Application of functional optimization techniques for the serial hybrid computer solution of partial differential equations*

by HIROSHI H. HARA
Lockheed-California Co.
Burbank, California

and
WALTER J. KARPLUS
University of California at Los Angeles
Los Angeles, California

INTRODUCTION

Since its introduction in 1961, the serial or CSDT (continuous-space-discrete-time) method for solving nonlinear parabolic partial differential equations in one space-dimension has received a considerable amount of attention. This effort has been justified by the engineering importance and the abundance of problems characterized by one-dimensional diffusion equations. Although initially introduced as a pure analog technique, interest in the CSDT method was stimulated particularly by the increasing availability and capabilities of hybrid computing systems. Utilizing the hybrid computer, a closed loop of analog elements is employed to integrate a second-order ordinary differential equation at successive time levels. The digital computer serves as a function memory and to control the iterative determination of an initial condition at each time level.

The major advantages of the CSDT approach over the more conventional DSCT (discrete-space-continuous-time) approach include the following:

1) The analog hardware requirements of the CSDT method are very small compared to those of the DSCT method. This consideration is particularly important in the treatment of equations with nonlinear and time-varying parameters, for in this case a DSCT simulation requires separate nonlinear function generators at each finite-difference grid point in the space domain.

2) The high-speed computational capabilities of modern iterative analog computers can be utilized to full advantage.

3) Problems involving moving boundaries can be solved readily by controlling the analog computer’s integration interval, since the problem space variable is represented by the computer time variable.

In practical applications, however, considerable difficulty is encountered in obtaining dependable results using the CSDT method. The reason for this lies in the fact that in addition to the errors normally encountered in hybrid simulations (errors due to truncation and analog inaccuracies), the mechanization of the CSDT technique requires the instrumentation on the analog computer of an inherently-unstable ordinary differential equation. This in turn causes errors, which normally can be tolerated in hybrid work, to accumulate and grow excessively as the solution progresses, leading to highly unreliable results.

It is the purpose of the present paper to introduce a method which retains the advantageous features of the serial CSDT method but which involves the solution on the analog computer of a stable rather than an unstable differential equation. In the following section the

*Research in hybrid computation in the Department of Engineering, University of California, Los Angeles is supported by the National Science Foundation under grant GK-1738.
classical CSDT method is briefly reviewed in order to provide a point of departure for the description of the new method. The third section is devoted to a general description of the new technique, while the computational algorithms handled on the digital computer are presented in the fourth section. The last section includes general comments on the new method as well as a description of the extension of the method to parabolic differential equations in two space-dimensions.

The classical CSDT method

We consider a linear one-dimensional diffusion equation

\[ \frac{\partial^2 u(x, t)}{\partial x^2} = \frac{1}{\alpha} \frac{\partial u(x, t)}{\partial t} \]  

(2.1)

where

\[ x \in (0, x') \quad \text{and} \quad t \in (0, t') \]

and the initial condition

\[ u(x, t = 0) = u_0(x) . \]  

(2.4)

Applying the classical CSDT approximation to equation (2.1),

\[ \frac{\partial u^i}{\partial x} \approx \frac{u^i - u^{i-1}}{\Delta x} , \quad i = 1, \ldots, N \]  

(2.5)

where

\[ u^i = u(x, t^i) \]

\[ t^i = i\Delta t \]

\[ \Delta t = \frac{t'}{N} \]

Equations (2.2) and (2.3) are, respectively, transformed into

\[ u^i(x = 0) = u_0^i = u_0(t^i) \]  

(2.6)

and

\[ u^i(x = x') = u^i_r = u_r(t^i) . \]  

(2.7)

while equation (2.4) is transformed into

\[ u^{i-0} = u_0^i(x) . \]  

(2.8)

Since equation (2.5) is now an ordinary differential equation, an analog computer can be utilized for its solution. That is, the independent variable x can be represented by the analog computer-time variable and therefore integrations can be carried out in a continuous manner. In solving for \( u^i \), the solution \( u^{i-1} \) of the previous step in time acts as a driving function. The generated function \( u^i \) then must be "remembered" so as to be used as the driving function for the next time step.

Furthermore, we now need to solve a two-point boundary value problem because the boundary condition (2.3) in the original problem has been converted to the final value of \( u^i \). That is, while the boundary condition (2.2) is used as one of the initial conditions (i.e., \( u^i(x = 0) \)), the other initial condition, \( \frac{du^i}{dx}(x = 0) \), which is unknown must be found such that (2.3) is satisfied at the end of the computer run. This is usually accomplished by an iterative technique.

A typical hybrid computer mechanization for the implementation of classical CSDT method is shown in Figure 1. Briefly, the computational steps are:

1) Initialize the analog integrators with one known initial condition \( u_0^i \) and a trial initial condition \( \frac{du^i}{dx}(x = 0) \).

2) Place the analog subsection in the COMPUTE mode during which the solution \( u^{i-1} \) obtained during the preceding solution cycle is played back via a DAL (digital-to-analog linkage), and the new solution \( u^i \) is recorded via an ADL (analog-to-

![FIGURE 1—Hybrid computer mechanization of the classical CSDT method](From the collection of the Computer History Museum (www.computerhistory.org))
digital linkage). The ADL usually includes a sample-hold amplifier which is connected to an analog-to-digital converter through a multiplexor switch. The DAL can include a hybrid interpolator which transforms discrete staircase functions appearing at digital-to-analog converters into smooth functions.

3) At the end of the run corresponding to \( x = x' \), compare the final value of \( u_i \) to the known boundary value \( u_{i+1} \). If the difference is within a specified error bound, increment \( i \) and go back to step 1. If not, improve the trial initial condition and repeat the above process.

As can be seen from Figure 1, the analog computer circuit contains a closed loop comprised of four operational amplifiers. Such a circuit is inherently unstable and can be expected to exhibit marked sensitivities to errors in initial conditions and component inaccuracies. The loop gain is seen to be inversely proportional to \( \Delta t \); hence the smaller \( \Delta t \) the more pronounced the instability of the analog loop. It is therefore not feasible with a mechanization of this type to reduce the truncation error inherent in the finite difference approximation of \( \frac{\partial u}{\partial t} \) by reducing \( \Delta t \). The loop-gain is also affected by the time scale factor employed in the analog loop, so that the faster the solution speed the larger the instability. It should be noted that the reversal of the computing direction (i.e., backward computation) still keeps the unstable loop intact. These considerations have severely limited the applicability of the serial CSDT method.

**A new CSDT method**

It is the objective of the present method to retain the advantageous features of the serial CSDT method while obviating the need for an even number of operational amplifiers in the analog loop. The method is applicable to nonlinear as well as to linear partial differential equations. For simplicity in exposition the discussion below makes reference to a linear parabolic partial differential equation in one space-dimension. It should be recognized, however, that most practical applications will involve equations with nonlinear and time-varying parameters. The key step in the derivation of this method involves the definition of a control function, which is generated digitally and imposed as a forcing function upon the analog circuit. Denoting this control function as \( \hat{u}^i \) and introducing this term in equation (2.5) results in

\[
\frac{d^2 u^i}{dx^2} = \frac{1}{\alpha \Delta t} (2\hat{u}^i - u^i - u^{i-1}) \quad (3.1)
\]

If \( \hat{u}^i(x) \) can be found such that

\[
\hat{u}^i(x) = u^i(x) \quad \text{for all} \quad x \in (0, x') \quad (3.2)
\]
equation (3.1) reduces to equation (2.5). Note that \( u^i \) is the only function for which to solve—with \( \hat{u}^i \) appearing as external driving functions. Of course, we must find or "optimize" one of the external functions, namely the control \( \hat{u}^i(x) \) such that the equality (3.2) is satisfied.

We can now re-formulate our problem as that of the optimal control: find a control policy \( \hat{u}^i \) in order to minimize (in our case, we would like to make it zero) the criterion function \( \psi \) which is defined by

\[
\psi = \int_{x=0}^{x'} (\epsilon^i)^2 \, dx \quad (3.3)
\]

where

\[
\epsilon^i = \hat{u}^i - u^i \quad (3.4)
\]

subject to the constraints

\[
\frac{d^2 u^i}{dx^2} = \frac{1}{\alpha \Delta t} (2\hat{u}^i - u^i - u^{i-1})
\]

with initial conditions

\[
u^i(x = 0) = u_0^i \quad (3.5)
\]

\[
\frac{du^i}{dx} (x = 0) = u_m^i \quad (3.6)
\]

and also satisfying the terminal constraint equation

\[
\psi(u^i) = u^i(x = x') - u_f^i = 0 \quad (3.7)
\]

Note that the initial condition in (3.6) is still unknown, but it turns out that \( u_m^i \) is automatically determined once a control policy \( \hat{u}^i \) is selected.

In order to simplify subsequent derivations, we introduce new notation:

\[
u_1^i = u^i
\]

\[
u_2^i = \frac{du^i}{dx} = \frac{du^i}{dx}
\]

\[
\frac{d^2 u^i}{dx^2} = (\epsilon^i)^2 \quad \text{(i.e.,} \ u_4(x = x') = \psi) \quad (3.8)
\]
Equations (3.1) through (3.9) now become

\[ \frac{du_1^i}{dx} = u_1^i \]

\[ \frac{du_2^i}{dx} = \frac{1}{\alpha \Delta t} (2u_1^i - u_1^i - u_2^{i-1}) \]  \hspace{1cm} (3.9)

\[ \frac{du_3^i}{dx} = (\epsilon)^2 . \]

with initial conditions

\[ u_1^i(x = 0) = u_5^i \]  \hspace{1cm} (3.10)

\[ u_2^i(x = 0) = u_6^i \]

\[ u_3^i(x = 0) = 0 \]

and

\[ \epsilon^i = \hat{\alpha}^i - u_1^i \]  \hspace{1cm} (3.11)

\[ \theta = u_4^i(x = x') \]  \hspace{1cm} (3.12)

\[ \psi = u_1^i(x = x') - u_4^i = 0 . \]  \hspace{1cm} (3.13)

In matrix form, (3.9) and (3.10) become, respectively

\[ \frac{dU^i}{dx} = F^i(x, U^i, u_1^i, \hat{u}^i) \]  \hspace{1cm} (3.14)

\[ U^i(x = 0) = U_0^i \]  \hspace{1cm} (3.15)

where

\[ U^i = \begin{bmatrix} u_1^i \\ u_2^i \\ u_3^i \end{bmatrix} \]

and

\[ F^i = \begin{bmatrix} f_1^i \\ f_2^i \\ f_3^i \end{bmatrix} = \begin{bmatrix} u_1^i \\ \frac{1}{\alpha \Delta t} (2u_1^i - u_1^i - u_2^{i-1}) \end{bmatrix} (\epsilon)^2 \]  \hspace{1cm} (3.16)

The gradient method or the steepest-descent method\(^{8}\) is now applied.

We consider the system equation (3.14) and the corresponding perturbation equation

\[ \frac{d}{dx} (\delta U^i) = \frac{\partial F^i}{\partial U^i} \delta U^i + \frac{\partial F^i}{\partial \delta} \delta \hat{u}^i \]  \hspace{1cm} (3.17)

where the partial derivatives are evaluated along the nominal trajectory

\[ \frac{\partial F}{\partial U} = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} & \frac{\partial f_1}{\partial u_3} \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} & \frac{\partial f_2}{\partial u_3} \\ \frac{\partial f_3}{\partial u_1} & \frac{\partial f_3}{\partial u_2} & \frac{\partial f_3}{\partial u_3} \end{bmatrix} \]  \hspace{1cm} (3.18)

The adjoint equation to (3.17) is defined to be

\[ \frac{d\lambda^i}{dx} = - \left[ \frac{\partial F^i}{\partial \delta} \right]^T \lambda^i \]  \hspace{1cm} (3.19)

where the superscript T denoting the transpose of the matrix and \( \lambda^i = [\lambda_1^i \lambda_2^i \lambda_3^i]^T \). If equation (3.17) is pre-multiplied by \( \lambda^T \) and the transpose of equation (3.19) post-multiplied by \( \delta U \), and the sum is integrated over the interval \( 0 \leq x \leq x' \), we obtain

\[ \lambda^T \delta U^i \int_{0}^{x'} \lambda^T(x) G^i(x) \delta \lambda^i(x) \, dx \]  \hspace{1cm} (3.20)

where

\[ G(x) = \frac{\partial F}{\partial \mu} = \begin{bmatrix} \frac{\partial f_1}{\partial \mu} & \frac{\partial f_2}{\partial \mu} & \frac{\partial f_3}{\partial \mu} \end{bmatrix} \]  \hspace{1cm} (3.21)

Since we are still free to choose \( \lambda \), we let

\[ \lambda^T(x = x') = \frac{\partial \theta}{\partial U}(x = x') = \frac{\partial u_4}{\partial U}(x = x'). \]  \hspace{1cm} (3.22)

Then, \( \lambda^T \delta U(x = x') = \delta \theta(x = x') \) and equation (3.20) becomes

\[ \delta \theta^i = \lambda^T \delta U^i(x = 0) + \int_{0}^{x'} \lambda^T(x) G^i(x) \delta \lambda^i(x) \, dx . \]  \hspace{1cm} (3.23)

The \( \lambda^i \)'s are the influence functions or Green's functions which give the effect of small changes in the control function \( \hat{\alpha}^i \) and the initial state variables, on the
criterion function $\bar{\theta}^i$. The influence functions are obtained by equation (3.19) with the boundary (terminal) condition (3.22). In view of our objective to minimize $\bar{\theta}$ (i.e., to make $\delta\bar{\theta}$ as negative as possible), $\delta\bar{\theta}^i$ should be chosen such as to make the integral in (3.23) always negative regardless of $\delta U^i(x = 0)$. This can be realized if we choose

$$\delta\bar{\theta}^i = - K_\theta G^{iT} \lambda^i$$  \hspace{1cm} (3.24)

where $K_\theta$ is a positive constant to be determined later. This is the “steepest descent” direction to the minimum $\bar{\theta}^i$.

In order to satisfy the constraint equation (3.13) we further need to obtain another set of influence functions, say $\xi^i$, which will indicate the effects of $\delta u^i(x)$ as well as $\delta U^i(x = 0)$ upon the constraint function $\Psi$. Following an identical derivation as above, we obtain

$$f_{\Psi i}(x = x') = \xi^{iT} \delta U^i(x = 0) + \int_{x=0}^{x'} \xi^{iT}(x) G^{iT} \delta\bar{\theta}^i(x) \, dx$$  \hspace{1cm} (3.25)

where $\xi^i$ is the solution of

$$\frac{d\xi^i}{dx} = - \left[ \frac{\partial F^i}{\partial U^i} \right]^{T} \xi^i$$  \hspace{1cm} (3.26)

with the boundary (terminal) condition

$$\xi^T(x = x') = \frac{\partial\Psi}{\partial U}(x = x').$$  \hspace{1cm} (3.27)

Now, if the nominal trajectory approximately satisfies the constraint (3.13), then by setting $\delta\Psi^i$ equal to the negative of $\Psi^i(x = x')$, the next trajectory should satisfy $\Psi = 0$. The steepest descent direction to minimize $\bar{\theta}$ and satisfy a given $\delta\Psi^i$ is

$$\delta\bar{\theta}^i(x) = -(K_\theta G^{iT} \lambda^i + K_\Psi G^{iT} \xi^i)$$  \hspace{1cm} (3.28)

where $K_\Psi$ is another constant to be determined.

In equation (3.10), while two of the initial conditions are fixed, we still are faced with the problem of determining the correct initial condition on $u_2^i$. Fortunately, both equations (3.23) and (3.25) also relate the effect of the change in the initial values of the state variable upon the criterion function and the constraint function, respectively. Furthermore, the choice of $\delta\bar{\theta}^i(x = 0)$ is directly dictated by the choice $\delta\bar{\theta}^i$ as shown below. From equation (3.15),

$$\delta U^i(x = 0) = [0 \delta u^i 0]^T$$  \hspace{1cm} (3.29)

where

$$\delta u^i(x = 0) = \delta \left[ \frac{du^i}{dx} \right]_{x=0}$$  \hspace{1cm} (3.30)

Since we are seeking $\delta\bar{\theta}^i$ such that $\delta\bar{\theta}^i = u_2^i$, equation (3.30) can be written as

$$\delta u^i(x = 0) = \delta \left[ \frac{d\delta\bar{\theta}^i}{dx} \right]_{x=0}$$  \hspace{1cm} (3.31)

Due to the fact that $\delta\bar{\theta}^i$ is to be updated by $\delta\bar{\theta}^i$, we can rewrite (3.31) as

$$\delta u^i(x = 0) = \left[ \frac{d}{dx} (\delta\bar{\theta}^i) \right]_{x=0}$$  \hspace{1cm} (3.32)

Therefore, once $\delta\bar{\theta}^i$ is chosen according to (3.28), $\delta U^i(x = 0)$ in equations (3.23) and (3.25) is automatically determined. Combining of equations (3.23), (3.25), (3.28), (3.29), and (3.32) yields

$$\delta\theta(x = x') = - \left[ \lambda^i \frac{d}{dx} (K_\theta G^{iT} \lambda^i + K_\Psi G^{iT} \xi^i) + K_\Psi G^{iT} \xi^i \right]_{x=0}$$

and

$$\delta\Psi(x = x') = - \left[ \lambda^i \frac{d}{dx} (K_\theta G^{iT} \lambda^i + K_\Psi G^{iT} \xi^i) + K_\Psi G^{iT} \xi^i \right]_{x=0}$$  \hspace{1cm} (3.34)

For desired changes of $\delta\theta(x = x') = \Delta\theta$ and $\delta\Psi(x = x') = \Delta\Psi$, we can solve linear equations (3.33) and (3.34) for $K_\theta$ and $K_\Psi$ and obtain

$$K_\theta = \frac{Q_{22} \Delta\theta - Q_{12} \Delta\Psi}{Q_{12} Q_{22} - Q_{11} Q_{22}}$$  \hspace{1cm} (3.35)
\[ K_\phi = \frac{Q_{11} \Delta \Psi - Q_{21} \Delta \theta}{Q_{11} Q_{21} - Q_{11} Q_{22}} \]

with the initial conditions
\[ u_1^i(x = 0) = u_0^i \]
\[ u_2^i(x = 0) = \frac{d\Delta^i}{dx} (x = 0) \]
\[ u_3^i(x = 0) = 0 \]

where
\[ \Delta^i = \Delta^i - u_1^i \]

and \( \Delta^i \) is the nominal control.

The two sets of adjoint equations to be solved are
\[ \frac{d\lambda_1^i}{dx} = -\frac{1}{\alpha \Delta t} \lambda_1^i + 2\epsilon^i \]  
with the terminal conditions
\[ \lambda_1^i(x = x^*) = \lambda_2^i(x = x^*) = 0 \]

as obtained from equations (3.19) and (3.22), and

\[ \frac{d\xi_1^i}{dx} = -\frac{1}{\alpha \Delta t} \xi_1^i \]  
with the terminal conditions
\[ \xi_1^i(x = x^*) = 1, \quad \xi_2^i(x = x^*) = 0 \]

as obtained from equations (3.26) and (3.27).

It can be shown from (3.28) that the updating equation for the control policy is
\[ \delta \Delta^i = - [K_\phi \left( \frac{2}{\alpha \Delta t} \lambda_2^i + 2\epsilon^i \right) \]
\[ + K_\psi \left( \frac{2}{\alpha \Delta t} \xi_1^i \right)] \]

where \( K_\phi \) and \( K_\psi \) are given by equations (3.35) and

Computational algorithm of the new CSDT method

In the previous section, basic equations necessary to perform automatic updating of the control policy were derived, but some of the equations were left in a general form. In this section, the key equations will be first written in more specific terms, and then the hybrid computational algorithm will be summarized.

The state equations to be solved are
\[ \frac{d\mathbf{u}_1^i}{dx} = \mathbf{u}_2^i \]
\[ \frac{d\mathbf{u}_2^i}{dx} = \frac{1}{\alpha \Delta t} (2\Delta^i - \mathbf{u}_1^i - \mathbf{u}_1^{i-1}) \]
\[ \frac{d\mathbf{u}_3^i}{dx} = \epsilon_i^2 \]
(3.36). Furthermore the equations in (3.37) simplify to

\[ Q_u = [\lambda_s \frac{d}{dx} \left( \frac{2}{\alpha \Delta t} \lambda_s^{\prime} + 2 \epsilon_i \right)]_{x=0} \]

\[ + \int_{-\infty}^{x} \left( \frac{2}{\alpha \Delta t} \lambda_s^{\prime} + 2 \epsilon_i \right) \frac{2}{\alpha \Delta t} \xi_t^{\prime} \, dx \]

\[ Q_{\xi_t} = - \left[ \frac{2}{\alpha \Delta t} \lambda_s^{\prime} \left( \frac{2}{\alpha \Delta t} \xi_t^{\prime} \right) \right]_{x=0} \]

\[ + \int_{-\infty}^{x} \left( \frac{2}{\alpha \Delta t} \lambda_s^{\prime} + 2 \epsilon_i \right) \left( \frac{2}{\alpha \Delta t} \xi_t^{\prime} \right) \, dx \quad (4.9) \]

\[ Q_u = \left[ \xi_t \frac{d}{dx} \left( \frac{2}{\alpha \Delta t} \lambda_s^{\prime} + 2 \epsilon_i \right) \right]_{x=0} \]

\[ + \int_{-\infty}^{x} \left( \frac{2}{\alpha \Delta t} \lambda_s^{\prime} + 2 \epsilon_i \right) \left( \frac{2}{\alpha \Delta t} \xi_t^{\prime} \right) \, dx \]

\[ Q_{\xi_t} = - \left[ \frac{2}{\alpha \Delta t} \lambda_s^{\prime} \left( \frac{2}{\alpha \Delta t} \xi_t^{\prime} \right) \right]_{x=0} \]

\[ + \int_{-\infty}^{x} \left( \frac{2}{\alpha \Delta t} \lambda_s^{\prime} + 2 \epsilon_i \right) \left( \frac{2}{\alpha \Delta t} \xi_t^{\prime} \right) \, dx . \]

Actual updating of the control policy is accomplished by

\[ \hat{u}^t (\text{NEW}) = \hat{u}^t (\text{OLD}) + \delta \hat{u}^t . \quad (4.10) \]

The initial estimate of the control policy can be obtained as follows. If we apply the forward CSTD approximation to equation (2.1), we obtain

\[ \frac{d^2 u^t}{dx^2} = \frac{1}{\alpha \Delta t} (u^{t+1} - u^t) \quad (4.11) \]

E squared this equation with (2.5) and simplifying,

\[ u^{t+1} = 2u^t - u^{t-1} . \quad (4.12) \]

This equation provides a rough estimate on the state function for the next time increment. Therefore, it is reasonable to use this equation to compute the first nominal control policy as

\[ \hat{u}^{t+1} = 2u^t - u^{t-1} = 2u^t - u^{t-1} \quad (4.13) \]

The overall hybrid computational algorithm then becomes:

1) With a nominal control policy \( \hat{u}^{t} \), solve equation (4.1) on the analog subsection of the hybrid computer with the initial conditions (4.2). Both \( \hat{u}^{t} \) and \( u^{t-1} \) are played back via DAL’s from the digital subsection into the analog subsection. Meanwhile, read \( \epsilon^t \) into the digital subsection via an ADL.

At the end of the run corresponding to \( x = x' \),

2) record the values of \( u_t^t \) (i.e., \( \theta \)) and \( u_{t-1}^t \). These values are used digitally to compute \( \Delta \theta \) and \( \Delta \Psi \) to be used in step 5. \( (\Delta \theta = -\beta u_t^t(x'), \Delta \Psi = -\beta u_{t-1}^t \) where \( 0 < \beta \leq 1 \); the more non-linear the problem the smaller the value of \( \beta \). If a convergence has been attained (e.g., Max | \( \epsilon_i \) | < specified bound), go to step 6.

3) Next, solve the adjoint equations (4.4) and (4.6) backward in analog computer time, \( t_c \), \( (t_c = x' - x) \) since we are given the terminal conditions instead of the usual initial conditions. Note that this process changes the signs of the left-hand sides of these equations. \( \epsilon^t \) which was recorded in step 1 is played back via a DAL in the reverse direction. The terms required by equation (4.8) are recorded via ADL’s. At the same time compute in the analog subsection the values of the integrals which appear in equations (4.9).

4) At the end of the adjoint run, record the final values of \( \lambda_{t+1} \), \( \xi_{t+1} \), and \( \xi_{t-1} \) which actually are \( \lambda_t \), \( \xi_t \) \( (x = 0) \), and \( \xi_{t-1} \) \( (x = 0) \), respectively.

5) Digitally compute \( Q_u, Q_{\xi_t}, Q_{u\xi_t}, \) and \( Q_{u\xi_t} \). The derivatives which need to be evaluated in (4.9) at \( x = 0 \) can be digitally approximated. Using these values and \( \Delta \theta \) and \( \Delta \Psi \) which were computed in step 2, obtain \( K_u \) and \( K_{\xi_t} \) according to (3.35) and (3.36). Update the control policy according to (4.8) and (4.10), and go to step 1.

6) Compute a new nominal control policy by equation (4.13). (For the very initial prediction of \( u^{t+1} \) where \( i = 0 \), \( u_t^0(x) \) can be used.) Increment \( i \) by one and go to step 1.

7) Repeat steps 1 through 6 until \( i = N \). A typical hybrid mechanization diagram for the new method is shown in Figure 2.

Discussion of the new method and its extension

With the application of the functional optimization technique, the unstable analog loop in the classical CSTD method has been replaced by a stable loop. Although the parameter (initial condition) optimization problem involved in the classical method has been replaced by a more complicated functional optimization problem in the new method, the high-speed computational capabilities of the hybrid computer make this
method practical. Furthermore, with removal of the unstable loop, it is now possible to reduce the size of $\Delta t$ in order to minimize the truncation error of the CSDT approximation.

A similar technique can be applied to the CSDT version of the well-known Crank-Nicholson numerical approximation

$$\frac{1}{2} \left[ \frac{d^2 u^t}{dx^2} + \frac{d^2 u^{t-1}}{dx^2} \right] = \frac{u^t - u^{t-1}}{\alpha \Delta t}. \quad (5.1)$$

In order to avoid the explicit computation of $\frac{d^2 u^{t-1}}{dx^2}$ as well as the unstable computational loop, introduce a new variable $w^t$ defined by

$$w^t = \frac{1}{2} (u^t + u^{t-1}) \quad (5.2)$$

from which

$$u^{t-1} = 2w^t - u^t. \quad (5.3)$$

Combining of (5.1), (5.2), and (5.3) yields

$$\frac{d^2 w^t}{dx^2} = \frac{2}{\alpha \Delta t} (u^t - w^t). \quad (5.4)$$

Equation (5.4) is now a stable equation in state variable $w^t$, and $u^t$ becomes the external control policy to be optimized. The criterion function $\vartheta$ to be minimized becomes (from 5.3)

$$\vartheta = \int_0^t \left[ (2w^t - u^t) - u^{t-1} \right]^2 dx. \quad (5.5)$$

The new CSDT and Peaceman-Rachford ("alternating direction implicit") methods can be combined to solve two dimensional diffusion equations. As an illustration, the procedure is explained for the simple constant diffusivity case:

$$\frac{\partial^2 u(x, y, t)}{\partial x^2} + \frac{\partial^2 u(x, y, t)}{\partial y^2} = \frac{1}{\alpha} \frac{\partial u(x, y, t)}{\partial t}. \quad (5.6)$$

where $x = 0$, $x = x^*$, $y = 0$, $y = y^*$, $t = 0$, $t = t^*$, with boundary and initial conditions

$$u(0, y, t) = u_0(y, t) \quad u(x^*, y, t) = u_0(y, t) \quad u(x, 0, t) = u_0(x, t) \quad u(x, y, 0) = u^0(x, y).$$

The basic process in passing from $t^i$ to $t^{i+1}$ has two parts; each part involving the solution of a set of ordinary differential equations by the CSDT technique.

We will need to find $u$ and $\frac{\partial^2 u}{\partial x^2}$ along the lines $y = y_m, m = 1, \ldots, M-1$, for the first part (note that $y = 0$, $y = y_m$ are boundaries). The second part will consist of finding $u$ and $\frac{\partial^2 u}{\partial y^2}$ along the lines $x = x_\ell, \ell = 1, \ldots, L-1$. At the end of the first part we will have passed to $t^{i+\frac{1}{2}}$. Thus, $u$ will be defined on a $(M + 1) \times (L + 1)$ grid at $t^{i+\frac{1}{2}}$ (the analog solutions are digitized only at these discrete points). During the process, there will be similar tables of $\frac{\partial u}{\partial x}$ and $\frac{\partial u}{\partial y}$.

With the foregoing in mind, we proceed to explain the procedure. At $t^i$, $u$ and $\frac{\partial u}{\partial y}$ are defined on the $(M + 1) \times (L + 1)$ grid in the interior of the region $x = 0, x^*$, $y = 0, y^*$. First, we find the set of functions $u_m^{i+1}(x) = u(x, y_m, t^{i+1})$.\]
on the lines \( y = y_m \) from the equations

\[
\frac{d^2 u_{m+1}(x)}{dx^2} + \frac{d^2 u_m(x)}{dy^2} = \frac{u_{m+1}(x) - u_m(x)}{\alpha \Delta t} - \frac{d^2 u_m(x)}{dy^2},
\]

\( m = 1, \ldots, M-1 \)  \( (5.7) \)

with analog integration. Because \( y \) is fixed, the spatial derivatives of \( (5.6) \) reduce to total derivatives. The term \( \frac{d^2 u_m(x)}{dy^2} \) is formed by interpolation from the previously stored \( \frac{d^2 u_m(x)}{dy^2} \) data on the \((M + 1) \times (L + 1)\) grid at \( t_i \).

The functional optimization method (as previously described) is used to solve equations \( (5.7) \) from

\[
\frac{d^2 u_{m+1}(x)}{dx^2} = \frac{2u_{m+1}(x) - u_{m+1}(x) - u_m(x)}{\Delta t} - \frac{d^2 u_m(x)}{dy^2},
\]

\( (5.8) \)

where the \( u_{m+1}(x) \) are the external control policies to be optimized (i.e., \( u_{m+1} \) converges to \( u_{m+1} \)). When equations \( (5.8) \) have been solved, \( u_{m+1}(x) \) and \( \frac{d^2 u_{m+1}(x)}{dx^2} \) (which is also available from the analog program) are digitized and stored over the grid at \( t_{i+1} \), and the first part is complete. The second part of the procedure consists of finding the set of functions on the lines \( x = x_l \) from the equations

\[
\frac{d^2 u_{m+1}(y)}{dy^2} + \frac{d^2 u_{m+1}(y)}{dx^2} = \frac{u_{m+1}(y) - u_{m+1}(y)}{\alpha \Delta t},
\]

\( \ell = 1, \ldots, L-1 \)  \( (5.9) \)

As before, the term \( \frac{d^2 u_{m+1}(y)}{dx^2} \) is obtained by interpolation of the \( \frac{d^2 u_{m+1}(x)}{dx^2} \) values stored on the grid at \( t_{i+1} \). These equations are solved with the same functional optimization procedure. The solutions \( u_{m+1}(y) \) and \( \frac{d^2 u_{m+1}(y)}{dy^2} \) are digitized and stored over the grid at \( t_{i+1} \). This completes the second part and also one step of the CSDT procedure.

The values of \( u \) over the grid for successive \( t_i \), \( i = 1, 2, \ldots \) are the solution to the problem defined by \( (5.6) \).

A few words regarding the details of generating certain of the second derivatives at grid boundaries may be helpful. The values of \( u_{m+1}(x) \) for \( m = 0, M \) are automatically available as boundary values. But, the values of the second derivatives \( \frac{d^2 u_{m+1}(x)}{dx^2} \) for \( m = 0, M \) must be approximated digitally for each \( x_l \). This is necessary because during the subsequent half-step (i.e., proceeding from \( t_{i+1} \) to \( t_{i+2} \)) they are both required for external driving functions. The same requirement is imposed on the second derivatives \( \frac{d^2 u_{m+1}(y)}{dy^2} \) for \( \ell = 0, L \). Of course, the second derivatives required to start the process at \( t = 0 \) can be calculated from \( u^0(x,y) \).

REFERENCES

1. S. H. JURY
   Solving partial differential equations
   Industrial and Engineering Chemistry Vol 53 pp 177-180 1961

2. D. R. ELLIS
   Analog solution of the diffusion equation with continuous space and discrete time variables
   MS Thesis Dept of Elec. Engr MIT Aug 1963

3. R. MORONEY
   Hybrid computer solution of parabolic partial differential equations via the continuous space discrete time method
   MIT Electronic Systems Lab DSR 9128-wp-16 Jan 1964

4. H. S. WITSENHAUSEN
   Hybrid solution of initial value problems for partial differential equations
   MIT Electronic Systems Lab DSR 9128 Memo No 8 Aug 1964

5. R. BENZ
   A hybrid computer solution of certain field problems
   Boeing Company Internal Memo Jan 1965

6. SHU-KWAN CHAN
   A study on certain hybrid techniques for the solution of the heat equation
   MIT Electronics Systems Lab DSR 9128 Memo No 11 Dec 1965

7. I. M. MANLEY
   Hybrid computer solution of the diffusion equation with continuous space and discrete time variables using sensitivity equations
   MS Thesis Dept of Engr Univ of California at Los Angeles 1966

8. R. VICHNEVETSKY
   A new stable computing method for the serial hybrid computer
574 Fall Joint Computer Conference, 1968

Integration of partial differential equations
Proceedings of the Spring Joint Computer Conference 1968
pp 143–150

9 A E BRYSON W R DENHAM
A steepest-ascent method for solving optimum programming

10 J TODD
Survey of numerical analysis