Hybrid computing techniques for solving parabolic and hyperbolic partial differential equations

T. I. Silvey† and J. R. Barker*

* *School of Mathematics, Bath University of Technology*
† *Electronic Associates Ltd. (formerly at Bath University)*

This paper describes the development of the serial method for solving partial differential equations of parabolic or hyperbolic type, and describes some recent improvements in the method which can be carried out on a hybrid computer.

(Received June 1969)

The solution of parabolic and hyperbolic partial differential equations by analogue or hybrid computers began with mechanical differential analysers in the 1930's and has been the subject of many papers. Most of the methods used involve the reduction of the partial differential equations to a set of ordinary differential equations by using finite differences, and there is a choice as to which variable is left in continuous form leading to various computing processes (Hartree, 1950; MacKay and Fisher, 1962).

As an example of a parabolic equation, consider the simple heat-conduction equation in one space dimension, x, and time, t, with $u(x, t)$ denoting temperature,

$$\frac{\partial^2 u}{\partial x^2} = \frac{\partial u}{\partial t}; \quad 0 \leqslant x \leqslant L, 0 \leqslant t < \infty \qquad (1)$$

given the initial condition $u(x,0)$, and the two end conditions $u(0,t)$ and $u(L,t)$. A finite-difference expansion of the second derivative in x leads to:

$$\left.\begin{array}{l} \dfrac{u_{n+1} - 2u_n + u_{n-1}}{(\Delta x)^2} = \dfrac{du_n}{dt}; \quad n = 1, 2, \ldots, N-1 \\ \text{where} \qquad\qquad N = L/\Delta x \end{array}\right\} \quad (2)$$

This is a set of ordinary differential equations suitable for simultaneous, i.e. 'parallel' solution by an analogue computer. No problems of stability arise, but the quantity of equipment required can be formidable, especially if the physical problem is more complicated, although Hartree and his group at the University of Manchester found close agreement between $u(x, t)$ and $u_n(t)$ using surprisingly coarse subdivisions in the x-direction.

Due to shortage of equipment (a mechanical differential analyser had no more than eight integrators) Hartree's group gave more attention to the serial method. In its practical form, this involves taking finite differences in t instead of in x, such as with the following backward-difference scheme:

$$\frac{d^2u_j}{dx^2} = \frac{u_j - u_{j-1}}{\Delta t}; \quad j = 1, 2, 3, \ldots; \quad t = j\Delta t \qquad (3)$$

So,

$$\frac{d^2u_j}{dx^2} - \frac{1}{\Delta t}u_j = -\frac{1}{\Delta t}u_{j-1} \qquad (4)$$

where

$$u_j(x) = u(x, j\Delta t)$$

The initial temperature distribution $u_0(x)$ and the boundary conditions [e.g. values of $u_j(0)$ and $u_j(L)$] are given. The variable x is represented by computer time τ through $x = \beta\tau$ where β is a constant and $0 \leqslant \tau \leqslant L/\beta$. The set of equations (4) can be solved one at a time, using a *single* differential equation solver, to produce the solution over the whole x-range at successive times Δt, $2\Delta t$, $3\Delta t$, etc. Each solution $u(x, j\Delta t)$ is put into a function store and played back into the equation solver on the $j + 1^{\text{th}}$ run. As is well known, solving the ordinary differential equation (4) with known boundary values $u_j(0)$ and $u_j(L)$ is done on an analogue computer by iteration, adjusting the unknown initial condition $du_j(0)/dx$ until the solution takes the value $u_j(L)$ at $x = L$. The difficulty is that the differential equation has a complementary function (C.F.) of the form

$$\text{C.F.} = A \exp(-x/\sqrt{\Delta t}) + B \exp(+x/\sqrt{\Delta t}) \qquad (5)$$

in which the second term grows exponentially throughout the run. As one can show that B is small in the solution required, any error in the computer which effectively alters B will give rise to an exponential growth of error which may cause the solution obtained to have serious errors in between the boundary values at $x = 0$ and $x = L$.

The smaller Δt is made in order to reduce the finite-difference error, the more sensitive does the solution become to the setting of $du_j(0)/dx$ and to noise in the computer; this is because the term $B \exp (x/\sqrt{\Delta t})$ in Eqn. 5 grows faster. In an extreme case, noise may make it impossible to match the required value of $u_j(L)$ because no two runs are alike.

The accuracy of the serial method can be optimised by decreasing the finite-difference interval Δt until the decreasing finite-difference error is matched by the increasing error due to noise and instability in the analogue computer. Some investigations were carried out by Silvey (1966) on the problem specified by Eqn. 1 with $u(x,0) = 1$ and $u(0,t) = u(L,t) = 0$. Using 1 or 2 seconds to scan the x-range from 0 to L, the optimum value of $\Delta t/L^2$ is about $0{\cdot}025$. The finite-difference error in $u(x, t)$ has a maximum of $0{\cdot}05$ at $t = 0{\cdot}15$ if $L = 1$. All this assumes that the solution exactly satisfies the far boundary value, but to achieve this within 1% requires adjustment of $du_j(0)/dx$ to within 3×10^{-3}% which is almost impossible. The noise level relative to the signals in the, now, obsolete mechanical differential analysers was significantly less than in the present-day electronic analogue computers.

In Eqn. 4, the formula is of backward-difference type with a finite-difference error proportional to Δt. Hartree used a central-difference formula

$$\frac{1}{2}\left(\frac{d^2u_j}{dx^2} + \frac{d^2u_{j-1}}{dx^2}\right) = \frac{u_j - u_{j-1}}{\Delta t} \qquad (6)$$

for which the error is proportional to $(\Delta t)^2$. Hartree's formula is more accurate than Eqn. 3, but is only marginally stable in that errors generated in the solution at any one time step are propagated to later times without attenuation, so that serious accumulation of error can occur if many steps are taken. The backward-difference formula (4) is completely stable in this respect as errors at each time step die out in the later steps. Mitchell (1963) suggested that the difference equation could be adjusted to be intermediate between backward and central differences to give the best compromise between accuracy of formula and stability. Fisher (see MacKay and Fisher, 1962) gave noise figures for the case of Hartree's central-difference formula which are similar to Silvey's later results (quoted above) for the backward-difference case.

Much of the early work using mechanical differential analysers was of high quality but tends to be forgotten. For example, Miura and Iwata proposed in 1966 that Richardson's process of h^2-extrapolation should be used to reduce finite-difference errors in the parallel method, yet it appears to have been regularly used—on the serial method—by Hartree's group, the theory being given by Hartree and Womersley in 1937. Neither the serial nor parallel method are restricted to the simple problem discussed so far. The co-ordinate system need not be Cartesian, the thermal properties of the substance can vary with temperature (Eyres *et al.*, 1946), or with position. Heat can be generated in the substance itself, e.g. by dielectric heating (Copple *et al.*, 1939), and changes of state can occur with consequent liberation of latent heat and movement of (say) a liquid-solid boundary (Butler *et al.*, 1962).

Recent improvements to the serial method

Considerably improved accuracy (Silvey, 1966) was obtained by rearranging the equations to compute, not the solution at each stage, but the difference between successive solutions as j was increased, but the improvement only occurs if the difference is small, that is, if Δt is small enough to keep $[u_j(x) - u_{j-1}(x)]/u_j(x)$ much less than unity for all x and j.

In an alternative approach, solutions of the differential equation (4) were computed from $x = 0$ ('forwards') and from $x = L$ ('backwards'), the solutions being matched in the middle by adjusting one unknown initial condition in each. The troublesome part of the complementary function (see Eqn. 5) is the 'B' term when computing forwards, and the 'A' term when computing backwards. However, if $L = 1$ and $\Delta t = 0{\cdot}025$, the term $B \exp (x/\sqrt{\Delta t})$ is $557B$ at the right-hand boundary but is only $24B$ at the centre point. Hence the instability trouble is much reduced for a given Δt, but, of course, matching the value and the slope in the middle is complicated.

Another technique considered was the use of higher order difference formulae to reduce the truncation error, but MacKay and Fisher (1962) pointed out that these introduce spurious additions to the true solution which, once started by errors in the analogue computer, may grow to significant size. However, a scheme has been invented (Silvey, 1966) which does not introduce spurious solutions. The finite-difference error of the equations is $0((\Delta t)^3)$ which is two orders of magnitude better than the simple scheme. Each time step is divided into thirds, and the partial differential equation reduces to three coupled ordinary differential equations:

$$\left.\begin{aligned}
&\frac{d^2u_{j-2/3}}{dx^2} + \frac{3}{2\Delta t}u_{j-2/3}\\
&\qquad = -\frac{1}{2\Delta t}u_j + \frac{3}{\Delta t}u_{j-1/3} - \frac{1}{\Delta t}\bar{u}_{j-1}\\
&\frac{d^2u_{j-1/3}}{dx^2} - \frac{3}{2\Delta t}u_{j-1/3}\\
&\qquad = \frac{1}{\Delta t}u_j - \frac{3}{\Delta t}u_{j-2/3} + \frac{1}{2\Delta t}\bar{u}_{j-1}\\
&\frac{d^2u_j}{dx^2} - \frac{11}{2\Delta t}u_j\\
&\qquad = -\frac{9}{\Delta t}u_{j-1/3} + \frac{9}{2\Delta t}u_{j-2/3} - \frac{1}{\Delta t}\bar{u}_{j-1}
\end{aligned}\right\} \qquad (7)$$

where $j = 0, 1, 2, \ldots ; t = j\Delta t$; the 'bar' over u_{j-1} denotes that this function has been stored.

The three interconnected second-order differential equations (needing six integrators) are solved simultaneously and all of them are driven by $\bar{u}_{j-1}$ which was generated and stored during the previous time step. In order to satisfy the boundary conditions at $x = 0$ and $x = L$, iterative techniques may be used. One possible method is to set approximate values of $du_{j-2/3}(0)/dx$, $du_{j-1/3}(0)/dx$, and $du_j(0)/dx$ on the appropriate integrators and then to adjust them one at a time (in a sequence) to satisfy the known boundary values of $u_{j-2/3}(L)$, $u_{j-1/3}(L)$, $u_j(L)$. The exact conditions under which this scheme would converge have not been investigated as we have concentrated our work on the method described in the next section.

Note that all three of the improvements discussed in this section could—in principle—be combined in one scheme.

A new stable method

The most significant and recent improvement in the serial method is the 'method of decomposition' due to Vichnevetsky (1968) and, independently, to one of us (T.I.S.) at Bath, but our publication was delayed for lack of suitable equipment with which to test the method. However, our version has some advantages and is likely to be the more accurate so that a brief account is now given together with some of our test results.

The simplest example of the method is obtained from the backward-difference scheme of Eqn. (4) by factorising the operator on the left-hand side, giving

$$\left(\frac{d}{dx}+\frac{1}{\sqrt{(\Delta t)}}\right)\left(\frac{du_j}{dx}-\frac{u_j}{\sqrt{(\Delta t)}}\right)=-\frac{\bar{u}_{j-1}}{\Delta t} \qquad (8)$$

where the bar again indicates the solution at the previous time step read out from a store.

A new variable v_j is introduced and defined by

$$\frac{du_j}{dx}-\frac{u_j}{\sqrt{(\Delta t)}}=\frac{\bar{v}_j}{\sqrt{(\Delta t)}} \qquad (9)$$

so that Eqn. 8 becomes

$$\frac{dv_j}{dx}+\frac{v_j}{\sqrt{(\Delta t)}}=-\frac{\bar{u}_{j-1}}{\sqrt{(\Delta t)}} \qquad (10)$$

However, although Eqn. 10 is stable, Eqn. 9 is unstable with increasing x. Putting

$$y = L - x \text{ and } dy = -dx \qquad (11)$$

into Eqn. 9 gives

$$\frac{du_j}{dy}+\frac{u_j}{\sqrt{(\Delta t)}}=-\frac{\bar{v}_j}{\sqrt{(\Delta t)}} \qquad (12)$$

which is stable for increasing y. The process of solution now consists of solving these two stable first order differential equations, *in turn*, representing x and then y by computer time.

Consider the process for the j^{th} time step. Firstly, Eqn. 10 is solved using integrator I_1 of **Fig. 1** which is driven by $\bar{u}_{j-1}(x)$ from the previous time step, and $v_j(x)$ is stored. Storage is achieved by the analogue-digital-converter ('ADC' in Fig. 1) which takes samples from the waveform $v_j(x)$ at regular and frequent intervals and puts them in the digital computer core-store. The driving function $\bar{u}_{j-1}(x)$, which had been similarly stored at an earlier stage, is passed to integrator I_1 via the digital-analogue-converter ('DAC'). The proper value for the initial condition in integrator I_1, solving Eqn. 10, is not known at this stage and a guessed value v_g is provided via I_2 which is also in the initial-condition state at this time.

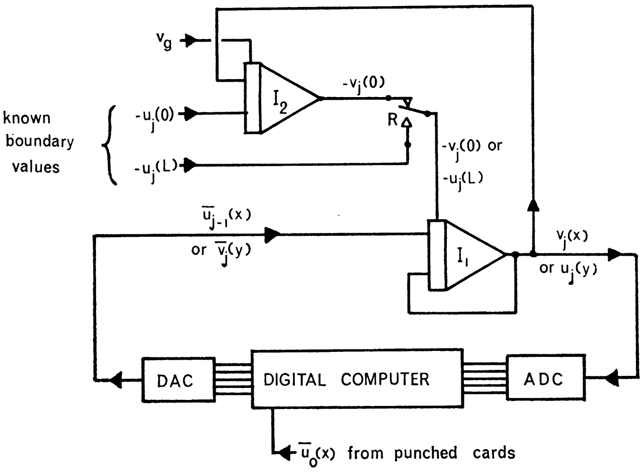


Fig. 1. Arrangement for solving heat conduction problem by new serial method. Upper labels refer to solving Eqn. 10 and lower ones to Eqn. 12

Secondly, Eqn. 12 is solved by integrator I_1 now driven by the stored $\bar{v}_j$ (read backwards). The initial value on integrator I_1 is $u_j(\text{at } y = 0) = u_j(L)$ which is the known boundary temperature at $x = L$ and $t = j\Delta t$. As the proper value for the integrator initial value for Eqn. 10, $v_j(\text{at } x = 0)$, was not known, the function $u_j(y)$ generated from Eqn. 12 will be in error; in particular, the value at the end of the integration will not be the known boundary temperature. From the measured discrepancy, a correction to $v_j(x = 0)$ can be calculated. Our method, which differs from Vichnevetsky's at this point, is to find the sensitivity coefficient relating changes in $u_j(y = L)$ to changes in $v_j(x = 0)$ and to use this in an automatic, self-checking, iterative procedure until u_j satisfies the boundary temperature to within 0·002 machine unit. A simple analysis shows that

$$\frac{\partial u_j(0)}{\partial v_j(0)}=-\tfrac{1}{2}[1-\exp\left(2L/\sqrt{\Delta t}\right)] \qquad (13)$$

which, as $\Delta t/L^2$ is always small, is very nearly equal to $-\frac{1}{2}$. Hence if the achieved value of $u_j(y = L)$ is less than the required value by $\Delta u_j(0)$, the correction to be added to $v_j(0)$ is closely given by

$$\Delta v_j(0) = -2\Delta u_j(0) \qquad (14)$$

where '0' denotes '$x = 0$'. This correction can be made in practice by using track-hold circuits or by accumulating the increments on an integrator.

In Fig. 1, integrator I_2 accumulates the corrections by integrating $\Delta u_j(0)$ from Eqn. 14 for any convenient short constant time S through a gain of $2/S$. In between corrections, this integrator is in the 'hold' state. The integrator is loaded with the guessed value of $v_j(0)$ (denoted v_g) at the beginning of the process for the j^{th} time step. After each correction, Eqns. 10 and 12 are run again. One correction is sufficient in theory as the process is linear but in practice two corrections are needed.

The final function u_j which satisfies the boundary conditions is retained inside the digital store ready to be used as the driving function $\bar{u}_{j-1}$ for the next time step, and the whole process is repeated.

The method was tried out on an E.A.I. 690 hybrid computer; the control program was implemented on the parallel-logic section and the digital computer was used merely for storage and playback. The analogue-digital-converter took 180 samples in the range 0 to L; the reconstructed waveforms from the DAC were staircases with 179 steps. Experiments using fewer samples in the range suggested that errors due to the discrete nature of the storage were insignificant compared with other errors. The analogue signals were converted into a digital form of 13 bits plus sign.

The sampling interval for storage and playback was

2 ms so that, with 180 samples, each integration from 0 to L took 0·36 s. It is anticipated however that by writing the function storage and playback program in Assembler language instead of FORTRAN IV, this time could be cut by at least a factor of 10. However, graphs of $u_j(x)$ have to be recorded on an electro-mechanical graph plotter for some values of j and, for these, the final integration of Eqn. 12 has to be slowed down.

Such graphs were obtained on the graph plotter of the E.A.I. 690 hybrid computer for the cooling problem with both boundary temperatures equal to zero, and an initial temperature distribution everywhere equal to unity, with $L = 1$, $\Delta t = 0{\cdot}005$. This describes the temperatures inside a parallel-sided slab, of thickness L, heated to unit temperature and then plunged into a cold fluid at zero temperature. The temperatures on the actual graph sheet were almost perfectly symmetrical about the centre of the slab, the biggest discrepancy being about 1 part in 100. The first column of **Table 1** shows the temperatures at the centre ($x = 0{\cdot}5$) at various times as printed from the digital store of the hybrid computer. The second column gives the corresponding values calculated by a digital computer from the known analytic solution of Eqns. 4. If the hybrid computer were completely free of errors, these two columns would agree exactly. At the 75th time step ($t = 0{\cdot}375$), the error is only 0·2% relative to the initial temperature. The maximum relative error in the table is 1% at about the 20th time step. The third column gives values calculated from the known analytic solution of the partial differential equation (1); these differ from the values in the second column on account of the finite-difference error. This latter error is greatest at the beginning of the process because of the very rapid initial cooling near the two boundaries. For example, at $x = 0{\cdot}1$, $t = 0{\cdot}005$, the discrepancy between the two analytic solutions reached 7%, but at the centre of the slab the worst is 1·2% at $t = 0{\cdot}02$. It should be remembered that the given boundary conditions are of an extreme nature and that less severe ones would be encountered in engineering practice, resulting in smaller errors.

One trial run and two correcting runs (at each time step) satisfy the boundary conditions to within 0·001, and the iterative method used here is thought likely to give greater accuracy than the method of superimposing solutions used by Vichnevetsky.

Table 1

Temperatures $u_j(0.5)$ at centre of slab at times $t = 0.005j$

	TEMPERATURES		
j	HYBRID COMPUTER	DIFFERENTIAL-DIFFERENCE EQNS. 4	PARTIAL DIFFERENTIAL EQN. 1
1	0.9969	0.9983	1.0000
3	0.9767	0.9802	0.9922
5	0.9322	0.9374	0.9493
6	0.9029	0.9088	0.9175
8	0.8368	0.8440	0.8458
10	0.7676	0.7759	0.7723
12	0.7001	0.7091	0.7022
15	0.6066	0.6165	0.6068
18	0.5246	0.5344	0.5236
22	0.4314	0.4411	0.4299
26	0.3551	0.3639	0.3529
32	0.2653	0.2726	0.2625
42	0.1636	0.1684	0.1602
54	0.0928	0.0945	0.0886
75	0.0366	0.0344	0.0314

The special feature of the method is the small amount of analogue equipment used, which leaves the bulk of the analogue available for simulating other systems to which the heat-flow equation might be coupled. In such a case, the computing time should be the same for each increment Δt of problem time, which would be so if a fixed number of runs are made at each time step, three being sufficient as already mentioned.

Satisfactory results were also obtained when one face of the slab was thermally insulated,

$$\frac{du_j}{dx} = 0 \quad \text{at } x = 0$$

and heat was drawn out through the other face according to

$$\frac{du_j}{dx} = -\, u_j(L) \quad \text{at } x = L$$

Hyperbolic equations such as the one-dimensional wave-equation

$$\frac{\partial^2 \psi}{\partial x^2} = \frac{1}{C^2}\frac{\partial^2 \psi}{\partial t^2}$$

can also be solved by the serial method with factorisation, and progress has been made in solving some non-linear equations.

The counterflow heat exchanger

Fig. 2 shows a simple heat exchanger in which heat transfer takes place across a thin wall separating two fluids moving in opposite directions along the x-axis at constant speeds v_1 and v_2. The temperatures T, θ and W are functions of time t and distance x; fluid turbulence prevents temperature variations at right angles to the fluid flow while the wall is so thin that heat conduction in the x-direction can be neglected. With these assumptions, the equations of the system are

$$\begin{aligned} &\text{Primary fluid:} && \frac{\partial T}{\partial x} = -\, k_1(W - T) + \frac{1}{v_1}\frac{\partial T}{\partial t} \\ &\text{Secondary fluid:} && \frac{\partial \theta}{\partial x} = k_2(W - \theta) - \frac{1}{v_2}\frac{\partial \theta}{\partial t} \\ &\text{Wall:} && 0 = \mu_1(T - W) + \mu_2(\theta - W) - \lambda\frac{\partial W}{\partial t} \end{aligned} \qquad (15)$$

where k_1, k_2, μ_1, μ_2 and λ are constants.

Carling (1968) described a serial process for the solution of these equations. The time derivatives are replaced by finite-difference approximations changing the system to two first-order ordinary differential equations to be integrated with respect to x, together with one algebraic equation:

$$\frac{dT_j}{dx} = -\, k_1(W_j - T_j) + \frac{1}{v_1}\frac{T_j - T_{j-1}}{\Delta t} \qquad (16)$$

$$\frac{d\theta_j}{dx} = k_2(W_j - \theta_j) - \frac{1}{v_2}\frac{\theta_j - \theta_{j-1}}{\Delta t} \qquad (17)$$

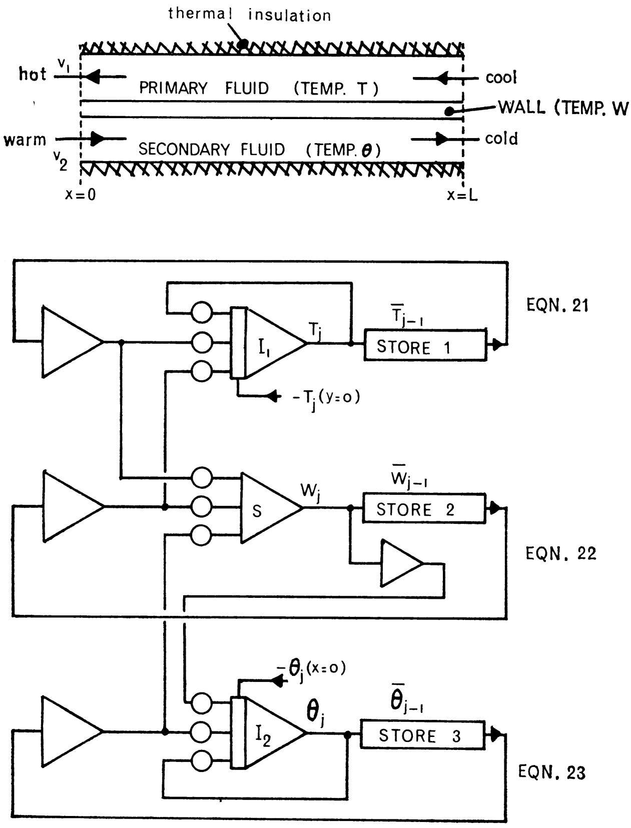


Fig. 2. Counterflow heat exchanger and its analogue

$$0 = \mu_1(T_j - W_j) + \mu_2(\theta_j - W_j) - \lambda\left(\frac{W_j - W_{j-1}}{\Delta t}\right) \quad (18)$$

where $j = 1, 2, 3, \ldots$; $t = t_j = j\Delta t$. $W_0(x)$, $\theta_0(x)$ and $T_0(x)$ are the given initial temperature distributions in the system. $T_j(L)$ and $\theta_j(0)$ are the boundary conditions representing the fluid inlet temperatures and are assumed to be known for all j, that is at all times $t \geqslant 0$. The finite-difference errors in all three equations are proportional to Δt. For constant coefficients, Eqn. 16 has a complementary function A exp $[k_1 + 1/(v_1\Delta t)]x$ so that it is unstable for x increasing, and decreasing Δt reduces the finite-difference error but worsens the instability. However, by computing the solution of Eqn. 16 'backwards', this instability is completely removed. (It is interesting to note that both integrations will then be carried out in the directions of flow of the fluids.)

Economy of equipment can be achieved by using the 'serial' method of solution in which one analogue of Eqns. 16–18 is used over and over again for successive time steps. This requires that hybrid computing facilities be available because three temperature functions $T_j(x)$, $\theta_j(x)$ and $W_j(x)$ generated at the j^{th} time step have to be stored for use at the $j + 1^{th}$ step, and Eqns. 16 and 17 have to be integrated alternately. With Carling's Eqn. 18, iteration is needed within each step so that control of the process becomes rather complicated. However, it has been found that if Eqn. 18 is changed to

$$\mu_1(T_{j-1} - W_{j-1}) + \mu_2(\theta_{j-1} - W_{j-1}) - \lambda\left(\frac{W_j - W_{j-1}}{\Delta t}\right) = 0 \quad (19)$$

the calculations at each time step are direct and not iterative so that the hybrid program is much simplified. Since the stability of this version of the method has been proved, both theoretically and by practical trial, a short account is given.

Equation 16 has to be computed backwards so that the substitution

$$x = L - y \quad (20)$$

is made in Eqn. 16 only, and all three equations are rearranged; bars over certain variables denote functions which are being re-played from high-speed function stores.

$$\frac{dT_j}{dy} + \left(k_1 + \frac{1}{v_1\Delta t}\right)T_j = k_1\bar{W}_j + \frac{1}{v_1\Delta t}\bar{T}_{j-1} \quad (21)$$

$$W_j = \left[1 - \frac{\Delta t}{\lambda}(\mu_1 + \mu_2)\right]\bar{W}_{j-1} + \frac{\Delta t}{\lambda}\mu_1\bar{T}_{j-1} + \frac{\Delta t}{\lambda}\mu_2\bar{\theta}_{j-1} \quad (22)$$

$$\frac{d\theta_j}{dx} + \left(k_2 + \frac{1}{v_2\Delta t}\right)\theta_j = k_2W_j + \frac{1}{v_2\Delta t}\bar{\theta}_{j-1} \quad (23)$$

Both integrations now proceed from 0 to L, and y and x are represented (in turn) on the analogue computer by $\beta\tau$ where τ is computer time and β a time-scaling factor.

Referring now to the computer diagram in Fig. 2, integration of Eqns. 21–23 proceeds as follows. Assume that integration along y and x at problem-time $t = j\Delta t$ is about to begin. Stores 1, 2 and 3 are already loaded respectively with $\bar{T}_{j-1}$, $\bar{W}_{j-1}$ and $\bar{\theta}_{j-1}$ produced during the previous time step. Eqns. 22 and 23 are run on the analogue using summer S and integrator I_2 integrating with respect to x from 0 to L. The new functions W_j and θ_j are written in the stores during this run but $\bar{T}_{j-1}$ is left unchanged. The next stage of the process is to integrate Eqn. 21 with respect to y from 0 to L using $\bar{T}_{j-1}$ from store 1 and $\bar{W}_j$ from store 2 as forcing functions to integrator I_1, noting that both these functions must be read out of the stores in *backwards* order because integration is with respect to y, not x. During this process, the solution T_j gradually replaces $\bar{T}_{j-1}$ in store 1. $\bar{W}_j$ is left in store 2 so that when the two stages of the j^{th} step are completed, all three stores have been up-dated and the process can be repeated for the $j + 1^{th}$ step, etc.

An error analysis has shown the process to be stable in the sense that an error introduced during one time step decays through the subsequent time steps and does not affect the final steady-state temperatures (assuming the problem to be one which has a steady state). If Δt is decreased, a single error decays by the same amount in a given interval of problem time. Consequently, errors such as amplifier bias or drift, which are introduced at every step and act in the same sense, will accumulate to some extent and may shift an observed steady state from its true value. This phenomenon was observed in preliminary runs using a relay-controlled computer with transistor digital logic and some rather inaccurate function storage equipment. With this, it was established that there is an optimum value of Δt in order to balance truncation errors against accumulated drift etc., and with a modern hybrid computer with function storage equipment producing errors of the order of 0·1%, we expect our version of Carling's scheme to produce solutions with steady state errors not exceeding 0·5%.

Parallel-logic control

In the hybrid solution of partial differential equations, it is necessary to control a number of integrators through 'reset', 'compute' and 'hold' states in different sequences, and to control function stores and switches, as the examples in this paper have shown. Control can be achieved via the digital computer program in the case of a full hybrid system, or by means of the parallel logic associated with the analogue computer.

The method developed at Bath of using the parallel logic is simple, systematic and readily checked; it differs from that usually used on the commercially available hybrid computers but could be used with them. It is based on an *n*-stage ring counter (**Fig. 3** shows $n = 9$) holding a pattern of $n - 1$ 'zeros' and a single 'one', which latter jumps from stage to stage on receipt of the leading edge of a clock pulse; this pattern defines the *n* stages of the control sequence. The outputs from the ring counter are de-coded to operate the integrators and switches in the required sequence. **Fig. 4** shows a typical integrator control table in which the logic signals R and H define the state of the integrator. **Table 2** shows a sequence of $n = 9$ states which might be required of a particular integrator, together with the corresponding values of R and H copied from Fig. 4. After inspecting the sequence of values of R and of H, Boolean statements for R and H are devised (see Table 2) and translated into hardware in **Fig. 5**. This process is carried out for every device which has to be controlled from the ring counter, until the decoder of Fig. 3 is complete.

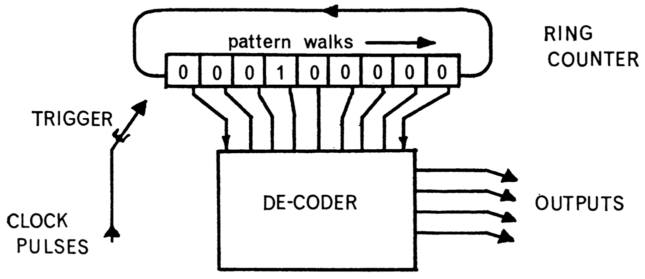
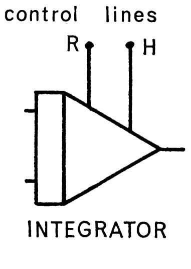


Fig. 3. Control system based on a ring counter



R	H	STATE
0	0	compute
0	1	hold
1	0	reset

Fig. 4. Integrator controls

Table 2

Integrator control logic table

STAGE NO. ON RING	INTEGRATOR STATE REQUIRED	*R*	*H*
1	Reset	1	0
2	Hold	0	1
3	Compute	0	0
4, 5	Hold	0	1
6, 7, 8, 9	Compute	0	0

Logic statements: $R =$ (stage 1)
$H =$ (stage 2) **or** (stage 4) **or** (stage 5)

Corresponding Hardware: Fig. 5.

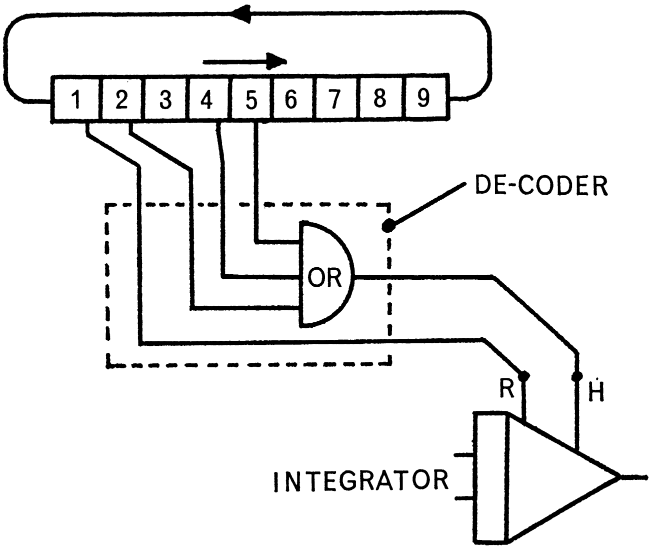


Fig. 5. Sequence control of one integrator

As it stands, each stage of the control sequence lasts for the same time interval, set by the clock pulses, and although this is satisfactory for most stages it is unlikely to be so for all. **Fig. 6** shows how, by introducing extra hardware, times which are an integral number of clock pulses can be obtained. Even more usefully, the adjustable timers which are available on analogue computers can be employed to stretch the times spent on particular

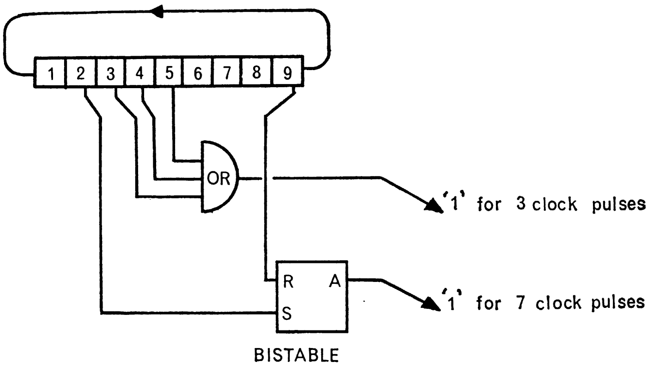


Fig. 6. Control signals which last for several clock pulses

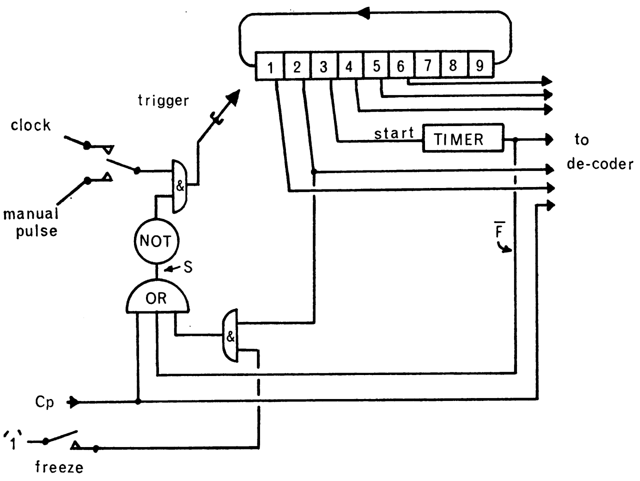


Fig. 7. Ring counter combined with adjustable timer and comparator signal

stages, as follows. As the clock pulse shifts the 'one' to stage 3 of the ring counter (**Fig. 7**), the output from that stage starts the timer which, in turn, inhibits the clock pulse input until the set time has elapsed. The next clock pulse is then allowed to move the 'one' to stage 4 on the ring. As a further example, when stage 6 is reached, suppose that some integrators have to compute until a particular analogue variable changes sign. It is simple to use a comparator to produce a logic signal, Cp on Fig. 7, which is 'one' during this time and switches smartly to 'zero' at the end. This signal can, as shown, inhibit the clock to give the appropriate time on stage 6.

For testing the logic, the free-running clock can be replaced by a single-shot (MANUAL) pulse, and another useful feature is a manually applied FREEZE signal which causes the computation to stop at some particular stage (stage 2 on Fig. 7) at which the integrators will all be in HOLD. Not shown on Fig. 7 is a means of obtaining a SET state in which potentiometers and initial integrator values can be set, and there are some technical details omitted which ensure that the inhibit signals can only open or shut the clock gate *in between* the clock pulses.

Some of the points made above will still be relevant when a digital computer is made to control the analogue integrators and switches.

Acknowledgements

We wish to thank A. S. Charlesworth and the computer technicians led by E. Perkins for assistance at Bath University on many occasions; also K. Jones of Electronic Associates Ltd. in connection with the computing runs on the heat flow problem; and the Science Research Council for financial assistance to T. I. Silvey.

References

Butler, R., Lloyd, E. C., Michel, J. G. L., and Sully, E. D. (1962). *A.I.C.A.*, No. 1, pp. 15–26.

Carling, L. N. (1968). *A.I.C.A.*, No. 1, pp. 6–18.

Copple, C., Hartree, D. R., Porter, A., and Tyson, H. (1939). *J. Inst. Elec. Engrs.*, Vol. 85, p. 56.

Eyres, N. R., Hartree, D. R., Ingham, J., Jackson, R., Sargent, R. J., and Wagstaff, J. B. (1946). *Phil. Trans. Roy. Soc.*, A, Vol. 240, pp. 1–57.

Hartree, D. R. (1950). *Calculating Instruments and Machines*, Cambridge University Press.

Hartree, D. R., and Womersley, J. R. (1937). *Proc. Roy. Soc.*, A, Vol. 161, pp. 353–366.

MacKay, D. M., and Fisher, M. E. (1962). *Analogue Computing at Ultra-high Speed*, Chapman and Hall.

Mitchell, E. E. L. (1963). S.A.G. Report No. 19. Electronic Associates Inc., Princeton, N.J.

Miura, T., and Iwata, J. (1966). *Simulation*, Vol. 6, pp. 105–108.

Silvey, T. I. (1966). M.Sc. Thesis. Bath University of Technology.

Vichnevetsky, R. (1968). Proc. Spring Joint Computer Conference, 1968. I.F.I.P. Congress, Edinburgh 1968. (Both in the press.)

Book review

Automatic Programming, Vol. 6, part 3, by P. Lucas and K. Walk, 1969; 182 pages. (*Pergamon Press Ltd.*, £2.00)

Part 3 of this volume contains just one paper: 'On the formal description of PL/1' by P. Lucas and K. Walk.

'This paper presents the tools and the design criteria for the formal description of programming languages. The results reported were achieved mainly during the development of the formal definition of PL/1 as documented in a series of technical reports (published by IBM Laboratory, Vienna). An appropriately tailored subset of PL/1 is used to illustrate these results. Their applicability is, however, not restricted to PL/1.' (quoted from first paragraph).

The paper must be heartily welcomed as the first public manifestation of the valuable and original work which has been carried out over a period of some five years in the IBM Laboratory at Vienna. It is recommended as a useful general overview of the subject, suitable as an introduction for a reader with some inclination towards formal studies and at least a nodding acquaintance with PL/1.

At the same time, it must be recognised that it is impossible to convey any full understanding of the properties of the 'Vienna' method and its application to PL/1 within the limits of a monograph. Like Gibbon's 'Decline and Fall', the true nature of the achievement can only be appreciated after completing the entire work in unabridged form. The reason for this is plain. The purpose of a formal language definition is to be complete—to fill all the 'gaps' that may be left in an informal description. A formalisation which appears to be partially complete can be wholly unsatisfying. And yet very few who embark on the complete work will ever finish it.

The main problem seems to me that the human reader does not take kindly to very large sets of definitions and axioms; he has an urgent need for worked examples and proofs. After all, even Gibbon has his jokes, and Shakespeare his comic relief. It is much to be hoped that the authors will have an opportunity to present their work more completely, with more concession to human weakness, between the covers of a book.

C. A. R. Hoare (Belfast)