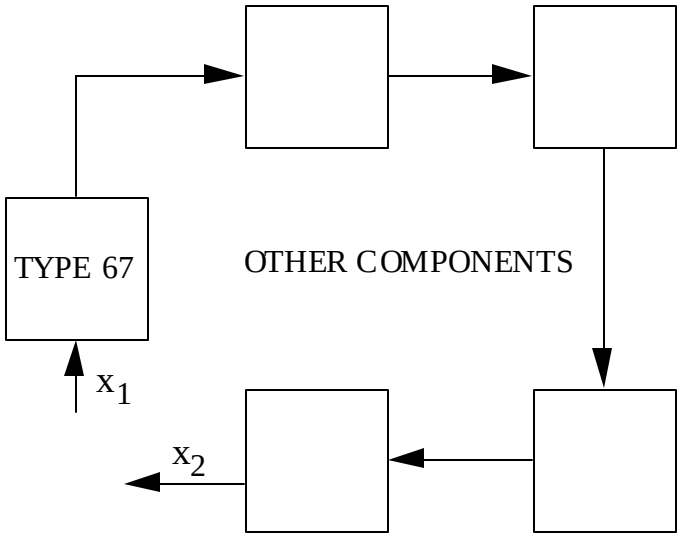

Figure 5.3.1 Successive substitution method block diagram: when $|x_2 - x_1| <$ tolerance, the equations at a time step are considered solved.

TRNSYS's original solver, solver 0, uses the successive substitution method. A block diagram of successive substitution method is presented in Figure 5.3.1. Given a set of inputs into TYPE 67, x_1, TYPE 67 will produce a set of outputs, which will be used by other types in the deck in a specified calling order. After TRNSYS has processed the other components, and is ready to call TYPE 67 again, the outputs from the other components, x_2, are compared to the original inputs to TYPE 67,

x_1. If the difference between x_1 and x_2 is less than the given tolerance, and similarly described differences between "x_1"and "x_2" for all other components in the deck are less than tolerance, then the equations are considered solved at that time step.

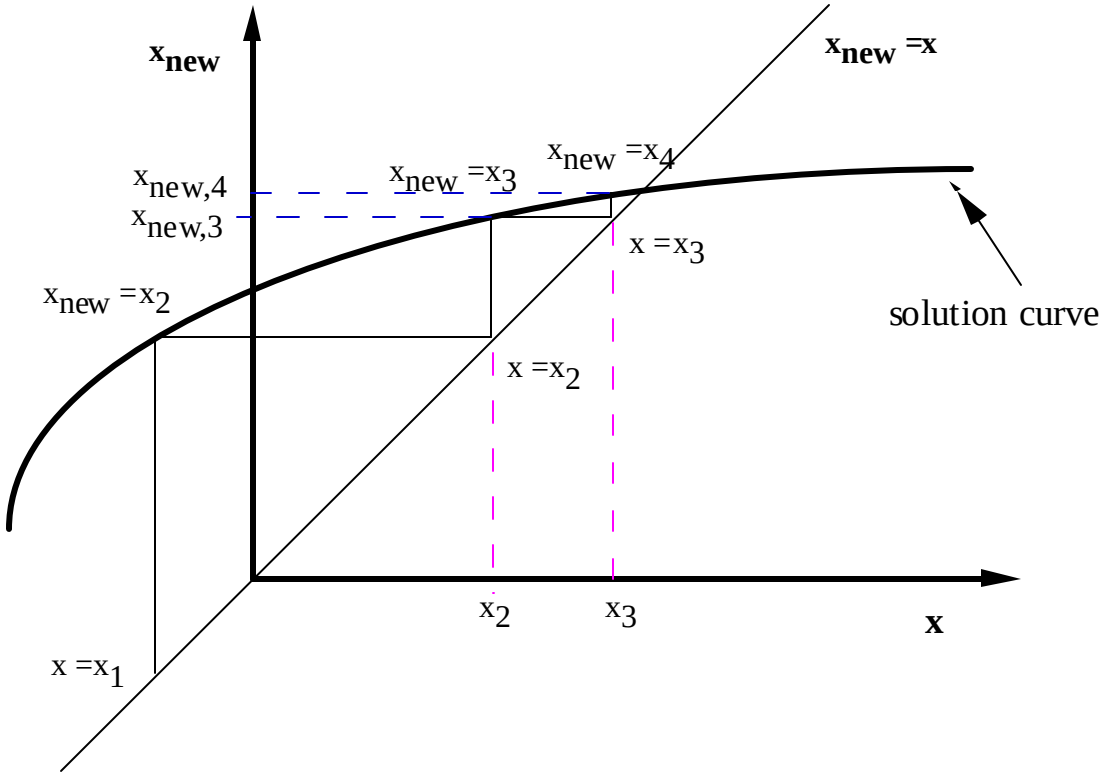

Figure 5.3.2 Convergence using the successive substitution method.

In order to arrive at a solution, the successive substitution method undergoes the following solving procedure as shown in Figure 5.3.2 :

1) Given a set of inputs to TYPE 67, x_1;

2) By processing the other components in the deck, TRNSYS will produce outputs on the solution curve, x_2, which are compared to the original inputs, x_1.

3) If $|x_2 - x_1| <$ tolerance, the timestep would be considered solved,but as the difference between the two does not lie within tolerance,

4) TRNSYS uses the outputs, x_2, as inputs to TYPE 67 and to the rest of the components.

5) These in turn produce outputs on the solution curve, x_3, which are again compared to x_2 for tolerance.

6) The process continues until $|x_{i+1} - x_i| <$ tolerance.

Successive substition method will converge as long as the following condition is met in the interval near the solution:

$$|g'(x)| < 1 \qquad (5.3.1)$$

where $g'(x)$ is the slope of the solution curve (Jaluria 1988). Successive substitution converges as long as the solution curve is at an obtuse angle to the 45° line corresponding to $x=x_{new}$. If the solution curve is perpendicular or at an acute angle with the 45° line, the solution method will either cycle or diverge.

5.3.2 Problems with the RFSS: Cycling Behavior

The RFSS method works well, however, given a set of parameters, such as was found in time step 2538, no solution could be found. As was previously mentioned, successive substitution can solve as long as the solution curve used is at an obtuse angle to the 45° line. However, at time step 2538, the curve was not at an obtuse angle to the 45° line. Rather, a chord taken between two equidistant points close to the solution was perpendicular to the 45° line, as is shown in Figure 5.3.3.

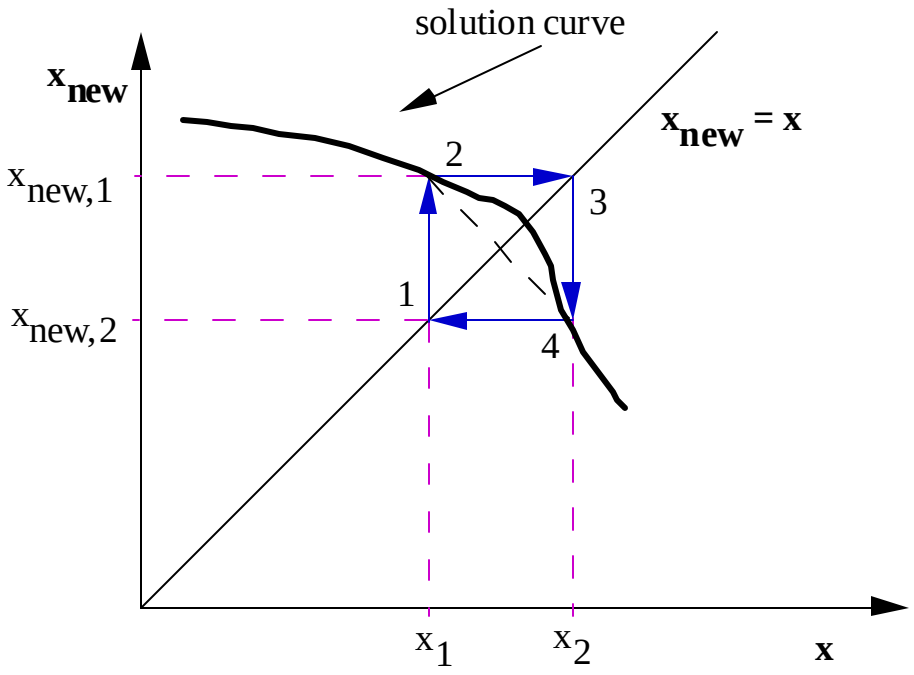

Figure 5.3.3 Cycling behavior can occur when a chord taken at two equidistant points of the solution curve near the solution is perpendicular to the $x = x_{new}$ line. The dashed line between points 2 and 4 is perpendicular to the solution curve.

As a result the outputs to TYPE 67 and to the rest of the deck oscillated between two sets of values as is shown in Figure 5.3.4.

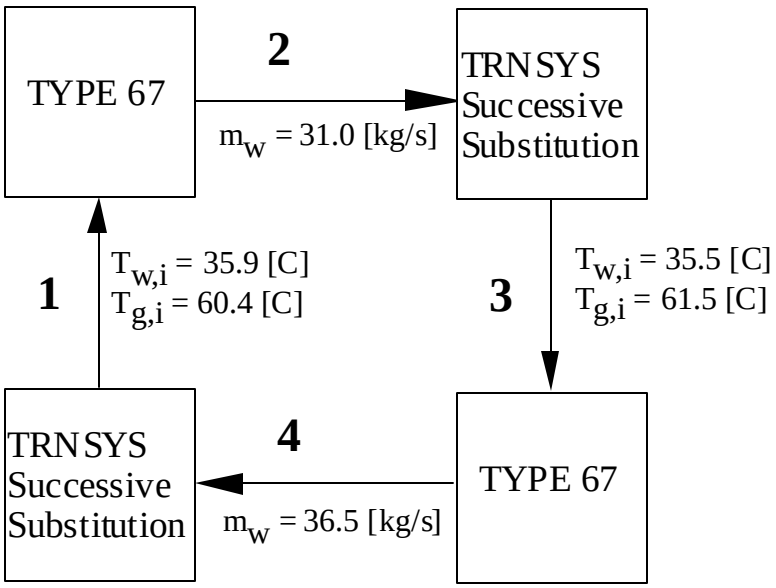

Figure 5.3.4 Problems with successive substitution: example of cycling behavior.

The cause of the cycling is due to the close relationship between the glycol inlet temperature and the water flow rate. In order to resolve this cycling problem, TYPE 44, a convergence promoter, was added to the TRNSYS deck. The external convergence promoter is designed to recognize cycling, and then replace TRNSYS's successive substitution method with the secant method to find next iteration values. A schematic of the secant solution method is provided in Figure 5.3.5. The initial values of the secant method, x_1 and $f(x_1)$, are found using the successive substitution method. After a user specified number of iterations, the convergence promoter employs the secant method. The addition of TYPE 44 to the deck removes the cycling problem.

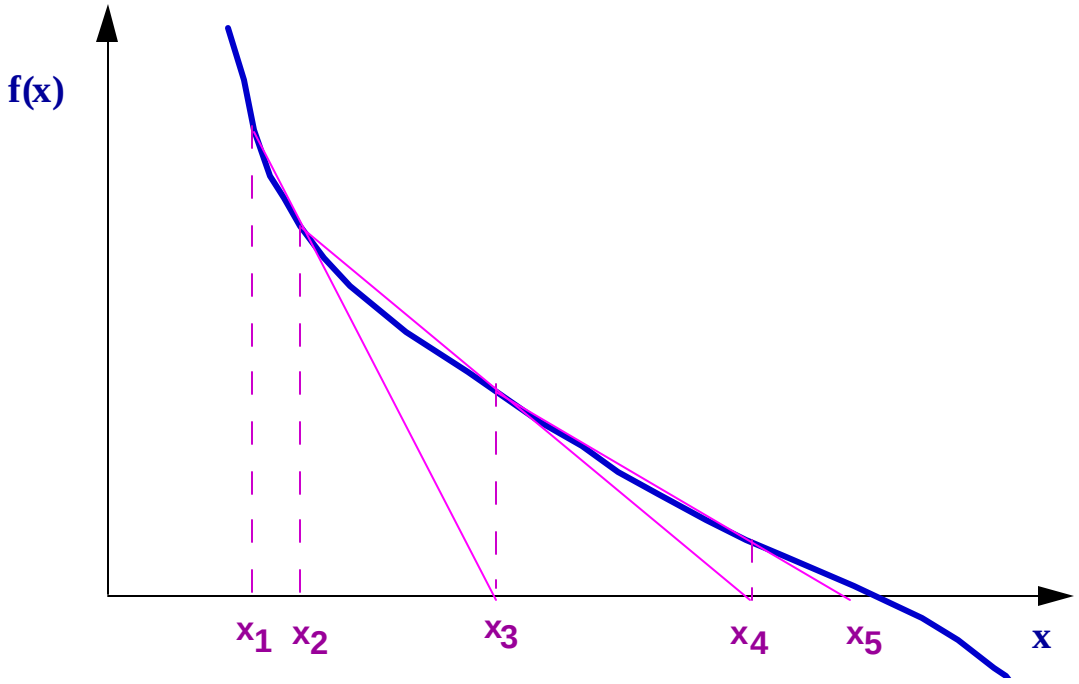

Figure 5.3.5 Example secant method solution using TYPE 44, the external convergence promoter.

5.3.3 The Effect of Tolerances on Simulation Performance

TRNSYS simulations allow the user to select error tolerances for the simulation. At a particular time step, TRNSYS will continue calculating until the solution falls within the specified tolerances. The choice of tolerances greatly affects the accuracy and computational effort required for a given simulation. Although low tolerances lead to more accurate solutions, low tolerances also require increased simulation time.

However, looser tolerances lead to more inaccurate results. Table 5.3.1 shows the effect of tolerances on simulation accuracy. Choosing a tolerance of 0.01 rather than 0.0001 cuts simulation time in half, yet reduces the accuracy of the simulation by 2.13 %. Choosing a tolerance of 0.001 reduces simulation accuracy by 0.14%. Simulations run to generate Table 5.3.1 employed 15 minute time steps. The tolerance value in the first column refer to the magnitude of the absolute integration error tolerance and the relative error tolerance controlling the convergence of input and output variables.

Table 5.3.1 The Effect of Tolerances on Simulation Accuracy and Simulation Time

Tolerances	Solar Fraction	Simulation Time [min.]
0.01	0.6199	27
0.001	0.6343	39
0.0001	0.6334	55

A tolerance of 0.001 was chosen for optimization runs described in Chapter 6.

5.4 Comparison of Solving Methods

The same simulation was performed using each of the three solving methods discussed. The RFSS method and TRNSYS's backsolving (BACK) method gave similar results as is shown in Figures 5.4.1-3. Twice in the 15 hours shown, the RFS1 method was unable to find a solution. Consequently the RFS1 curve does not match the others. It is interesting to note that the backsolver predicts a large water flow rate at the commencement of water flow. At this first timestep of water flow, the only significant difference between the RFSS and backsolving methods occurs for the first 15 hours of the year. However as will be shown, these small variations between the solving methods accumulate with time into large discrepancies in tank temperatures.

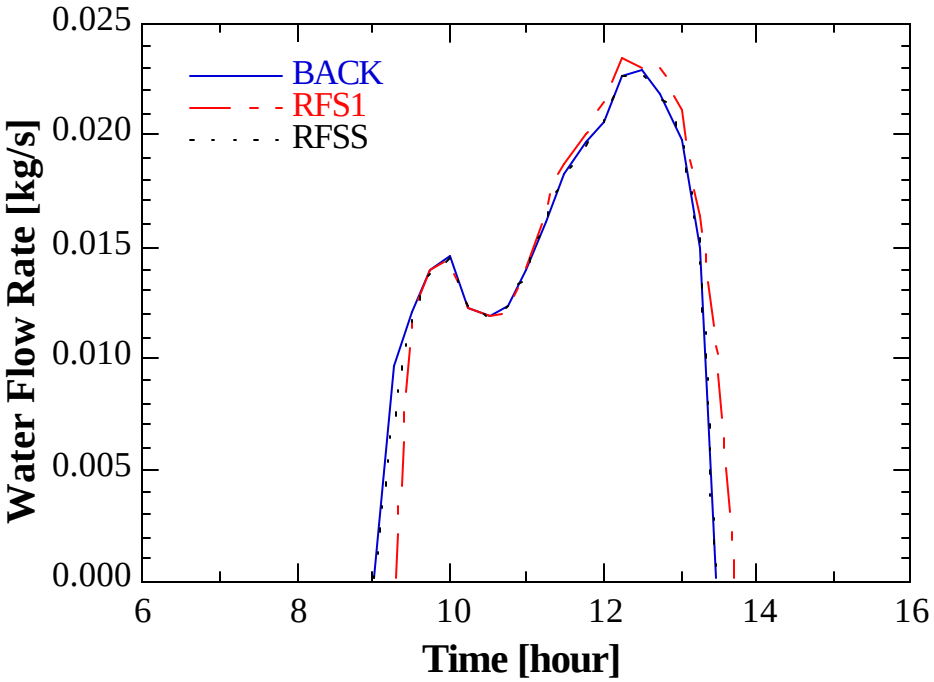

Figure 5.4.1 Comparison of RFSS and backsolving (BACK) methods for generating water flow rate. The RFSS and backsolver methods give nearly the same results.

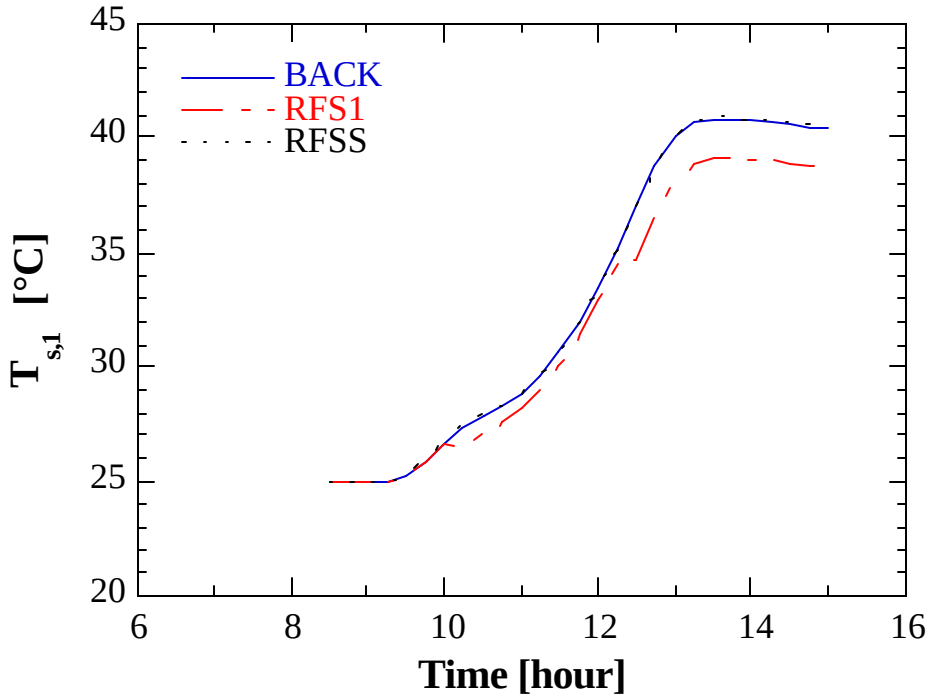

Figure 5.4.2 Comparison of the 3 solving methods for generating tank top node temperature, $T_{s,1}$. The RFSS and the backsolving methods yield approximately the same tank temperatures on the first day of simulation

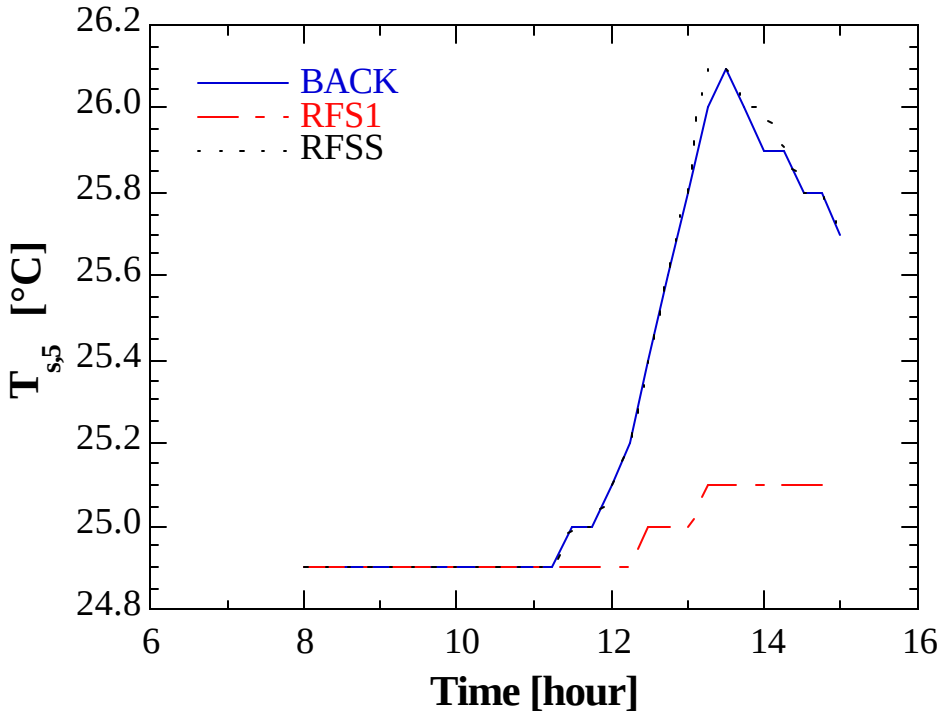

Figure 5.4.3 Comparison of the 3 solving methods for generating tank bottom node temperature, $T_{s,5}$.

In Figures 5.4.4-6 comparisons between the RFSS and backsolving methods are also shown at the time step 2538 where the RFSS had such difficulty. Although the water flow rates are nearly the same for most of the time interval presented in Figure 5.4.4; in Figures 5.4.5,6 the two solving methods show tank top and bottom node temperatures differing by 4°C. This 4°C disparity can be attributed to an accumulation of small variations between the two simulations over the prior 2532 time steps.

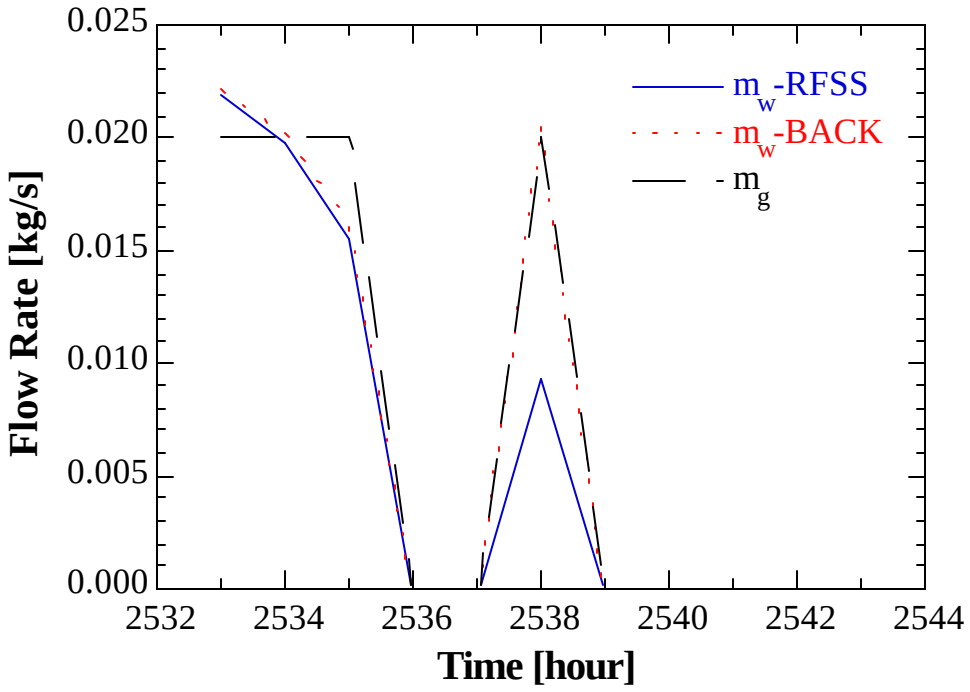

Figure 5.4.4 Comparison of RFSS and backsolving methods for generating water flow rate. Although time step 2538 is the time step that the RFSS had such a difficult time solving for, the RFSS solution appears to be more believable than the BACK solution.

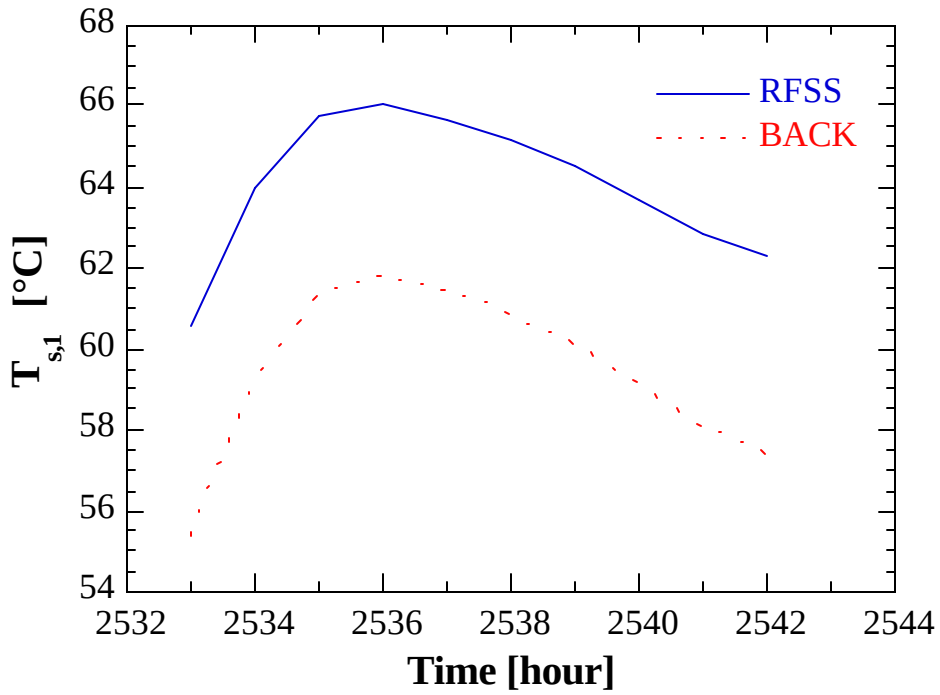

Figure 5.4.5 Comparison of RFSS and backsolving methods for generating tank top node temperature. The roughly 4° C temperature difference for the two methods can be attributed to variations between the two methods over time.

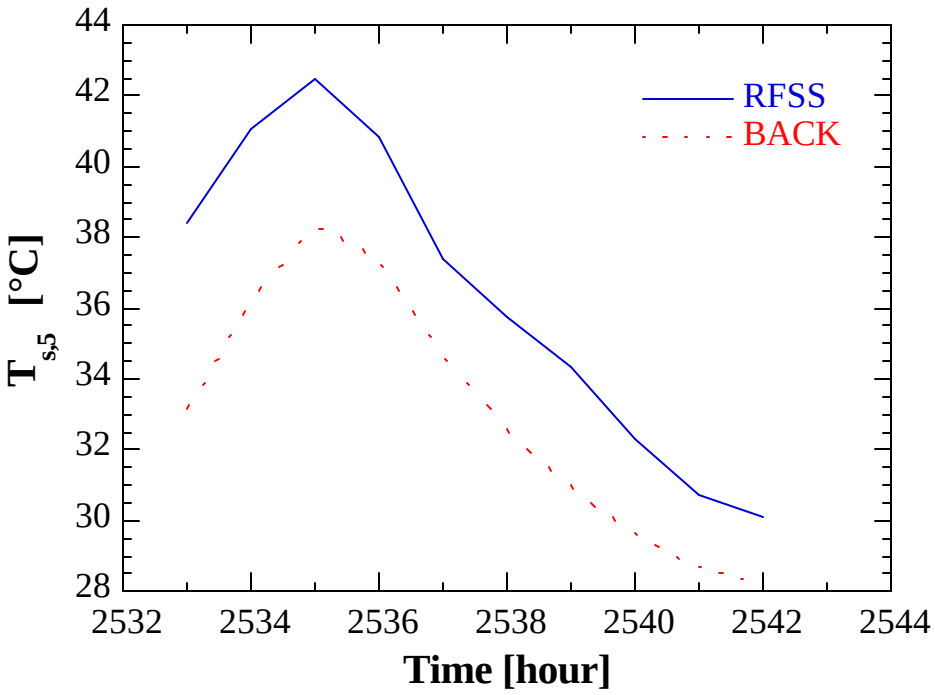

Figure 5.4.6 Comparison of RFSS and backsolving methods for tank bottom node temperature.

5.5 The Possibility of Multiple Solutions

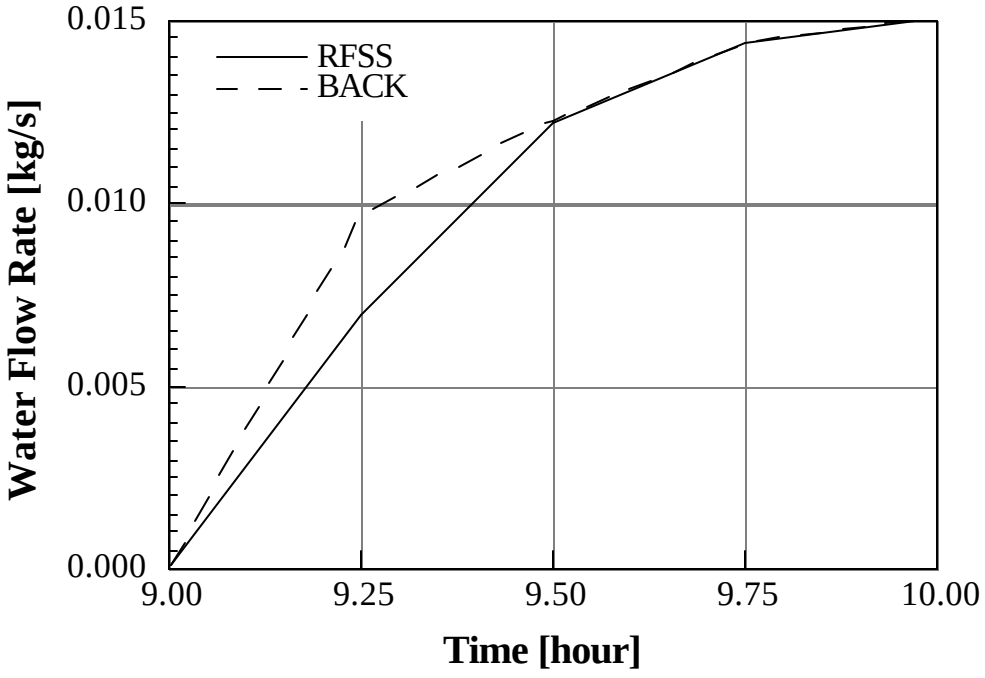

Figure 5.5.1 Discrepancy in water flow rates for RFSS and BACK methods.

In Figure 5.4.1 the backsolver and RFSS methods each predicted different water flow rates at timestep 9.25. Figure 5.5.1 presents the same data, but on a smaller scale, thereby magnifying the differences found at timestep 9.25. The backsolver reported a water flow rate of 0.00972 kg/s while the RFSS method reported a water flow rate of 0.00696 kg/s. It was considered that there might be more than one solution for the simulation at 9.25 hours. A combination of variables, which include water flow rate, heat exchanger inlet and outlet temperatures and tank temperature, together form the solution for a timestep. As both the RFSS and the backsolver had found solutions, it could be assumed then that both solutions were accurate, and that multiple solutions might exist for that timestep. A program was written that modeled the heat exchanger, the piping and the solar collector. Constant solar radiation, water inlet temperature and tank head at 9.25 hours (which were common to both solutions) were inserted as constants, and a range of water flow rates was inserted, so that an error could be generated. The error was of the form:

$$Error = \oint_{loop} dP_{shear} - \oint_{loop} dP_{static} \qquad (5.5.1)$$

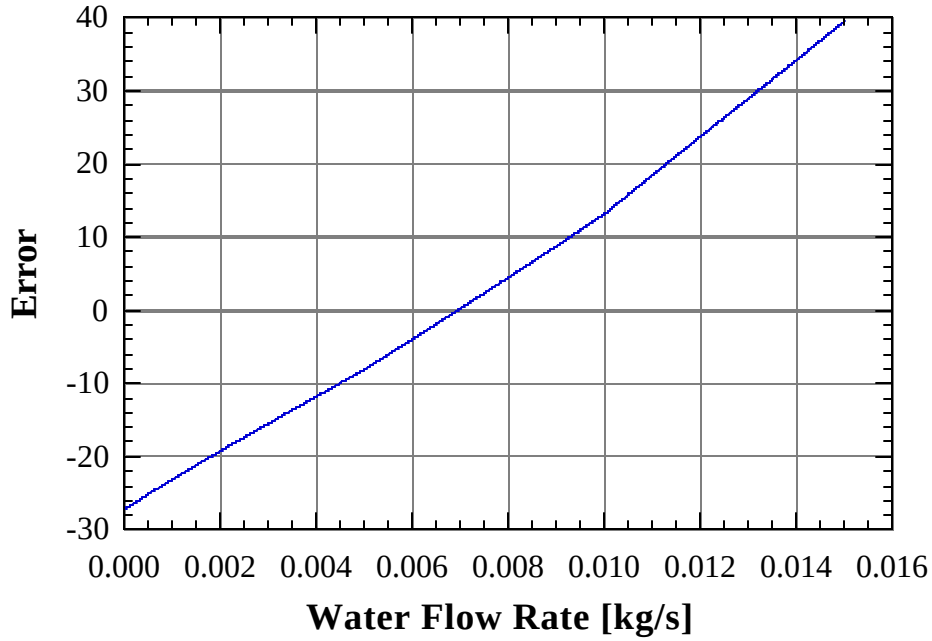

Figure 5.5.2 Error curve for constant inlets at time step 9.25.

In Figure 5.5.2, a plot of error as a function of water flow rate the error curve crosses the horizontal axis at the water flow rate solution. If there were multiple solutions the error curve would cross the water flow rate axis at each solution, that is, more than once. Yet the error curve was nearly linear. The curve crossed the horizontal axis at the water flow rate value found using the RFSS method, 0.007 kg/s, which indicates that the solution found using the RFSS method was correct, that the solution using the backsolving method was incorrect, and most importantly that the difference in the two solving methods shown in Figure 5.4.1 is due not to the presence of multiple solutions, but to problems with the backsolver. However this investigation does not preclude the possibility of multiple solutions at this and other timesteps; instead it only invalidates the suggestion that multiple solutions may be the cause of the two distinct solutions found in Figure 5.5.1.

5.6 Conclusions

The validity of the RFSS method has been established. The RFSS method not only can approximately match the backsolver's solution, it can find a solution where the backsolver cannot. When the two solvers give different solutions at the commencement of water flow, the RFSS method is presumed to give the more accurate solution. There are no multiple solutions. As the RFSS method is superior is all aspects, the RFSS solving scheme will be used in TYPE 67.