Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science

W.H. Hundsdorfer, B.P. Sommeijer, J.G. Verwer

A numerical study of a 1D-scalar semi-conductor model equation

Department of Numerical Mathematics Note NM-N8602 August

A Numerical Study of a 1D-Scalar Semi-Conductor Model Equation

W.H. Hundsdorfer, B.P. Sommeijer, J.G. Verwer
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

This note reports on an investigation to the performances of integration techniques currently in use in numerical simulation programs for the time-dependent semi- conductor equations. The subject of investigation is a simple 1*D*-scalar equation, a convection-diffusion equation, which was derived from equations modelling a simple diode in such a way that some important characteristic features were maintained.

1980 Mathematics Subject Classification: 65M20, 65M99.
Note: Contract research carried out by order of Drs. S.J. Polak, Nederlandse Philips Bedrijven B.V., ISA-ISC-TIS/CARO. Gebouw SAQ 2, 5600 MD Eindhoven, The Netherlands.

Contents

1. A DESCRIPTION OF THE MODEL PROBLEM

(Based on a note received from Dr. C. den Heyer, Philips; November 7, 1985).

The model problem is the convection-diffusion equation

$$p_t = (\mu\alpha^{-1}p_x + \mu p\psi_x)_x,\ x\in I = [0,x_e],\ t>0\ , \tag{1.1a}$$

with the initial condition

$$p(x,0) = 10^{20},\ x\in I\ , \tag{1.1b}$$

and the boundary conditions

$$p(0,t) = 10^{20},\ t>0\ , \tag{1.1c}$$

$$p_x(x_e,t) = 0,\ t>0\ .$$

The parameter values are

$$\mu = 500,\ \alpha = 38.6,\ x_e = 10^{-3}\ , \tag{1.2}$$

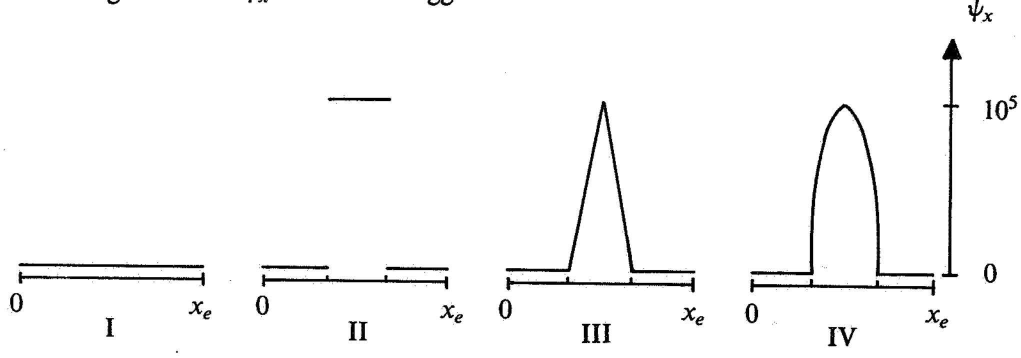
and $\psi = \psi(x)$ satisfies

$$\psi_x(x) = 0,\quad x = 0,x_e\ .$$

$$\psi_x(0.5x_e - x) = \psi_x(0.5x_e + x)\ ,\quad \text{(symmetry around } x = x_e/2) \tag{1.3}$$

$$\psi_x(x)\in[0,10^5] \text{ for } x\in I .$$

The following functions ψ_x have been suggested:



The problem (1.1) is known to possess a solution of exponential type, in particular sharp layers occur. In an attempt to annihilate most of the exponential layers, one often considers a transformation of variables $p\rightarrow\phi$ (in the general semi-conductor equation, ϕ is the quasi-Fermilevel for holes)

$$p = n_i e^{\alpha(\phi-\psi)} , \tag{1.4a}$$

where, in our example, n_i is a constant given by

$$n_i = 1.2_{10}10 . \tag{1.4b}$$

The equation (1.1a), i.e.,

$$p_t = J_x \quad \text{with} \quad J = \mu\alpha^{-1}p_x + \mu p\psi_x ,$$

then transforms into

$$p\alpha(\phi_t-\psi_t) = J_x \quad \text{with} \quad J = \mu p\phi_x \cdot$$

or, as here $\psi_t = 0$,

$$p\alpha\phi_t = J_x \text{ with } J = \mu p\phi_x \text{ and } p = n_i e^{\alpha(\phi-\psi)} . \tag{1.5}$$

So this equation, with the given boundary and initial conditions, replaces (1.1a). It is expected that the solution ϕ is easier to approximate than the solution p in (1.1a). Note that (1.5) is strongly non-linear. One of the points to be examined is to find whether the anticipated expectation on the transformation (1.4a) comes out numerically. The following lemma plays a role here:

Lemma. *Let $p,\tilde{p}$ and $\phi,\tilde{\phi}$ be any given two solutions (corresponding to two different initial functions). Then, at any point (x,t) of interest,*

$$\left|\frac{\tilde{p}-p}{p}\right|\leqslant\epsilon \Leftrightarrow |\tilde{\phi}-\phi| \leqslant\frac{1}{\alpha}(\epsilon+0(\epsilon^2)),\ \epsilon\rightarrow 0 . \tag{1.6}$$

Proof. From equation (1.4a) we find

$$\frac{\tilde{p}-p}{p} = e^{\alpha(\tilde{\phi}-\phi)}-1 . \tag{1.7}$$

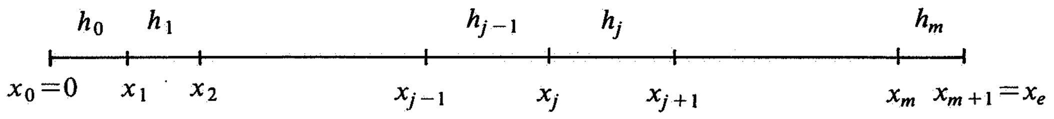
Now consider the condition $|e^{\alpha a}-1|\leqslant\epsilon, a\in\mathbb{R}$. If $a\geqslant 0$, this holds iff $e^{\alpha a}\leqslant 1+\epsilon$ or $a\leqslant\alpha^{-1}\log(1+\epsilon)=\alpha^{-1}(\epsilon+0(\epsilon^2))$. If $a\leqslant 0$, the condition holds iff $e^{\alpha a}\geqslant 1-\epsilon$ which is equivalent to $\alpha a\geqslant\log(1-\epsilon)$ or $|a|\leqslant-\alpha^{-1}\log(1-\epsilon)=\alpha^{-1}(\epsilon+0(\epsilon^2))$. If we replace $\tilde{\phi}-\phi$ in (1.6), (1.7) by the real number a, the proof is complete. □

This lemma shows that a perturbation of ϕ in the usual absolute sense causes a perturbation in p in a relative sense. Numerically it means that if we work in (ψ,ϕ) variables and require *relative errors in* p less than ϵ (this seems natural as p mostly is extremely large), that then an *absolute error in* ϕ is required less than ϵ/α. In the numerical investigation of the use of the transformation (1.4a), the above observation must be reckoned with.

2. The spatial discretization (cf. den Heyer's note)

We introduce the nonequidistant grid

$$\Delta_h = \{x_j : x_0 = 0, x_{j+1} = x_j + h_j \; (j=0(1)m), \; x_{m+1} = x_e\} .$$



For convenience later on we write $\psi'=\psi_x$. At $x=x_j$ we consider the following discretization of (recall that $p_t=J_x$).

$$J_x \equiv (\mu\alpha^{-1}p_x + \mu p\psi')_x . \tag{2.1}$$

At x_j, J_x is replaced by its discrete counterpart

$$\frac{J_{j+1/2}-J_{j-1/2}}{(h_j+h_{j-1})/2} \tag{2.2}$$

so that, for $j=1(1)m$, the *semi-discrete equations* are (capital P is the semi-discrete approximation for p)

$$\dot{P}_j = \frac{J_{j+1/2}-J_{j-1/2}}{(h_j+h_{j-1})/2} . \tag{2.3}$$

The next step is to express $J_{j\pm 1/2}$ in terms of $P_{j\pm1}, P_j$. This is done in a special way (Scharfetter-Gummel, exponential fitting [4, 5]).

Consider the equation

$$J = \mu\alpha^{-1}p_x + \mu p\psi' \tag{2.4}$$

on the interval $[x_j, x_{j+1}]$. We approximate (2.4) on $[x_j, x_{j+1}]$ by

$$J_{j+1/2} = \mu\alpha^{-1}p_x + \mu\psi'_{j+1/2}p \qquad (J_{j+1/2}, \psi'_{j+1/2} \text{ are constant}) \tag{2.5}$$

and now integrate this equation analytically on $[x_j, x_{j+1}]$. We write

$$J_{j+1/2}\alpha\mu^{-1} = p_x + \alpha\psi'_{j+1/2}p$$

which in turn is rewritten to

$$J_{j+1/2}\alpha\mu^{-1}e^{\alpha\psi'_{j+1/2}x} = (e^{\alpha\psi'_{j+1/2}x}p)_x .$$

Integration yields

$$\frac{J_{j+1/2}}{\mu\psi'_{j+1/2}}[e^{\alpha\psi'_{j+1/2}x}]_{x_j}^{x_{j+1}} = [e^{\alpha\psi'_{j+1/2}x}p]_{x_j}^{x_{j+1}} ,$$

which is the same as (we already use the semi-discrete notation for p)

$$\frac{J_{j+1/2}}{\mu\psi'_{j+1/2}}[e^{\alpha\psi'_{j+1/2}x}]_0^{h_j} = [e^{\alpha\psi'_{j+1/2}h_j}P_{j+1} - P_j] .$$

Finally, we get

$$J_{j+½} = \mu\psi'_{j+½}\,\frac{(e^{\alpha\psi'_{j+½}h_j}P_{j+1}-P_j)}{(e^{\alpha\psi'_{j+½}h_j}-1)} \tag{2.6a}$$

and, likewise,

$$J_{j-½} = \mu\psi'_{j-½}\,\frac{(e^{\alpha\psi'_{j-½}h_{j-1}}P_j-P_{j-1})}{(e^{\alpha\psi'_{j-½}h_{j-1}}-1)}\,. \tag{2.6b}$$

Before inserting these expressions into (2.3) we still modify them slightly by replacing

$$\psi'_{j+½}\simeq\Delta\psi_{j+½}/h_j,\ \Delta\psi_{j+½} = \psi_{j+1}-\psi_j\,. \tag{2.6c}$$

It has turned out that this yields a somewhat better shock positioning and, furthermore, this replacement naturally occurs if we derive the semi-discretization of the transformed system (1.5). Thus we find

$$J_{j+½} = \frac{\mu\Delta\psi_{j+½}}{h_j}\,\frac{(e^{\alpha\Delta\psi_{j+½}}P_{j+1}-P_j)}{(e^{\alpha\Delta\psi_{j+½}}-1)}\,, \tag{2.6a'}$$

and, likewise,

$$J_{j-½} = \frac{\mu\Delta\psi_{j-½}}{h_{j-1}}\,\frac{(e^{\alpha\Delta\psi_{j-½}}P_j-P_{j-1})}{(e^{\alpha\Delta\psi_{j-½}}-1)}\,. \tag{2.6b'}$$

These expressions are substituted into (2.3) to obtain

$$\dot{P}_j = (\frac{2}{h_j+h_{j-1}})\,(A_{jj-1}P_{j-1} + A_{jj}P_j + A_{jj+1}P_{j+1}), \tag{2.7}$$

where

$$A_{jj-1} = \frac{\mu\Delta\psi_{j-½}}{h_{j-1}}\,(e^{\alpha\Delta\psi_{j-½}}-1)^{-1}\,, \tag{2.8a}$$

$$A_{jj+1} = \frac{\mu\Delta\psi_{j+½}}{h_j}\,(e^{\alpha\Delta\psi_{j+½}}-1)^{-1}e^{\alpha\Delta\psi_{j+½}}\,, \tag{2.8b}$$

$$A_{jj} = -\frac{\mu\Delta\psi_{j+½}}{h_j}(e^{\alpha\Delta\psi_{j+½}}-1)^{-1} - \frac{\mu\Delta\psi_{j-½}}{h_{j-1}}\,(e^{\alpha\Delta\psi_{j-½}}-1)^{-1}e^{\alpha\Delta\psi_{j-½}}\,, \tag{2.8c}$$

and $j=1(1)m$. For $j=1$ we insert the known left boundary value for $P_0(t)$ (cf. (1.1c)). For $j=m$ we have to incorporate the right boundary condition $P_x(x_e,t)=0$. We insert the simple replacement $P_{m+1}=P_m$. Hence, we then arrive at the *continuous time, semi-discrete system*

$$\dot{P} = AP + B\,,\quad t>0\,,\quad P(0)\text{ given}, \tag{2.9}$$

with A the (non-symmetric) $m\times m$-matrix formed above (recall the change for the first and last row due to boundary conditions) and B the m-vector

$$B = [\frac{2}{h_0+h_1}A_{10}P_0(t),\,0,\,\cdots,0]^T\,. \tag{2.10}$$

Since $P_0(t)$ is a constant B is independent of t.

Remarks 1° . Note that when $\alpha\Delta\psi>>1$ we have, approximately a forward difference in space ($A_{jj-1}\simeq0$). If $\alpha\Delta\psi<<-1$ we have approximately a backward difference in space ($A_{jj+1}\simeq0$). This means the scheme is adapted (locally) to the stream direction.
2° In case $\alpha\Delta\psi\simeq0$, the expressions (2.8) must be rewritten in order to avoid cancellation or zero

division. The largest possible value for $\alpha\Delta\psi$ is 3,860,000h. This value must lie in the exponent range of our Cyber. We get no difficulties if $5h << x_e = 10^{-3}$. □

We next consider *the* (ψ,ϕ) *formulation*, i.e., the transformation (1.4a). Following the note of C. den Heyer, we work the transformation directly into the semi-discretization (2.9) rather than starting from scratch with (1.5). Note that the semi-discrete variable for $\phi(x_j,t), \Phi_j$ say, satisfies

$$P_j = n_i e^{\alpha(\Phi_j - \psi_j)} \tag{2.11}$$

and thus (ψ is here independent of t)

$$\dot{P}_j = \alpha P_j \dot{\Phi}_j \,. \tag{2.12}$$

Now consider the equation (2.3), with $J_{j\pm½}$ given by (2.6a'b'). Let us take $J_{j+½}$. We rewrite its expression to

$$J_{j+½} = \frac{\mu\alpha\Delta\psi_{j+½}}{\alpha h_j} \frac{(e^{\alpha\Delta\psi_{j+½}} - P_j/P_{j+1})}{(e^{\alpha\Delta\psi_{j+½}} - 1)} P_{j+1}. \tag{2.13}$$

Substituting $P_j/P_{j+1} = \exp(\alpha(\Phi_j - \Phi_{j+1}) - \alpha(\psi_j - \psi_{j+1}))$ then leads to

$$J_{j+½} = \frac{\mu P_{j+1}}{\alpha h_j} \frac{\alpha\Delta\psi_{j+½}}{1 - e^{-\alpha\Delta\psi_{j+½}}} (1 - e^{-\alpha\Delta\Phi_{j+½}}) \,, \tag{2.14a}$$

where $\Delta\psi_{j+½} = \psi_{j+1} - \psi_j$, $\Delta\Phi_{j+½} = \Phi_{j+1} - \Phi_j$. Likewise we get

$$J_{j-½} = \frac{\mu P_j}{\alpha h_{j-1}} \frac{\alpha\Delta\psi_{j-½}}{1 - e^{-\alpha\Delta\psi_{j-½}}} (1 - e^{-\alpha\Delta\Phi_{j-½}}) \,. \tag{2.14b}$$

Substitution of (2.12), (2.14 ab) into (2.3) yields the semi-discretization for the (ψ,ϕ)-formulation:

$$\dot{\Phi}_j = \frac{2}{(h_j + h_{j-1})} (\gamma_j^{(1)} \gamma_j^{(2)} (e^{\alpha(\Phi_{j+1} - \Phi_j)} - 1) + \gamma_{j-1}^{(1)} (e^{\alpha(\Phi_{j-1} - \Phi_j)} - 1)) \,, \tag{2.15}$$

where

$$\gamma_j^{(1)} = \frac{\mu}{\alpha h_j} \frac{\Delta\psi_{j+½}}{1 - e^{-\alpha\Delta\psi_{j+½}}} \,, \quad \gamma_j^{(2)} = e^{-\alpha\Delta\psi_{j+½}} \,. \tag{2.16}$$

We emphasize that the semi-discrete solutions $P_j(t)$ of system (2.7) and $\Phi_j(t)$ of system (2.15) are completely identical via relation (2.11).

REMARKS 1. If $\Delta\psi_{j-½} \simeq 0$ the danger of cancellation or even zero division arises. In this case we must rewrite the expression $z/(1 - e^{-z})$, $z \simeq 0$.

2. In the derivation above it has been tacitly assumed that the differences $\Delta\psi$ are nonnegative, which in our example is always true since $\Delta\psi \simeq h\psi'$ and $\psi' \geq 0$ (see p. 2). However, in situations where $\psi' < 0$ it is recommended to use the alternative expression for J which arises by dividing through P_j in equation (2.13). Otherwise the transformation may "kill" the expressions for J since $z/(1 - e^{-z})$ will nearly vanish for $z << -1$.

3. An interesting point to notice is that the semi-discrete system (2.15) is stable in l^∞, i.e., any two solutions Φ and $\tilde{\Phi}$ (corresponding to two different initial functions) satisfy the usual stability inequality

$$\|\Phi(t+\tau) - \tilde{\Phi}(t+\tau)\|_\infty \leq \|\Phi(t) - \tilde{\Phi}(t)\|_\infty \,, \quad \forall t, \tau \geq 0 \,. \tag{2.17}$$

This can be seen by computing the Jacobian matrix of this system and, subsequently, by observing that the logarithmic matrix norm μ_∞ satisfies

$$\mu_\infty \leqslant 0, \ \forall \Phi, \tilde{\Phi} \in \mathbb{R}^m . \tag{2.18}$$

Application of the backward Euler scheme to (2.15) then guarantees unconditional (any $\tau > 0$) numerical stability in l^∞. These results on stability are extensively discussed in [2], Sections 1.5, 2.4. □

3. Some remarks on a relevant conservation law

Consider the integral (conservation of charge integral)

$$c(t) = \int_0^{x_e} p(x,t)dx . \tag{3.1}$$

One is interested in a correct simulation in time of

$$\dot{c}(t) = \int_0^{x_e} (\mu\alpha^{-1}p_x + \mu p\psi_x)_x dx = \mu\alpha^{-1}p_x + \mu p \psi_x |_0^{x_e} = -\mu\alpha^{-1}p_x(0,t) . \tag{3.2}$$

It follows that if $p_x(0,t)=0$, we have conservation of charge or, in mathematical terms, conservation in l^1. In passing we note that if $p_x(0,t) \geqslant 0$, the PDE problem is stable in l^1.

For the semi-discrete PDE we consider the quadrature approximation $C(t)$ of $c(t)$ given by

$$C(t) = \sum_{j=1}^{m} (\frac{h_j + h_{j-1}}{2}) P_j(t) \tag{3.3}$$

for which

$$\dot{C}(t) = \sum_{j=1}^{m} (A_{jj-1}P_{j-1} + A_{jj}P_j + A_{jj+1}P_{j+1}) . \tag{3.4}$$

Using $A_{j-1j} + A_{jj} + A_{j+1j} = 0$ for all j (zero column sum), a simple calculation yields

$$\begin{aligned} \dot{C}(t) &= (A_{10}P_0 - A_{01}P_1) + (A_{mm+1}P_{m+1} - A_{m+1m}P_m) \\ &= (A_{10}P_0 - A_{01}P_1) + (A_{mm+1} - A_{m+1m})P_m \end{aligned} \tag{3.5}$$

since $P_{m+1} = P_m$ (discrete von Neumann condition). We note that

$$A_{mm+1} - A_{m+1m} = \mu\frac{\Delta\psi_{m+½}}{h_m} (e^{\alpha\Delta\psi_{m+½}} - 1)^{-1} (e^{\alpha\Delta\psi_{m+½}} - 1) = \mu\frac{\Delta\psi_{m+½}}{h_m} .$$

In conclusion, if $\Delta\psi_{m+1/2} = 0$, and this is true for the functions ψ' under examination, then

$$\dot{C}(t) = A_{10}P_0 - A_{01}P_1 . \tag{3.6}$$

If $\Delta\psi_{½} = 0$, then $A_{01} - A_{10} = 0$ (as above for $A_{mm+1} - A_{m+1m}$) and we may write

$$\dot{C}(t) = A_{10}(P_0 - P_1) . \tag{3.7}$$

As $\Delta\psi_{½} = 0$, there holds

$$A_{10} = \mu\frac{\Delta\psi_{½}}{h_0} (\alpha\Delta\psi_{½} + \frac{1}{2}\alpha^2(\Delta\psi_{½})^2 + \cdots)^{-1} = \mu\alpha^{-1}h_0^{-1} ,$$

so that, finally,

$$\dot{C}(t) = -\mu\alpha^{-1}(P_1(t) - P_0(t))/h_0 . \tag{3.8}$$

This result is in agreement with (3.2) for the fully continuous problem. It follows that if $P_0(t) = P_1(t)$ (this condition can be enforced in the scheme in case of a von Neumann condition at $x=0$) that then the semi-discretization conserves the quantity $C(t)$ given by (3.3) exactly. In our tests attention will be paid as to how well $C(t)$ will be approximated. In this connection we emphasize that in the (ψ,ϕ)-formulation the above derivation does not apply.

4. A DISCUSSION OF THE POINTS TO BE INVESTIGATED

(This section reproduces the proposal made by us to Dr. W. Schilders, Philips, at Dec. 18, 1985).

For the time-integration of the two semi-discretizations ((ψ,p)-formulation and (ψ,ϕ)-formulation) we will employ the familiar backward Euler scheme (notation $\dot{y}=f(t,y)$)

$$y^{n+1} = y^n + \tau f(t_{n+1},y^{n+1}) \tag{4.1}$$

and the second-order combination of TR and BDF_2

$$y^{n+\gamma} = y^n + \frac{1}{2}\gamma\tau f(t_n,y^n) + \frac{1}{2}\gamma\tau f(t_{n+\gamma},y^{n+\gamma}), \quad \gamma = 2-\sqrt{2} \tag{4.2}$$

$$y^{n+1} = \frac{-1}{\gamma(\gamma-2)}y^{n+\gamma} + \frac{(\gamma-1)^2}{\gamma(\gamma-2)}y^n + \frac{\gamma-1}{\gamma-2}\tau f(t_{n+1},y^{n+1})$$

proposed in [1]. Note that (4.2) is a diagonally implicit RK scheme.

The following points will be specifically investigated by *numerical experimentation:*

1°. Does the use of the more complicated (ψ,ϕ)-formulation pay off in terms of accuracy/efficiency and computation of $C(t)$?
2°. Which of the two schemes (4.1), (4.2) performs better, for both the (ψ,p) and (ψ,ϕ)-formulation?
3°. Is it possible to correct for the anticipated loss of accuracy in the computation of $C(t)$, when using the (ψ,ϕ)-formulation, in a simple and not too expensive way (this point will be considered only if it seems relevant enough in relation to 1° and 2°)?

Clearly, the above questions must be considered for semi-discretizations with a *small enough spatial error.* Because the solutions of three of the problems (1.1) are not known in advance, we must first determine the grids on which the fully continuous solutions can be represented up to a reasonable accuracy by the semi-discrete ones. For this purpose we shall employ the BDF code GEARB [3] (in a slightly adapted version).

5. THE NUMERICAL INVESTIGATION

The numerical investigation has been restricted to the set of 4 problems introduced in Section 1. The corresponding ψ functions are

Problem I: $\psi(x) = c$

Problem II: $$\psi(x) = \begin{cases} c & , x\in[0,1/3\ x_e] \\ c+10^5(x-1/3\ x_e), & x\in[1/3\ x_e,2/3\ x_e] \\ c+1/3\ 10^5 x_e & , x\in[2/3\ x_e,x_e] \end{cases}$$

Problem III: $$\psi(x) = \begin{cases} c & , x\in[0,1/3\ x_e] \\ c+10^5((3x/x_e-2)x+1/3\ x_e) & , x\in[1/3\ x_e,1/2\ x_e] \\ c+1/12\ 10^5 x_e+10^5((-3x/x_e+4)x-5/4\ x_e), & x\in[1/2\ x_e,2/3\ x_e] \\ c+1/6\ 10^5 x_e & , x\in[2/3\ x_e,x_e] \end{cases}$$

Problem IV: $$\psi(x) = \begin{cases} c & , x\in[0,1/3\ x_e] \\ c+10^5(((-12x/x_e+18)x/x_e-8)x+10/9\ x_e), & x\in[1/3\ x_e,\ 2/3\ x_e] \\ c+2/9\ 10^5 x_e & , x\in[2/3\ x_e,x_e] \end{cases}$$

Here c is an arbitrary constant which does not enter into the *PDE* (only ψ_x contributes). Hence problem I reads $p_t = \mu\alpha^{-1} p_{xx}$ together with the initial and boundary conditions specified in Section 1. Its solution is $p(x,t) \equiv 10^{20}$ so that only problems II-IV remain to be dealt with. However, despite its triviality, problem I is of some interest from a numerical point of view in connection with the threat of round-off. We shall discuss this in Section 5.1. The remaining subsections shall then only deal with problems II-IV.

5.1. Some remarks on round-off

Consider the constant coefficient, linear *ODE* problem (2.9). The backward Euler scheme for this problem reads

$$P^{n+1} = P^n + \tau A P^{n+1} + \tau B\,. \tag{5.1}$$

In a practical computation A,B usually contain round-off errors. To analyse the effects of round-off errors, we have to consider the perturbed scheme

$$\tilde{P}^{n+1} = \tilde{P}^n + \tau \tilde{A} \tilde{P}^{n+1} + \tau \tilde{B} + D^{n+1}\,. \tag{5.2}$$

Here $\tilde{A},\tilde{B}$ represent the perturbed A,B and D^{n+1} represents errors which, e.g., may arise in the numerical solution of the systems of linear algebraic equations. If we denote $\Delta A = \tilde{A} - A$ and $\Delta B = \tilde{B} - B$, and use

$$(I-\tau\tilde{A})^{-1} - (I-\tau A)^{-1} = (I-\tau A)^{-1}\tau\Delta A(I-\tau\tilde{A})^{-1}\,, \tag{5.3}$$

we obtain the following recursion for the round-off errors $\tilde{P}^n - P^n$:

$$\begin{aligned}\tilde{P}^{n+1} - P^{n+1} = {} & (I-\tau\tilde{A})^{-1}(\tilde{P}^n - P^n) + (I-\tau A)^{-1}\tau\Delta A(I-\tau\tilde{A})^{-1}P^n + \\ & (I-\tau\tilde{A})^{-1}(D^{n+1} + \tau\Delta B) + (I-\tau A)^{-1}\tau\Delta A(I-\tau\tilde{A})^{-1}\tau B\,.\end{aligned} \tag{5.4}$$

A rigorous analysis of this recursion falls outside the scope of our report. We therefore now discuss the following example which is of clear relevance to our problems I-IV, and in particular to problem I.

EXAMPLE. We consider only *one step* and set $\tilde{P}^n = P^n$. Furthermore, put $\Delta B = B = 0$ *(homogeneous problem*; this is the case if we have in problems I-IV Neumann boundary conditions at both endpoints) and ignore D^{n+1}. Recursion (5.4) is then given by

$$\tilde{P}^{n+1} - P^{n+1} = (I-\tau A)^{-1}\tau\Delta A(I-\tau\tilde{A})^{-1}P^n\,, \tag{5.5}$$

and hence

$$\|\tilde{P}^{n+1} - P^{n+1}\|_\infty \leqslant \|(I-\tau A)^{-1}\|_\infty \cdot \|\tau\Delta A\|_\infty \cdot \|(I-\tau\tilde{A})^{-1}\|_\infty \cdot \|P^n\|_\infty\,. \tag{5.6}$$

For A we take

$$A = ah^{-2}\begin{bmatrix} -1 & 1 & & & \\ 1 & -2 & 1 & & \\ & \cdot & \cdot & \cdot & \\ & & 1 & -2 & 1 \\ & & & 1 & -1 \end{bmatrix}_{m\times m}, \tag{5.7}$$

which is exactly the finite difference matrix of problem I if we replace its Dirichlet boundary condition $p(0,t) = 10^{20}$ by the Neumann condition $p_x(0,t) = 0$, and discretize it in the same way as at the right endpoint $x = x_e$. There holds (see [2], Section 1.5, 2.4)

$$\mu_\infty[A] = 0,\ \mu_\infty[\tilde{A}] \leqslant \mu_\infty[A] + \|\Delta A\|_\infty\,, \tag{5.8}$$

so that, for all $\tau>0$,

$$\|(I-\tau A)^{-1}\|_\infty \leq 1,\ \|(I-\tau\tilde{A})^{-1}\|_\infty \leq \frac{1}{1-\tau\|\Delta A\|_\infty},\ \tau\|\Delta A\|_\infty<1\,. \tag{5.9}$$

This simplifies (5.6) to

$$\|\tilde{P}^{n+1} - P^{n+1}\|_\infty \leq \frac{\tau\|\Delta A\|_\infty}{1-\tau\|\Delta A\|_\infty}\ \|P^n\|_\infty\,. \tag{5.10}$$

Next suppose that η is the machine precision (for our Cyber, $\eta\simeq 10^{-14}$), i.e., the entries $\tilde{a}_{ij},a_{ij}$ of $\tilde{A},A$ satisfy

$$|\tilde{a}_{ij}-a_{ij}|\leq\eta|a_{ij}|\,. \tag{5.11}$$

For matrix (5.7) we thus have $\|\Delta A\|_\infty\leq 4\eta a h^{-2}$, so that, finally, we arrive at the round-off error inequality

$$\|\tilde{P}^{n+1}-P^{n+1}\|_\infty \leq \frac{\tau\eta\sigma}{1-\tau\eta\sigma}\ \|P^n\|_\infty,\ \sigma = 4ah^{-2},\ \tau\eta\sigma<1\,. \tag{5.12}$$

The following lemma shows that if we allow arbitrary perturbations ΔA, the inequality (5.12) is sharp.

Lemma. *It is possible to have equality in (5.12).*

Proof. Let $P^n = ce, e=[1,\cdots,1]^T$, $c>0$ a constant (ce is, e.g., the (steady state) solution of problem I). Let $\Delta A = D_1AD_2$ with

$$D_1 = \tfrac{1}{2}\sqrt{\eta}\,diag(2,-1,1,\cdots,-1,2),\ D_2 = \sqrt{\eta}\,diag(-1,1,-1,\cdots,-1)\,, \tag{5.13}$$

where it is supposed that m is odd. There holds $Ae=0$, and $\tilde{A}e=\Delta Ae=\eta\sigma e$ where $\sigma=4ah^{-2}$. Hence e is an eigenvector for $A,\tilde{A}$ and ΔA. By using (5.5) it then directly follows that

$$\tilde{P}^{n+1}-P^{n+1} = \frac{\tau\eta\sigma}{1-\tau\eta\sigma}\ ce\,, \tag{5.14}$$

which proves the lemma. □

From the above we may conclude that (after one step) *the relative error* in P^{n+1}, due to round-off in A, may be as large as $\tau\eta\sigma/(1-\tau\eta\sigma)$. Since the entries in A may be extremely large, say about 10^{10}, this relative error may also become quite large. For example (cf. problem I),

$$a = \mu/\alpha = \frac{500}{38.6}\,,\ h = \frac{x_e}{50}\,,\ x_e = 10^{-3}\,,\ \eta = 10^{-14} \tag{5.15}$$

yields

$$\frac{\tau\eta\sigma}{1-\tau\eta\sigma} \simeq \frac{1.3_{10^{-3}}\tau}{1-1.3_{10^{-3}}\tau} \simeq 1.3_{10^{-3}}\tau\,. \tag{5.16}$$

In our applications τ is always below 10^{-7} so that the relative error due to round-off will play no role. However, for marching to steady state calculations, where τ is usually taken larger, the above analysis could be relevant. Computations with the (trivial) problem I have confirmed this.

5.2. The semi-discrete solutions (for problems II-IV)

For comparing the integration formulas (4.1), (4.2) we need accurate reference solutions of the semi-discretizations (2.9) and (2.15) for a suitable space grid. Note that if $\Phi_j(t)$, $1 \leqslant j \leqslant m$, has been computed, $P_j(t)$ is also known via relation (2.11). Since Φ is better scaled over the domain of interest we work here in Φ-variable. The reference solutions were computed with the well-known code *GEARB* [3] (BDF formulas of order 1 to 5) using tolerance values $TOL = 10^{-8}, 10^{-10}$. For all the experiments discussed in this section, this has resulted in an absolute difference in Φ_j (for these two *TOL* values) of less than 10^{-6}. We note that this is more than sufficient for our aim, even when taking into account the implication of the lemma in Section 1 (we discuss this point later). The accuracy range of interest for P_j is, in the relative sense, approximately 1% to 10%.

In order to select a suitable space grid - such that $P_j(t)$ represent $p(x_j,t)$ reasonably well - we started with a uniform space grid of size $h = x_e/50$, and simply doubled the number of points until we found a maximal relative difference between values on two successive grids of approximately 1% for all selected output points. These output points are $t = 10^{-12}(*10)10^{-7}$. It turned out, for all three problems, that our 1% - criterion was satisfied when comparing solutions for 400 and 800 grid points. There upon we have determined, again for three problems, one fixed non-uniform grid consisting of 146 points. This non-uniform grid was constructed by trial and error such that, when compared with the values on the uniform grid with 800 points, our error criterion of 1% was (approximately) satisfied. The grid points of the final non-uniform grid are given in Appendix 1. Note that around the points $x/x_e = 1/3, 1/2, 2/3$ the grid is finer. This is to be expected, of course, in view of the form of the functions ψ_x (see Section 1). However, during the experiments we also found it necessary to refine near $x = 0$ (the outflow boundary) and $x = x_e$ (isolated).

Summarizing, using *GEARB* we have computed accurate reference ODE solutions at the output times $t = 10^{-12}(*10)10^{-7}$. These solutions correspond to the non-uniform grid given in Appendix 1. For problem II we made plots of the solution, both for the P and Φ variable, with P and t on logarithmic scale. Figures 5.1a, 5.1b and 5.1c show the P-solution from three different sides. Since the (relative) errors in space are bounded by approximately 1%, one may interprete these plots as the solution of the PDE. Observe that the solution shows a shock-like behaviour (internal layer in space and time). The large variation starts if t approaches 10^{-8}. We note that numerically a steady state is reached within the interval $10^{-7} \leqslant t \leqslant 10^{-6}$. Also observe that in our domain of interest ($0 \leqslant x \leqslant 10^{-3}$, $0 \leqslant t \leqslant 10^{-7}$)$\min_j(P_j)$ and $\max_j(P_j)$ vary approximately between 10^5 and 10^{20}, which shows that relative error measuring in P is necessary. For comparison, Figures 5.2a and 5.2b show the solution of the second problem in Φ. Clearly, in Φ-variable the variation is much less, which was the aim of the transformation. Plots of the problems III and IV are omitted because they show a similar qualitative behaviour. Finally, Table 5.1 contains the (nearly) exact values $C(t)$ given by (3.3).

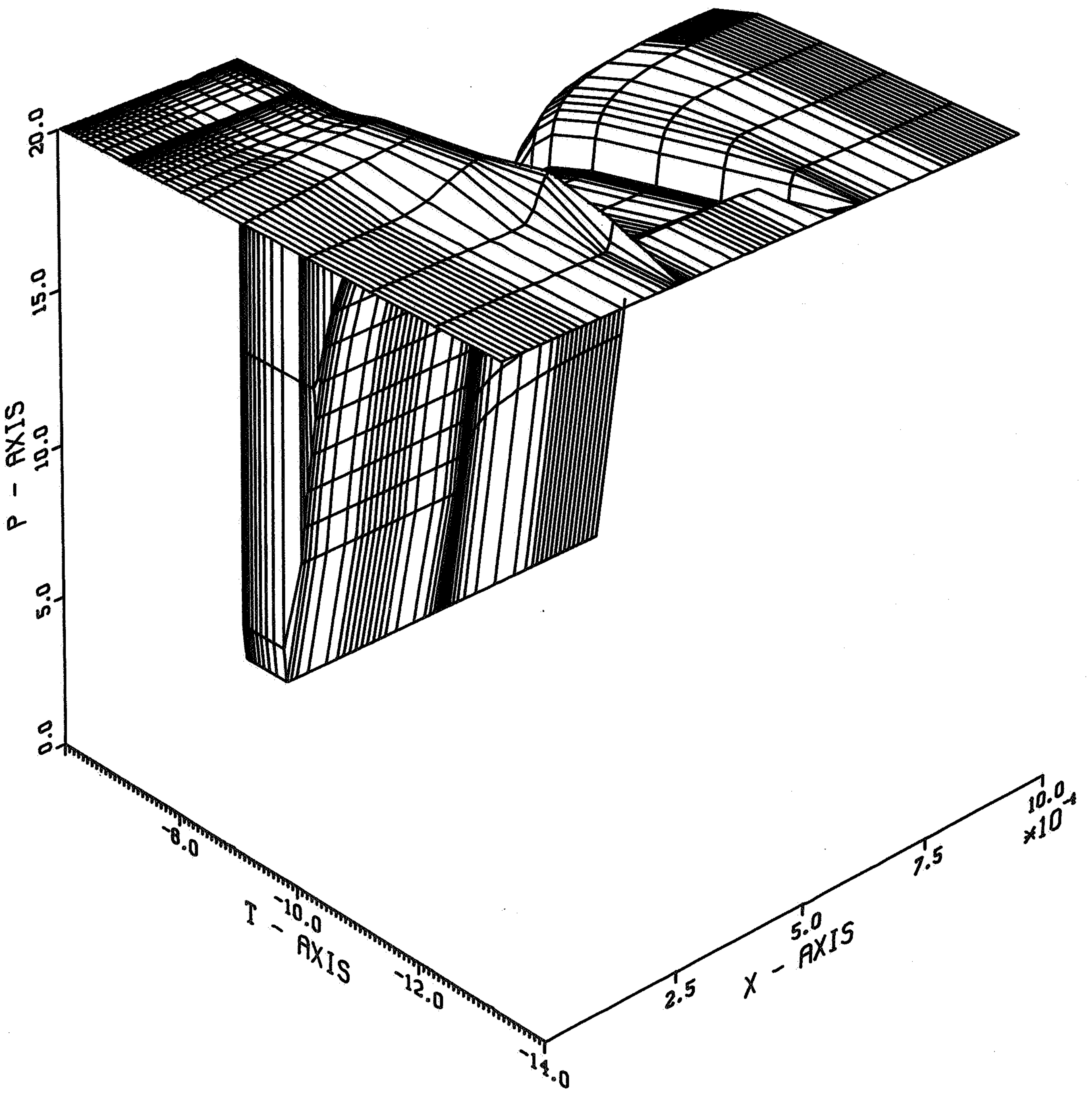


FIGURE 5.1a. The solution of problem II in (ψ,p)-formulation.

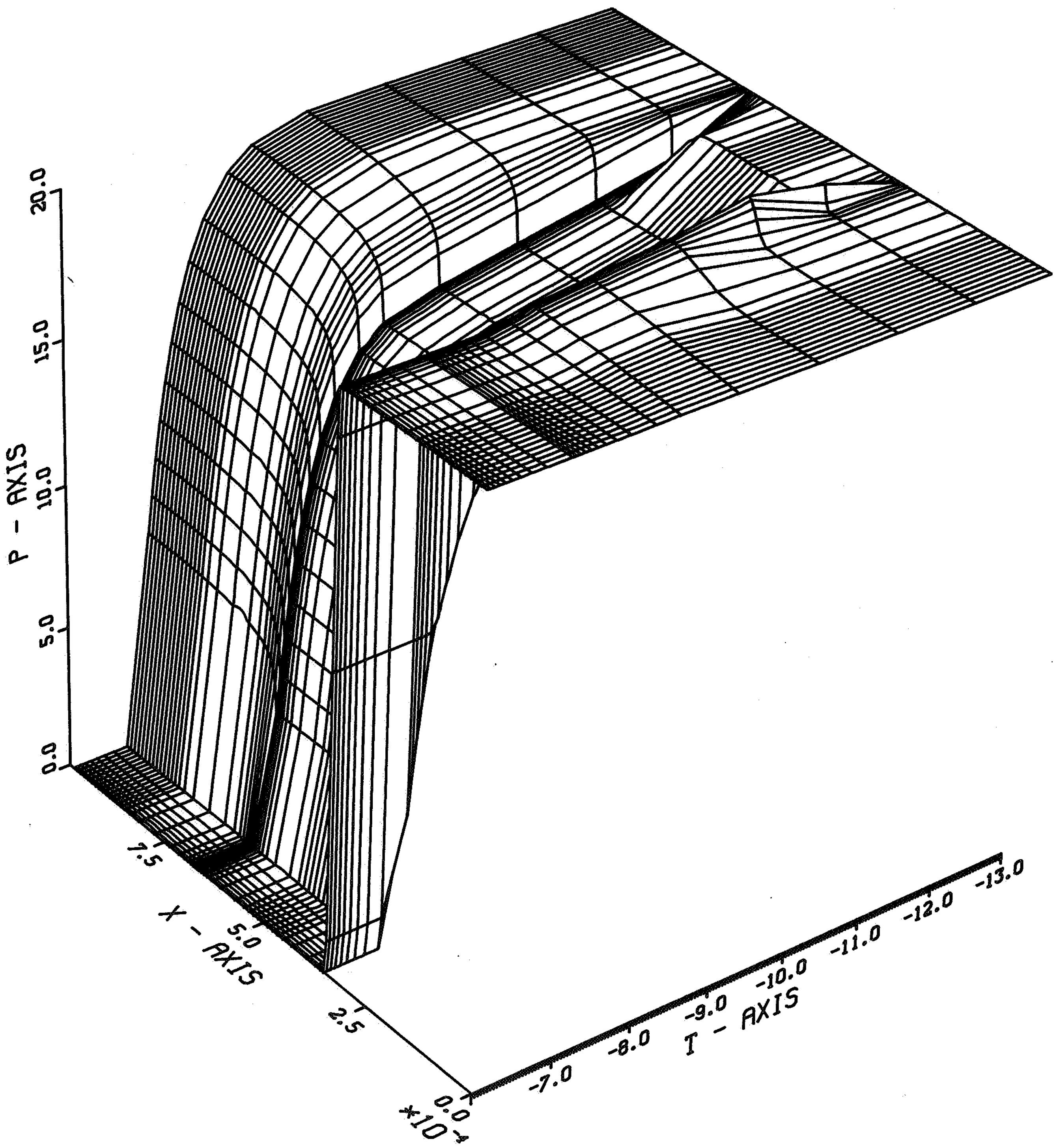


FIGURE 5.1b. The solution of problem II in (ψ,p)-formulation.

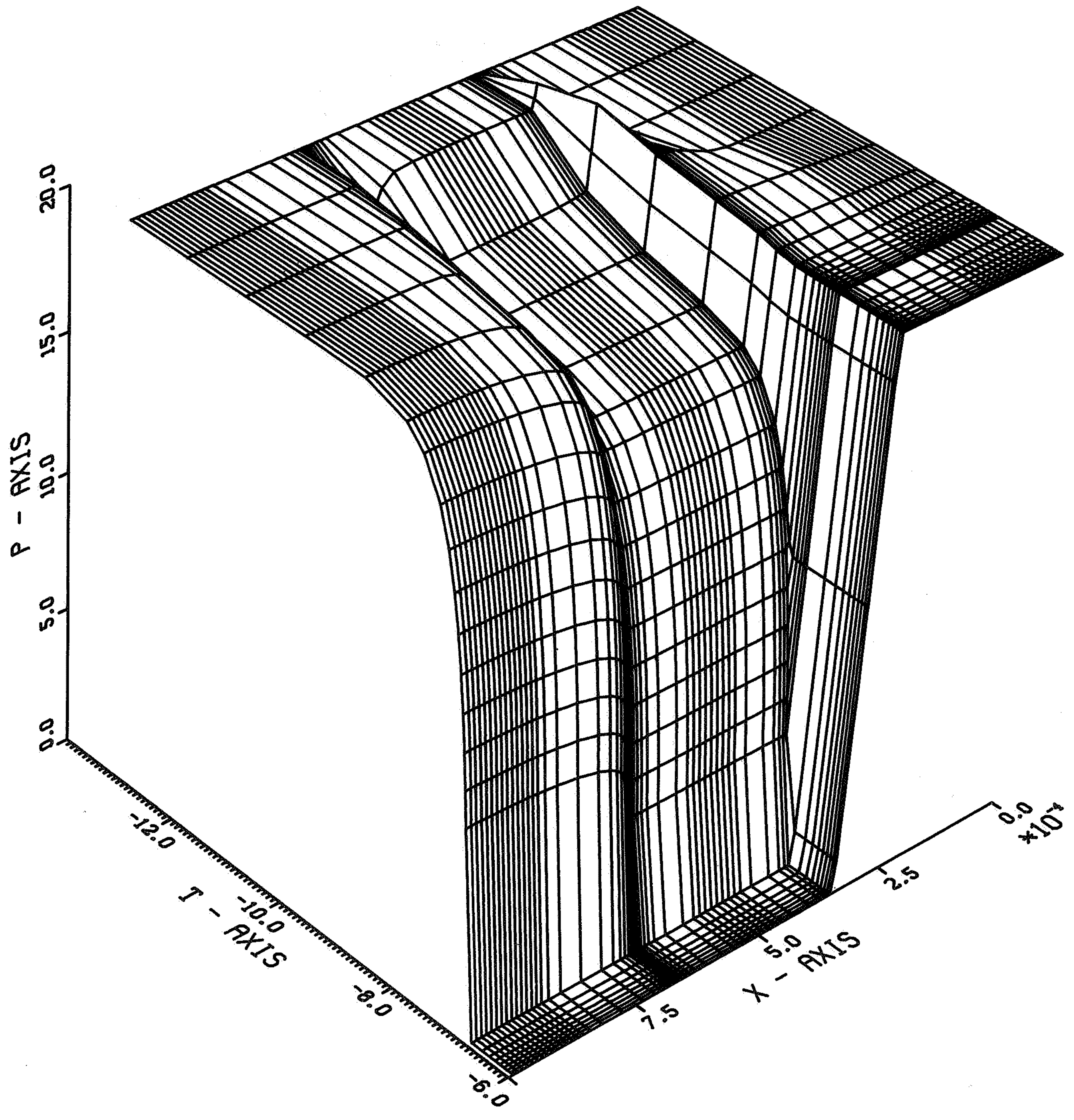


FIGURE 5.1c. The solution of problem II in (ψ,p)-formulation.

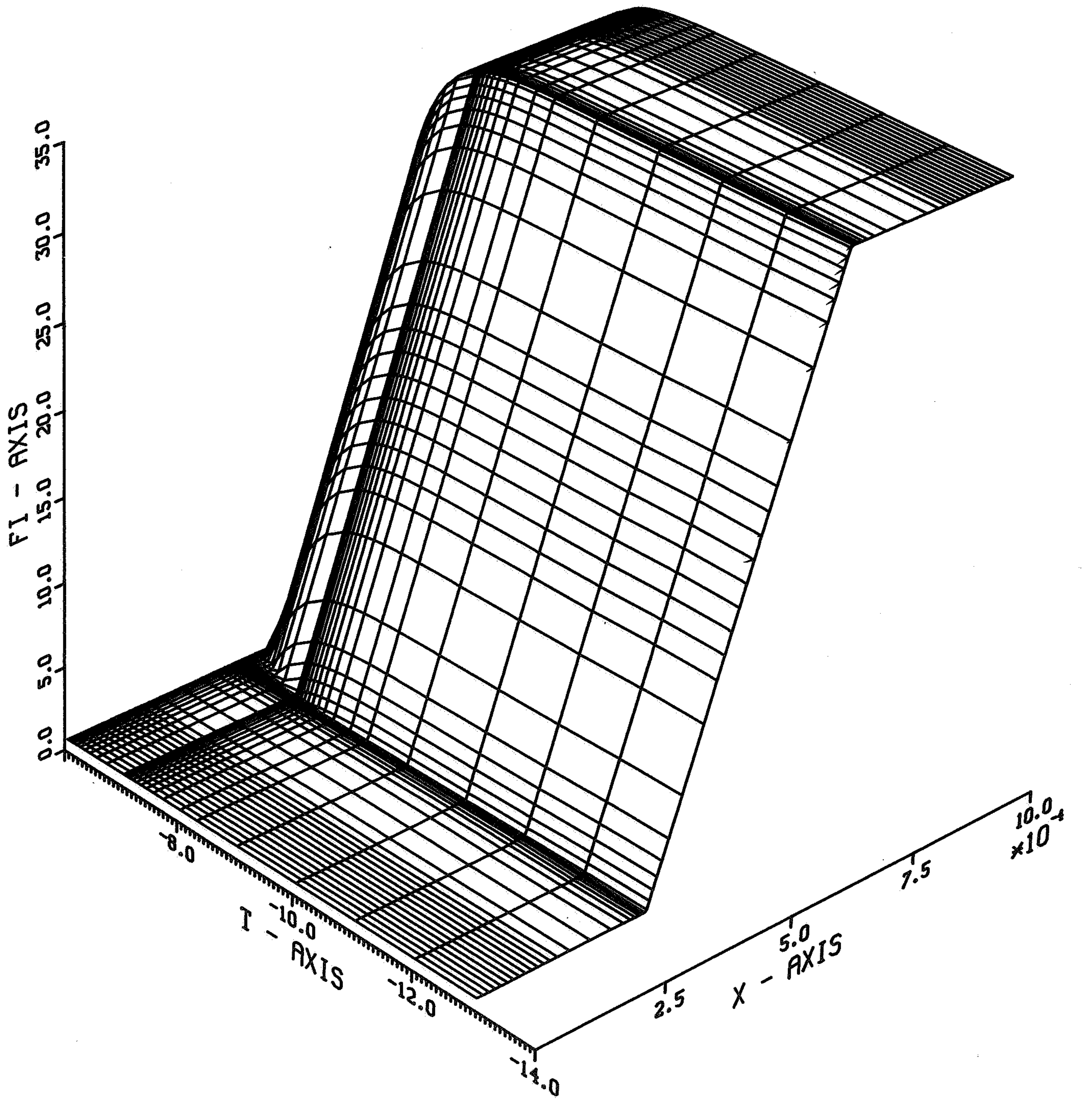


FIGURE 5.2a. The solution of problem II in (ψ,ϕ)-formulation.

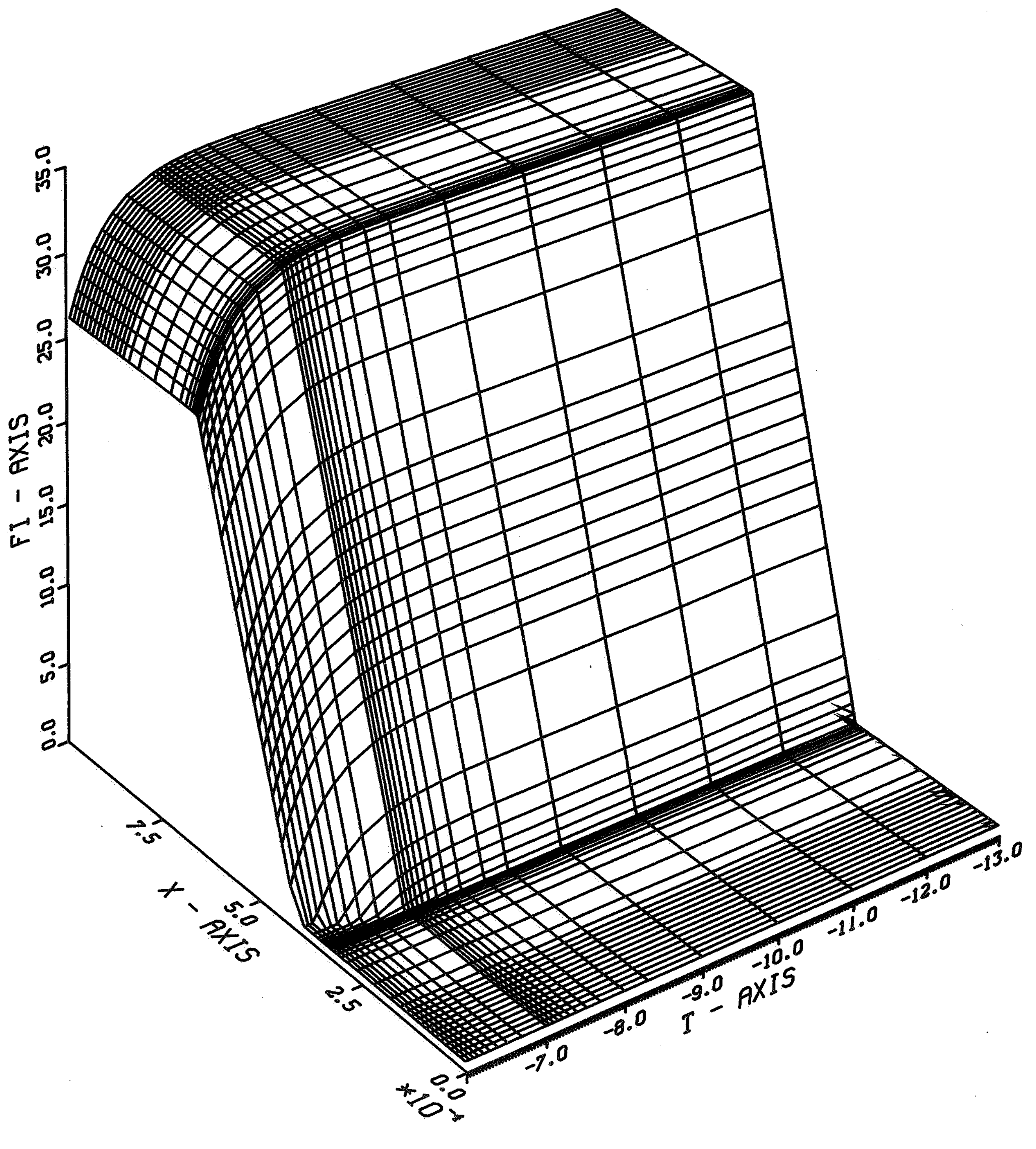


FIGURE 5.2b. The solution of problem II in (ψ,ϕ)-formulation.

TABLE 5.1 Values of $C(t)*10^{-17}$

	Problem		
t	II	III	IV
0	.9946	.9946	.9946
10^{-12}	.9946	.9946	.9946
10^{-11}	.9946	.9946	.9946
10^{-10}	.9946	.9946	.9946
10^{-9}	.9648	.9701	.9685
10^{-8}	.4279	.4487	.4427
10^{-7}	.3305	.3387	.3363

We wish to conclude this section with some remarks on the feasibility of the discretization in space. When carrying out the computations reported here we noticed the following points:

1°. The exponential fitting scheme gives an excellent shock positioning (also on coarser grids) without undershoot and overshoot and hardly any smearing of the shock. The idea of exponential fitting is of clear value in this respect.

2°. However, to reach a relative error of at most about 1% in the domain of interest, quite a number of grid points must be used and, no doubt, a non-uniform grid is a prerequisite. So our impression is that the performance of the exponential fitting scheme, despite the correct shock handling, is insufficient for the present problems in that it requires quite a number of grid points for a moderate accuracy.

3°. It is advisable to refine near the boundaries $x=0,x_e$. This improves the accuracy.

4°. We have discretized the Neumann condition $P_x(x_e,t)=0$ using the simple, first-order replacement $P_{m+1}(t)=P_m(t)$. Obviously, a second-order discretization is possible. This will probably enlarge the accuracy somewhat.

5.3. The integration

As noted in Section 4 one of our tasks is to carry out a comparison between the backward Euler scheme (4.1) and the second-order scheme (4.2). On the basis of our experiences gained during the experiments with *GEARB*, we now consider it necessary to compare the schemes (4.1), (4.2) when provided with *stepsize and local error control.* The semi-discretizations (2.9), (2.15) are very stiff and a ratio of approximately 10^6 between the smallest and largest steplength, used in *GEARB*, was observed in all our experiments with this code. Consequently, variable steplength is a prerequisite here and so we decided to implement the schemes (4.1), (4.2) that way.

Another experience gained in Section 5.2 is that *GEARB* performed very well on our semi-discrete PDE, and so the obvious question arose whether the higher-order BDF formulas — when provided with steplength and error control as in *GEARB* — wouldn't be very appropriate. Consequently, we have decided to involve *GEARB* also in the numerical comparison. We applied two versions, namely with its order maximal 1, in which case the only scheme used is backward Euler, and with its order maximal 5 as we used it in Section 5.2.

To sum up, in the remainder of the investigation we compare *4 integrators*. These are

GEARB :	The code by Hindmarsh [3] using the BDF formulas with orders 1 to 5.
GEARB 1 :	The code by Hindmarsh [3] using only the 1-st order backward Euler scheme (4.1).

BE :	Our code based on the backward Euler scheme (4.1). So *BE* differs from *GEARB* 1 in that we use a different local error estimator, a different stepsize change procedure and a different Jacobian update procedure.
BELL:	Our code based on the 2-nd order scheme (4.2). Also in this case the local error estimator and the various control procedures in our code are different from the ones proposed in [1]. This is based partly on efficiency considerations and partly on personal taste.

For a detailed description of *BE* and *BELL* we refer to Appendix 2. Here it is enough to say that the estimation and control procedures built in in *BE* and *BELL* are of the sort one uses in stiff *ODE* codes. After each trial step (double step in *BE*) estimates EST_j of the local truncation error for component $j=1,2, \cdots ,m$ are computed. These values are then "normed" to obtain the estimate EST which is to be compared with the user supplied tolerance parameter TOL. If $EST \leqslant TOL$, the step has been successful and the integration is continued with a possibly new value for the stepsize τ and a possible update of the Jacobian matrix in the Newton process. Otherwise, if $EST > TOL$, the step is redone as it is customary (possibly a Jacobian update, etc.). We remark that we use the analytical Jacobians and the initial trial step is prescribed (in all experiments discussed below, this first trial step $\tau = 10^{-15}$).

We conclude this section with an *important remark* on how the component twise estimates EST_j are "normed". This differs for the two *ODE* systems (2.9), (2.15) according to the remark made after the lemma in Section 1. In *all* 4 codes we "norm" EST_j to the *relative* error expression

$$EST = (\frac{1}{m} \sum_{j=1}^{m} (EST_j / P_j)^2)^{1/2} \tag{5.17}$$

if we integrate (2.9), and we use the *absolute* L^2-norm

$$EST = (\frac{1}{m} \sum_{j=1}^{m} EST_j^2)^{1/2} \tag{5.18}$$

if we integrate (2.15). We observe that the factor α^{-1} featuring in the lemma is recovered in the numerical experiments (see (5.19) and Tables 5.2 - 5.5).

Nomenclature

In the tables of result we use the following nomenclature:

TOL:	user supplied tolerance parameter
t:	output point
cd_Φ:	$-\log_{10}(max_j \lvert \Phi_j(t) - \Phi_j^n \rvert)$
cd_p:	$-\log_{10}(\max_j \lvert (P_j(t) - P_j^n)/P_j(t) \rvert)$
$cd_{p\Phi}$:	$-\log_{10}(\max_j \lvert [P_j(t) - n_i \exp(\alpha(\Phi_j^n - \psi_j))]/P_j(t) \rvert)$
STEPS:	number of accepted steps; in brackets the number of rejected steps. It is noted that one step with the *BELL* scheme (4.2) is counted as two steps in *STEPS*.
F_{eval}:	number of calls of the *ODE* operator.

J_{eval}: number of Jacobian matrix (in analytic form) evaluations; any evaluation is of course accompanied by an *LU*-decomposition.

$C(t)$: approximations for the values $C(t)$ given by (3.3), the exact values of which were given in Table 5.1.

CPU: central processor time in sec. on the *CYBER* 750.

We recall that $P_j(t), \Phi_j(t)$ are the reference solutions computed by $GEARB(TOL = 10^{-10})$ on the non-uniform grid as discussed in Section 2. We emphasize that the number of significant digits $cd_{p\Phi}$ measures the accuracy obtained after a back-transformation of the computed approximation Φ_j^n. With reference to the lemma in Section 1, we expect the following approximate relation between $cd_{p\Phi}$ and cd_Φ

$$cd_{p\Phi} \simeq cd_\Phi - \log_{10}\alpha \simeq cd_\Phi - 1.6\,. \tag{5.19}$$

The tables of result in Section 5.4 indeed reveal this.

5.4. *Results in* (ψ,ϕ)

TABLE 5.2. Results for *BELL* applied to Problem II in (ψ,ϕ)-formulation

TOL	*t*	cd_Φ	$cd_{p\Phi}$	*STEPS*	F_{eval}	J_{eval}	$C(t)*10^{-17}$	*CPU*
	10^{-12}	2.50	0.88	20(0)	88	10	.9957	0.62
	10^{-11}	2.48	0.92	42(0)	204	27	1.000	1.65
10^{-2}	10^{-10}	2.90	1.30	56(0)	261	33	1.009	2.20
	10^{-9}	3.04	1.44	62(0)	286	36	.9885	2.54
	10^{-8}	3.62	2.03	68(0)	307	37	.4306	2.86
	10^{-7}	4.33	2.74	100(1)	429	44	.3305	3.82
	10^{-12}	2.98	1.39	20(0)	87	15	.9951	0.61
	10^{-11}	3.14	1.56	46(0)	212	38	.9959	1.69
10^{-3}	10^{-10}	3.36	1.77	64(0)	282	45	1.000	2.34
	10^{-9}	3.45	1.86	74(0)	321	49	.9732	2.78
	10^{-8}	3.95	2.36	86(0)	370	54	.4292	3.29
	10^{-7}	4.16	2.57	108(0)	465	70	.3305	4.12
	10^{-12}	3.55	1.96	34(0)	142	27	.9948	1.03
	10^{-11}	3.70	2.11	84(0)	349	59	.9950	2.67
10^{-4}	10^{-10}	3.94	2.36	116(0)	468	66	.9961	3.62
	10^{-9}	4.03	2.44	138(0)	549	70	.9671	4.31
	10^{-8}	4.36	2.78	156(0)	619	76	.4284	4.96
	10^{-7}	5.38	3.79	184(0)	746	100	.3305	6.03

TABLE 5.3. Results for *BE* applied to Problem II in (ψ,ϕ)-formulation

TOL	*t*	cd_ϕ	$cd_{p\phi}$	*STEPS*	F_{eval}	J_{eval}	$C(t)*10^{-17}$	*CPU*
	10^{-12}	2.05	0.38	12(0)	30	4	.9786	0.24
	10^{-11}	1.83	0.12	24(0)	62	5	.9032	0.66
10^{-2}	10^{-10}	2.11	0.59	34(0)	90	6	.8874	1.03
	10^{-9}	2.32	0.77	40(0)	105	6	.8552	1.31
	10^{-8}	3.53	1.95	46(0)	121	7	.4298	1.61
	10^{-7}	2.90	1.33	60(0)	174	15	.3305	2.20
	10^{-12}	2.43	0.81	24(0)	64	2	.9883	0.47
	10^{-11}	2.22	0.58	60(0)	156	3	.9573	1.31
10^{-3}	10^{-10}	2.49	0.93	84(0)	224	3	.9477	1.95
	10^{-9}	2.69	1.12	102(0)	265	4	.9158	2.42
	10^{-8}	3.28	1.69	114(0)	294	5	.4339	2.80
	10^{-7}	4.17	2.59	138(2)	385	13	.3305	3.63
	10^{-12}	2.89	1.29	64(0)	139	1	.9925	1.02
	10^{-11}	2.68	1.08	188(0)	416	2	.9820	3.19
10^{-4}	10^{-10}	2.94	1.36	254(0)	562	2	.9777	4.42
	10^{-9}	3.12	1.53	304(0)	662	2	.9460	5.33
	10^{-8}	3.79	2.20	332(0)	728	3	.4298	5.98
	10^{-7}	4.70	3.11	400(3)	909	6	.3305	7.40

TABLE 5.4. Results for *GEARB* 1 applied to Problem II in (ψ,ϕ)-formulation

TOL	*t*	cd_ϕ	$cd_{p\phi}$	*STEPS*	F_{eval}	J_{eval}	$C(t)*10^{-17}$	*CPU*
	10^{-12}	1.78	0.32	8(0)	13	3	.9756	0.20
	10^{-11}	1.70	-0.06	16(0)	27	6	.8713	0.48
	10^{-10}	1.94	0.44	23(0)	36	8	.8430	0.71
10^{-2}	10^{-9}	2.03	0.52	28(0)	41	10	.7664	0.93
	10^{-8}	2.87	1.29	33(0)	46	11	.4127	1.14
	10^{-7}	overflow *)						
	10^{-12}	2.16	0.52	14(0)	26	5	.9826	0.27
	10^{-11}	1.93	0.25	30(0)	58	9	.9216	0.65
10^{-3}	10^{-10}	2.22	0.68	44(0)	73	13	.9073	0.94
	10^{-9}	2.40	0.85	55(0)	84	17	.8713	1.20
	10^{-8}	3.91	2.32	63(0)	92	19	.4296	1.43
	10^{-7}	1.97	0.47	78(0)	126	25	.3305	1.81
	10^{-12}	2.55	0.94	31(0)	60	7	.9899	0.45
	10^{-11}	2.34	0.71	85(0)	123	11	.9672	1.01
10^{-4}	10^{-10}	2.62	1.05	119(0)	157	21	.9593	1.44
	10^{-9}	2.79	1.22	143(0)	181	29	.9256	1.81
	10^{-8}	3.78	2.19	160(1)	202	34	.4298	2.13
	10^{-7}	4.45	2.86	201(7)	269	40	.3305	2.68

*) Due to divergence in the Newton process. It can easily be prevented through a safety check on each Newton iterant. In fact, we have built in such a check in *BELL* and *BE*.

TABLE 5.5. Results for *GEARB* applied to Problem II in (ψ,ϕ)-formulation

TOL	t	cd_{ϕ}	$cd_{p\phi}$	*STEPS*	F_{eval}	J_{eval}	$C(t)*10^{-17}$	*CPU*
	10^{-12}							
	10^{-11}							
10^{-2}	10^{-10}							
	10^{-9}			Exactly the same results as for *GEARB* 1				
	10^{-8}							
	10^{-7}							
	10^{-12}	2.16	0.52	14(0)	26	5	.9826	0.27
	10^{-11}	2.35	0.80	30(0)	53	7	.9827	0.62
10^{-3}	10^{-10}	2.55	0.99	45(0)	70	12	.9683	0.93
	10^{-9}	2.69	1.12	55(0)	80	15	.9241	1.17
	10^{-8}	2.94	1.34	63(0)	89	18	.4411	1.41
	10^{-7}	3.98	2.39	82(1)	123	24	.3305	1.82
	10^{-12}	3.49	1.90	24(0)	42	6	.9947	0.39
	10^{-11}	3.14	1.56	56(0)	89	9	.9947	0.88
10^{-4}	10^{-10}	3.89	2.31	87(0)	120	15	.9933	1.30
	10^{-9}	3.87	2.29	104(0)	137	18	.9618	1.61
	10^{-8}	3.41	1.82	120(1)	159	21	.4325	1.93
	10^{-7}	4.29	2.70	144(4)	209	24	.3305	2.41

Inspection of Tables 5.2 - 5.5 gives rise to the following observations and conclusions:

1. As we expected, if one works with the (ψ,ϕ)-formulation one looses accuracy ($\simeq$1.6 digits) when the approximations are transformed back. This implies that the problem in (ψ,ϕ)-formulation requires a more accurate numerical solution process than the original (ψ,p)-formulation (see also the next section).
2. When comparing the 4 codes (for problem II in (ψ,ϕ)-formulation) there is no doubt that *GEARB* is the most efficient one, although *BELL* also performs satisfactorily. The first-order codes *BE* and *GEARB* 1 fall behind, even in the modest accuracy range $1 \leqslant cd_{p\phi} \leqslant 2$. Apparently, it is advantageous here to use higher-order formulas (in *GEARB* the order ranges from 1 to 5).
3. *BELL* integrates quite accurately when taking into account that its underlying *RK* integration formula (4.2) is only of 2-nd order (and *L*-stable [1]). Yet *GEARB* is more efficient due to its higher-order formulas. It is likely that our implementation of *BELL* can be improved a little (see also Section 5.5). However, we don't think it can be made faster than *GEARB*.
4. Observe that, as expected, the accuracy of the $C(t)$-values (compare with the exact values in Table 5.1) is of the same level as those of P.

We have repeated the experiments for the problems III, IV and came to the same conclusions as for problem II. To save space, the tables of results for problem III and IV are therefore omitted.

5.5. Intermezzo on BELL

Before continuing with our results in the (ψ,p)-formulation, where we deal with a constant coefficient linear *ODE* problem, we wish to draw attention to a phenomenon which has to do with the numerical solution of the implicit algebraic equations and with the *stability of the code* for nonlinear problems. We confine ourselves to *BELL* since the phenomenon did not turn up with the other codes.

The phenomenon is best illustrated from Table 5.6 which shows results of *BELL*, applied to problem II in (ψ,ϕ)-formulation for $TOL=10^{-3}$, using 4 criteria for acceptance of the Newton iterants. In the code the acceptance criterion is

$$\max_j \left| \frac{\text{Newton correction for component j}}{\text{current value of component j}} \right| < TOL/f \qquad (5.20)$$

where f is a *parameter* which in Table 5.6 takes on the values $f=10,100,400,1000$. Thus the larger f, the more stringent is the acceptance criterion.

TABLE 5.6. Performance of *BELL* for different values of f, applied to Problem II in (ψ,ϕ)-formulation $TOL=10^{-3}$.

TOL/f	t	cd_ϕ	$cd_{p\phi}$	$STEPS$	F_{eval}	J_{eval}	$C(t)*10^{-17}$	CPU
	10^{-12}	2.99	1.39	20(0)	63	2	.9951	.42
	10^{-11}	3.14	1.55	46(0)	155	4	.9962	1.19
$TOL/10$	10^{-10}	3.37	1.78	64(0)	211	5	1.000	1.73
	10^{-9}	3.45	1.86	76(0)	249	5	.9740	2.13
	10^{-8}	4.13	2.55	108(1)	309	6	.4287	2.65
	10^{-7}	4.80	3.22	1474(12)	3461	8	.3305	21.37
	10^{-12}	2.98	1.39	20(0)	79	13	.9951	.58
	10^{-11}	3.14	1.56	46(0)	190	29	.9959	1.53
$TOL/100$	10^{-10}	3.37	1.78	64(0)	256	32	.9999	2.12
	10^{-9}	3.45	1.86	74(0)	293	34	.9732	2.53
	10^{-8}	4.06	2.48	86(0)	332	36	.4289	2.96
	10^{-7}	4.82	3.23	442(10)	1192	39	.3305	8.40
	10^{-12}	2.98	1.39	20(0)	87	15	.9951	.64
	10^{-11}	3.14	1.56	46(0)	210	38	.9959	1.72
$TOL/400$	10^{-10}	3.36	1.77	64(0)	280	45	1.000	2.36
	10^{-9}	3.45	1.86	74(0)	319	49	.9732	2.79
	10^{-8}	4.02	2.44	86(0)	363	51	.4290	3.22
	10^{-7}	3.44	1.85	114(1)	490	66	.3305	4.23
	10^{-12}	2.98	1.39	20(0)	93	15	.9951	.67
$TOL/1000$	10^{-11}	3.14	1.56	46(0)	227	41	.9959	1.82
	10^{-10}	3.36	1.77	64(0)	302	51	1.000	2.51
	10^{-9}	3.45	1.86	74(0)	342	56	.9732	2.93
	10^{-8}	3.92	2.34	84(0)	383	60	.4292	3.36
	10^{-7}	4.44	2.85	108(1)	508	81	.3305	4.37

We see in Table 5.6 that up to and including $t=10^{-8}$ the variation in f does have hardly any influence. However, for the interval $10^{-8} \leqslant t \leqslant 10^{-7}$, the value of f plays a significant role. For example, for $f=10$ *BELL* needs $1474-108=1366$ steps on this interval, but only $108-84=24$ for $f=1000$. If we count the number of evaluations of the Jacobian, we observe the reverse. For $f=1000, J_{eval}$ is much larger than for $f=10$. Here lies the explanation for the difference in performance when f varies. For f large, at each step the implicitly defined approximations $y_{n+\gamma}, y_{n+1}$ are more or less exactly determined which is obviously not true if f is small. Apparently, the *BELL* formula itself (the case $f=1000$, say) has no difficulties with the problem. It yields an accurate result using only 54 full steps. The effect of using a too small value for f, e.q. $f=10$ or $f=100$, is that the code carries out only a very few Jacobian updates and Newton iterations per *RK* stage (on the average approximately 2 for $f = 10, 100$) which in turn forces the code to integrate with smaller values for τ than necessary. The issue at hand is that in these circumstances the *stability in time* of the accepted Newton iterants for the *RK* scheme (4.2) bounds the steplength through the stepsize and local error control mechanism. So, although the scheme (4.2) itself is all right here as is illustrated for $f = 1000$, when implemented and applied to a *nonlinear problem* its stability may deteriorate. We believe that the source of the difficulties lies with the trapezoidal rule in (4.2) and that the above phenomenon is related to the notion of internal stability in RUNGE-KUTTA-ROSENBROCK methods ([2]. Ch.9; See in particular Section 9.4. The interested reader should also read the experiment with the *RK* code *STRIDE* pp. 182-183 in [2] where a behaviour similar to the one here is discussed).

Finally we remark that on the basis of the experiences reported in this section, we decided to put f equal to 500 in the experiments of Table 5.2.

5.6. Results in (ψ, p)

TABLE 5.7. Results for *BELL* applied to Problem II in (ψ, p)-formulation

TOL	*t*	cd_p	*STEPS*	F_{eval}	J_{eval}	$C(t)*10^{-17}$	*CPU*
	10^{-12}	1.59	34(0)	52	1	.9946	0.23
	10^{-11}	1.36	90(0)	136	1	.9946	0.76
	10^{-10}	2.78	118(0)	178	1	.9946	1.11
10^{-2}	10^{-9}	2.82	136(0)	205	1	.9649	1.41
	10^{-8}	2.04	158(0)	238	1	.4275	1.71
	10^{-7}	0.94	322(0)	484	1	.3305	2.72
	10^{-12}	2.19	62(0)	94	1	.9946	0.40
	10^{-11}	2.00	176(0)	265	1	.9946	1.26
	10^{-10}	3.39	230(0)	346	1	.9946	1.77
10^{-3}	10^{-9}	3.43	268(0)	403	1	.9648	2.19
	10^{-8}	2.69	314(0)	472	1	.4278	2.65
	10^{-7}	1.57	658(0)	988	1	.3305	5.07
	10^{-12}	2.82	124(0)	187	1	.9946	0.83
	10^{-11}	2.65	366(0)	550	1	.9946	2.47
10^{-4}	10^{-10}	4.02	476(0)	715	1	.9946	3.37
	10^{-9}	4.05	554(0)	832	1	.9648	4.06
	10^{-8}	3.35	648(0)	973	1	.4278	4.81
	10^{-7}	2.23	1384(0)	2077	1	.3305	9.76

TABLE 5.8. Results for *BE* applied to Problem II in (ψ,p)-formulation

TOL	t	cd_p	$STEPS$	F_{eval}	J_{eval}	$C(t)*10^{-17}$	CPU
	10^{-12}	0.71	58(0)	58	1	.9946	0.37
	10^{-11}	0.54	174(0)	174	1	.9946	1.23
10^{-2}	10^{-10}	1.89	226(0)	226	1	.9946	1.74
	10^{-9}	1.94	262(0)	262	1	.9638	2.16
	10^{-8}	1.16	318(0)	318	1	.4318	2.69
	10^{-7}	-0.06	794(0)	794	1	.3305	5.88
	10^{-12}	1.21	172(0)	172	1	.9946	1.11
	10^{-11}	1.10	540(0)	540	1	.9946	3.39
10^{-3}	10^{-10}	2.36	686(0)	686	1	.9946	4.49
	10^{-9}	2.41	796(0)	796	1	.9648	5.36
	10^{-8}	1.66	964(0)	964	1	.4278	6.57
	10^{-7}	0.55	2470(0)	2470	1	.3305	16.37
	10^{-12}	1.71	536(0)	536	1	.9946	3.40
	10^{-11}	1.62	1700(0)	1700	1	.9946	10.19
10^{-4}	10^{-10}	2.84	2146(0)	2146	1	.9946	13.19
	10^{-9}	2.90	2486(0)	2486	1	.9646	15.54
	10^{-8}	2.16	3014(0)	3014	1	.4283	18.93
	10^{-7}	1.08	7774(0)	7774	1	.3305	49.49

TABLE 5.9. Results for *GEARB* 1 applied to Problem II in (ψ,p)-formulation

$TOL^{*)}$	t	cd_p	$STEPS$	F_{eval}	J_{eval}	$C(t)*10^{-17}$	CPU
	10^{-12}	0.86	79(0)	114	5	.9946	0.43
	10^{-11}	0.70	242(0)	280	14	.9946	1.27
	10^{-10}	2.01	312(0)	350	27	.9946	1.78
10^{-3}	10^{-9}	2.06	362(0)	400	35	.9640	2.21
	10^{-8}	1.29	435(0)	473	40	.4308	2.69
	10^{-7}	0.10	1069(0)	1107	72	.3305	5.43
	10^{-12}	1.36	239(0)	274	13	.9946	1.08
	10^{-11}	1.25	749(0)	787	39	.9946	3.30
10^{-4}	10^{-10}	2.51	961(0)	999	51	.9946	4.32
	10^{-9}	2.57	1120(0)	1158	59	.9645	5.14
	10^{-8}	1.80	1353(0)	1391	71	.4288	6.23
	10^{-7}	0.70	3402(0)	3440	173	.3305	14.57
	10^{-12}	1.86	759(1)	795	41	.9946	3.16
	10^{-11}	1.78	2420(1)	2458	124	.9946	9.98
10^{-5}	10^{-10}	2.99	3049(1)	3087	156	.9946	12.63
	10^{-9}	3.05	3528(1)	3566	180	.9647	14.74
	10^{-8}	2.32	4292(1)	4330	218	.4281	17.96
	10^{-7}	1.25	11204(1)	11242	563	.3305	45.69

*) In the run with $TOL=10^{-2}$, *GEARB* 1 failed. Therefore, we choose $TOL=10^{-3},10^{-4},10^{-5}$.

TABLE 5.10. Results for *GEARB* applied to Problem II in (ψ,p)-formulation

TOL	t	cd_p	*STEPS*	F_{eval}	J_{eval}	$C(t)*10^{-17}$	*CPU*
	10^{-12}	0.85	24(0)	45	5	.9946	0.20
	10^{-11}	1.15	62(0)	98	9	.9946	0.64
	10^{-10}	2.19	87(0)	123	14	.9946	0.96
10^{-2}	10^{-9}	2.22	103(0)	139	18	.9652	1.23
	10^{-8}	2.79	119(0)	157	21	.4274	1.50
	10^{-7}	0.83	190(3)	248	26	.3305	2.17
	10^{-12}	1.90	35(0)	64	6	.9946	0.30
	10^{-11}	2.42	88(0)	122	11	.9946	0.80
10^{-3}	10^{-10}	3.21	129(0)	164	17	.9946	1.24
	10^{-9}	3.27	154(0)	189	22	.9647	1.56
	10^{-8}	3.10	178(0)	214	26	.4279	1.87
	10^{-7}	1.44	274(8)	336	34	.3305	2.76
	10^{-12}	3.05	50(0)	80	6	.9946	0.39
	10^{-11}	2.91	117(0)	159	11	.9946	1.04
10^{-4}	10^{-10}	4.20	170(0)	216	19	.9946	1.58
	10^{-9}	4.15	202(0)	248	24	.9648	1.96
	10^{-8}	4.68	233(0)	280	28	.4279	2.34
	10^{-7}	2.21	342(0)	393	34	.3305	3.24

Before discussing the Tables 5.7 - 5.10, a few comments are in order. Since the problem in P-variable is linear (see (2.9)), we now have to solve, in all 4 codes, linear systems of algebraic equations of the same generic form, instead of the nonlinear systems which arise in the integration of (2.15). For example, one step of the backward Euler scheme (4.1) applied to (2.9) gives rise to the linear system

$$(I-\tau A)P^{n+1} = P^n + \tau B^{n+1}. \tag{5.21}$$

Consequently, during the integration process no reevaluations of the Jacobian matrix are necessary and, once $I-\tau A$ has been LU-decomposed, one integration step only costs a forward-backward solve. The LU-decomposition has to be done at the start of the process and any time τ changes. Our housemade codes *BE* and *BELL* were applied this way, but not *GEARB*1 and *GEARB* (there is no facility for distinguishing linear and nonlinear problems; although it is easy to built in we decided to leave this undone, just for convenience). Consequently, in the comparison, this merely technical point has to be reckoned with.

Inspection of Tables 5.7 - 5.10 leads us to the following observations and conclusions:

1. As in the (ψ,ϕ)-formulation *GEARB* is again the most efficient code for the present problem. *BE* and *GEARB*1, the first-order codes, clearly fall behind. Again we see that the 2-nd order *BELL* code performs satisfactorily. However, it cannot compete with *GEARB* which is faster for a required level of accuracy. So again we may conclude that the use of higher-order formulas is beneficial here.
2. In all runs we observe a drop in accuracy at the final subinterval. This is caused by the internal layer in time (see Fig. 5.1). Recall that in the (ψ,ϕ)-formulation this drop in accuracy does not occur (see also the next section).
3. Comparison of the total charge values $C(t)$ with those of Table 5.1 reveals that if we work in P-

variable, $C(t)$ is computed very accurately and, in general, significantly more accurately than P itself. This is due to the fact that the accuracy by which $P_j, j>2$, is computed, has no influence. Solely the error in P_1^n as an approximation for $P_1(t_n)$ and the discretization error of the integration formulas, when applied to (3.8), determine the error in the total charge values (see Appendix 3 for an explanation). Apparently, both these errors are small.

4. When we compare the performances of *BE* and *GEARB* 1, e.g. for respectively $TOL = 10^{-4}$ and $TOL = 10^{-5}$, we see that the *CPU* time of *BE* is larger than that of *GEARB* 1, whereas the number of steps taken by *BE* is significantly smaller. The explanation lies in the stepsize control implemented in *BE*. Within the layer, the changes in τ are extremely small so that one can say that *BE* resolves the layer using almost constant stepsizes. However, any change of τ in *BE* involves an *LU*-decomposition of the matrix in (5.21). The code has not been protected to omit too small changes in τ so as to reduce the number of decompositions. This explain the discrepancy in the *CPU* times of *GEARB* 1 and *BE*. Finally, the *CPU* times for *BELL* are also influenced negatively, because *BELL* uses the same strategy as *BE*. However, our conclusion 1 remains unchanged.

Like in the previous section, we have repeated all experiments for problems III and IV. It turned out that the conclusions above remain unchanged. In order to save space we omit the tables of result.

6. Conclusions and suggestions

With reference to Section 4 and Section 5.3 - where we decided to implement stepsize and local error control and to test also *GEARB* 1 and *GEARB* - we finish our report with the following conclusions and suggestions.

Concerning the integrators:

1. For the present set of test problems the code *GEARB* is to be preferred to the housemade code *BELL* based on the 2-nd order *RK* formula (4.2), while the first-order codes *BE* and *GEARB* 1 cannot compete to both *GEARB* and *BELL*. It is fair to say, however, that the non-optimized code *BELL* performs satisfactorily on these difficult problems. *GEARB* is very clearly benefited by its higher-order formulas (order 1-5) and thus is faster, for a certain level of accuracy, than *BELL*. This conclusion is valid for both the (ψ,ϕ) and the (ψ,p)-formulation, not only in the high accuracy range, but also if one is satisfied with a moderate accuracy of 1% to 10%, say.
2. A side conclusion derived from conclusion 1 is that the application of higher-order time integration formulas is advantageous for the present *ODE* problem. We remark this since in the numerical solution of time-dependent *PDE*s the order in time of most of the popular schemes is equal to two.
3. Another side conclusion is that the method of lines approach, which enables us to use sophisticated stiff *ODE* codes, proves to be very useful here.

The (ψ,p) versus the (ψ,ϕ)-formulation:

To begin with we recall that the number of significant digits cd_p in Tables 5.7 - 5.10 must be compared to the number $cd_{p\Phi}$ in Tables 5.2 - 5.5. The numbers $cd_{p\Phi}$ were computed after the back-transformation $(\Phi \rightarrow P)$ which entails a loss of approximately 1.6 digits (cf. the end of Section 5.3). This partially explains why for a chosen value of *TOL* for all integrators $cd_{p\Phi}$ is mostly much smaller than cd_p.

1. The effect of the transformation differs per time interval. The reduction in cd_p due to the sharp layer at the end of the integration interval, is not seen in $cd_{p\Phi}$ (compare Tables 5.2, 5.7 (*BELL*) and 5.5, 5.10 (*GEARB*)). This is a clear advantage of using the Φ-variable. On the other hand, for

the initial part of the interval the P-variable is more attractive. When we consider the whole integration interval we see that working in Φ requires much less integration steps than in P.

2. With respect to the accuracy of the quantity $C(t)$, it is evident that working in P is to be preferred (see also point 3 at the end of Section 5.6). However, a word of warning is in order here. High accuracy in $C(t)$ may be misleading in the sense that it gives no guarantee for the accuracy of P. For example, in Table 5.8 we observe a nearly exact approximation to $C(10^{-7})$ for $TOL = 10^{-2}$, whereas the corresponding error in P is about 100%.
3. The advantage of a higher-order code, like *GEARB*, is that if TOL is decreased, the CPU time does not increase too much. Bearing this in mind, it is attractive to run the transformed problem (2.15) with a small value of TOL, since this yields the best overall accuracy (compare Table 5.5, $TOL = 10^{-4}$ to Table 5.10). Of course, the same argument applies to the untransformed problem (2.9). However, here the drop in accuracy near the time layer remains quite large (see Table 5.10).

Final conclusion and some suggestions for further research:

For the present set of test problems the code *GEARB* is very suitable and faster than the other codes considered in this report. The question whether the (ψ,p) or the (ψ,ϕ)-formulation is to be preferred is difficult to answer since here the accuracy in $C(t)$ has to be taken into account. Clearly, integrating in Φ-variable is attractive but yields larger errors in $C(t)$. The original (ψ,p)-formulation would certainly become more attractive if the code could be protected against the drop in accuracy in the layer, without abandoning too much of its speed. Another idea is to develop a technique for correcting for the loss of accuracy in $C(t)$ when integrating in Φ-variable (cf. point 3 of Section 4). These two points could be the subject of further research. Finally, of greatest interest is to find out to which extent our experiences on the model problem (1.1) carry over to realistic semi-conductor problems.

ACKNOWLEDGEMENT. We are grateful to our colleague P. de Zeeuw for providing us with the software to draw the plots in the Figures 5.1 and 5.2.

REFERENCES

[1] R.E. BANK, W.M. COUGHRAN, Jr., W. FICHTNER & D.J. ROSE, *Computational aspects of semiconductor device simulation,* Numerical Analysis manuscript 85-3, AT & T Bell Laboratories, 1985.

[2] K. DEKKER & J.G. VERWER, *Stability of Runge-Kutta methods for stiff, nonlinear differential equations,* North-Holland, 1984.

[3] A.C. HINDMARSH, *GEARB, Solution of ordinary differential equations having banded Jacobian,* UICD-30059, Rev. 2, Lawrence Livermore Laboratory, June 1977.

[4] A.M. IL'IN, *Differencing scheme for a differential equation with a small parameter affecting the highest derivative,* Math. Notes Acad. Sc. USSR 6, 596-602, 1969.

[5] S. SELBERHERR, *Analysis and simulation of semi-conductor devices,* Springer-Verlag, 1984.

Appendix 1. The non-uniform grid associated to the semi-discretizations (2.9), (2.15).

I	X(I)	H(I)
0	0.	.5842104656E-05
1	.5842104656E-05	.5842104656E-05
2	.1168420931E-04	.8542104656E-05
3	.1752631397E-04	.5842104656E-05
4	.2336841863E-04	.5842104656E-05
5	.2921052328E-04	.5842104656E-05
6	.3505262794E-04	.5842104656E-05
7	.4089473259E-04	.5842104656E-05
8	.4673683725E-04	.5842104656E-05
9	.5257894191E-04	.5842104656E-05
10	.5842104656E-04	.5842104656E-05
11	.6426315122E-04	.5842104656E-05
12	.7010525588E-04	.5842104656E-05
13	.7594736053E-04	.5842104656E-05
14	.8178946519E-04	.5842104656E-05
15	.8763156985E-04	.5842104656E-05
16	.9347367450E-04	.5842104656E-05
17	.9931577916E-04	.5842104656E-05
18	.1051578838E-03	.5842104656E-05
19	.1109999885E-03	.5842104656E-05
20	.1168420931E-03	.5842104656E-05
21	.1226841978E-03	.5842104656E-05
22	.1285263024E-03	.5842104656E-05
23	.1343684071E-03	.5842104656E-05
24	.1402105118E-03	.5842104656E-05
25	.1460526164E-03	.6130663997E-05
26	.1521832804E-03	.7969863196E-05
27	.1601531436E-03	.1036082215E-04
28	.1705139658E-03	.1346906880E-04
29	.1839830346E-03	.1750978944E-04
30	.2014928240E-03	.2276272627E-04
31	.2242555503E-03	.2276272627E-04
32	.2470182765E-03	.1750978944E-04
33	.2645280660E-03	.1346906880E-04
34	.2779971348E-03	.1036082215E-04
35	.2883579569E-03	.7969863196E-05
36	.2963278201E-03	.6130663997E-05
37	.3024584841E-03	.4715895382E-05
38	.3071743795E-03	.4287177620E-05
39	.3114615571E-03	.3897434200E-05
40	.3153589913E-03	.3543122000E-05
41	.3189021133E-03	.3221020000E-05
42	.3221231333E-03	.2928200000E-05
43	.3250513333E-03	.2662000000E-05
44	.3277133333E-03	.2420000000E-05
45	.3301333333E-03	.2200000000E-05
46	.3323333333E-03	.2000000000E-05
47	.3343333333E-03	.2200000000E-05
48	.3365333333E-03	.2420000000E-05

49	.3389533333E-03	.2662000000E-05
50	.3416153333E-03	.2928200000E-05
51	.3445435333E-03	.3221020000E-05
52	.3477645533E-03	.3543122000E-05
53	.3513076753E-03	.3897434200E-05
54	.3552051095E-03	.4287177620E-05
55	.3594922872E-03	.4715895382E-05
56	.3642081825E-03	.6602253535E-05
57	.3708104361E-03	.9243154949E-05
58	.3800535910E-03	.1294041693E-04
59	.3929940079E-03	.1811658370E-04
60	.4111105916E-03	.1811658370E-04
61	.4292271753E-03	.1294041693E-04
62	.4421675923E-03	.9243154949E-05
63	.4514107472E-03	.6602253535E-05
64	.4580130008E-03	.5998142749E-05
65	.4640111435E-03	.5998142749E-05
66	.4700092863E-03	.5998142749E-05
67	.4760074290E-03	.5998142749E-05
68	.4820055718E-03	.5998142749E-05
69	.4880037145E-03	.5998142749E-05
70	.4940018573E-03	.5998142749E-05
71	.5000000000E-03	.5998142749E-05
72	.5059981427E-03	.5998142749E-05
73	.5119962855E-03	.5998142749E-05
74	.5179944282E-03	.5998142749E-05
75	.5239925710E-03	.5998142749E-05
76	.5299907137E-03	.5998142749E-05
77	.5359888565E-03	.5998142749E-05
78	.5419869992E-03	.6602253535E-05
79	.5485892528E-03	.9243154949E-05
80	.5578324077E-03	.1294041693E-04
81	.5707728247E-03	.1811658370E-04
82	.5888894084E-03	.1811658370E-04
83	.6070059921E-03	.1294041693E-04
84	.6199464090E-03	.9243154949E-05
85	.6291895639E-03	.6602253535E-05
86	.6357918175E-03	.4715895382E-05
87	.6405077128E-03	.4287177620E-05
88	.6447948905E-03	.3897434200E-05
89	.6486923247E-03	.3543122000E-05
90	.6522354467E-03	.3221020000E-05
91	.6554564667E-03	.2928200000E-05
92	.6583846667E-03	.2662000000E-05
93	.6610466667E-03	.2420000000E-05
94	.6634666667E-03	.2200000000E-05
95	.6656666667E-03	.2000000000E-05
96	.6676666667E-03	.2200000000E-05
97	.6698666667E-03	.2420000000E-05
98	.6722866667E-03	.2662000000E-05
99	.6749486667E-03	.2928200000E-05

100	.6778768667E-03	.3221020000E-05
101	.6810978867E-03	.3543122000E-05
102	.6846410087E-03	.3897434200E-05
103	.6885384429E-03	.4287177620E-05
104	.6928256205E-03	.4715895382E-05
105	.6975415159E-03	.6130663997E-05
106	.7036721799E-03	.7969863196E-05
107	.7116420431E-03	.1036082215E-04
108	.7220028652E-03	.1346906880E-04
109	.7354719340E-03	.1750978944E-04
110	.7529817235E-03	.2276272627E-04
111	.7757444497E-03	.2276272627E-04
112	.7985071760E-03	.1750978944E-04
113	.8160169654E-03	.1346906880E-04
114	.8294860342E-03	.1036082215E-04
115	.8398468564E-03	.7969863196E-05
116	.8478167196E-03	.6130663997E-05
117	.8539473836E-03	.4868420547E-05
118	.8588158041E-03	.4868420547E-05
119	.8636842247E-03	.4868420547E-05
120	.8685526452E-03	.4868420547E-05
121	.8734210658E-03	.4868420547E-05
122	.8782894863E-03	.4868420547E-05
123	.8831579069E-03	.4868420547E-05
124	.8880263274E-03	.4868420547E-05
125	.8928947480E-03	.4868420547E-05
126	.8977631685E-03	.4868420547E-05
127	.9026315891E-03	.4868420547E-05
128	.9075000096E-03	.4868420547E-05
129	.9123684302E-03	.4868420547E-05
130	.9172368507E-03	.4868420547E-05
131	.9221052712E-03	.4868420547E-05
132	.9269736918E-03	.4868420547E-05
133	.9318421123E-03	.4868420547E-05
134	.9367105329E-03	.4868420547E-05
135	.9415789534E-03	.4868420547E-05
136	.9464473740E-03	.4868420547E-05
137	.9513157945E-03	.4868420547E-05
138	.9561842151E-03	.4868420547E-05
139	.9610526356E-03	.4868420547E-05
140	.9659210562E-03	.4868420547E-05
141	.9707894767E-03	.4868420547E-05
142	.9756578973E-03	.4868420547E-05
143	.9805263178E-03	.4868420547E-05
144	.9853947384E-03	.4868420547E-05
145	.9902631589E-03	.4868420547E-05
146	.9951315795E-03	.4868420547E-05
147	.1000000000E-02	

Appendix 2. Implementation of the methods

A.2.1. The Backward Euler method

Consider the differential equation

$$\dot{y}(t) = f(t,y(t)), \qquad \text{(A.2.1)}$$

where y may stand for either P or Φ. In the following we describe the strategy used to solve this equation by the backward Euler method with variable stepsizes. The tolerance *TOL* determines the local accuracy of the time integration.

Suppose the approximation y^n to $y(t_n)$ has been computed already. We then perform the two steps

$$y^{n+1} = y^n + \tau f(t_{n+1},y^{n+1}), \qquad \text{(A2.2)}$$

$$y^{n+2} = y^{n+1} + \tau f(t_{n+2},y^{n+2}), \qquad \text{(A2.3)}$$

after which we estimate the local truncation error, check whether it is sufficiently small, and calculate a new stepsize τ.

To perform the two steps (A2.2), (A2.3) we have to solve twice an algebraic system of the form

$$y = \bar{y} + \tau f(\bar{t},y) \qquad \text{(A2.4)}$$

with unknown y. In the (ψ,p)-formulation (A2.4) is a linear system. In the (ψ,ϕ)-formulation the function f is nonlinear, and then the systems are solved by the modified Newton scheme

$$Y_{j+1} = Y_j + \Delta Y_j \ , \ \Delta Y_j = (I-\tau J)^{-1}(-Y_j+\bar{y}+\tau f(\bar{t},Y_j)) \qquad \text{(A2.5)}$$

for $j=0,1,\cdots$. The starting vector Y_0 is found by linear extrapolation of the two most current y^k vectors. The matrix J is a Jacobian matrix $f'(t_k,y^k)$ which is held fixed as long as the Newton process converges. Initially, at $n=0$, we put $J=f'(t_0,y^0)$ and $Y_0=y^0$. If $\|\Delta Y_j\|_\infty \leqslant 1/10\ TOL$ with $0 \leqslant j \leqslant 2$ we consider the Newton iteration to be converged and $y=Y_{j+1}$ is accepted. Otherwise, if j becomes larger than 2, we restart the iteration with a new Jacobian $J=f'(t_n,y^n)$ and with Y_0 equal to the last computed Y_j. (To avoid overflow the same is done if $\|\Delta Y_j\|_\infty > 10$). If we still obtain no convergence with this new Jacobian the current stepsize τ is decreased by a factor 10 and (A2.2), (A2.3) are computed again with this new stepsize. All arising linear systems of algebraic equations are solved by the NAG routines *F01LEF* (factorization) and *F04LEF* (forward-backward substitutions).

Now having computed the approximations y^{n+1},y^{n+2} to $y(t)$ at $t_{n+1}=t_n+\tau, t_{n+2}=t_n+2\tau$ we estimate the error in the L^2-norm. The local truncation error of Euler's method is proportional to $\tau^2\|y''(t)\|+0(\tau^3)$. Therefore the quantity

$$EST = \|y^n-2y^{n+1}+y^{n+2}\|$$

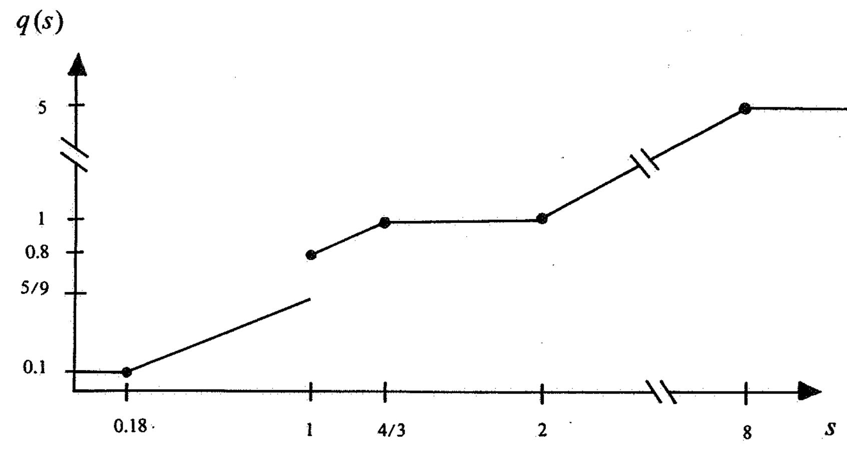
is a good measure for this local error. Moreover no extra computations are involved since y^n,y^{n+1} and y^{n+2} are known. If

$$EST \leqslant TOL$$

the steps (A2.2), (A2.3) are accepted, otherwise rejected. Our aim is to get *EST* near *TOL*. This would happen (approximately) with the stepsize $\tau_{\text{new}}=[TOL/EST]^{\frac{1}{2}}\tau$. Instead we take

$$\tau_{\text{new}} = q([TOL/EST]^{\frac{1}{2}})\tau$$

with q the piecewise linear function



This q has been chosen so as to avoid unnecessary changes in the stepsize. (Each new stepsize requires a new factorization of $I-\tau J$). Moreover the rate of change is bounded to avoid instabilities, and the new stepsize is a bit pessimistic to increase the chance that the next steps are accepted. With this new stepsize we compute y^{n+3}, y^{n+4} if the last step has been accepted, and y^{n+1}, y^{n+2} if it was rejected.

The above error control is based on absolute errors, suited for the (ψ,ϕ)-formulation. In the (ψ,p)-formulation the Newton correction ΔY_j and EST are treated in a relative sense (cf. (5.17)).

A2.2. The BELL-scheme

The diagonally implicit Runge-Kutta method given by (4.2) is implemented in a similar way as the backward Euler method. Instead of (A2.2), (A2.3) we now perform the steps

$$y^{n+\gamma} = y^n + \frac{1}{2}\gamma\tau f(t_n, y^n) + \frac{1}{2}\gamma\tau f(t_{n+\gamma}, y^{n+\gamma}), \tag{A2.6}$$

$$y^{n+1} = \frac{-1}{\gamma(\gamma-2)} y^{n+\gamma} + \frac{(\gamma-1)^2}{\gamma(\gamma-2)} y^n + \frac{\gamma-1}{\gamma-2}\tau f(t_{n+1}, y^{n+1}). \tag{A2.7}$$

The algebraic systems to be solved have again the form (A2.4) and we solve them as with the backward Euler method, except for the stopping criterion $\|\Delta Y_j\|_\infty \leq \frac{1}{10} TOL$, which was replaced by $\|\Delta Y_j\|_\infty \leq \frac{1}{500}\, TOL$ (see section 5.5).

The local truncation error of this method is given by

$$C\tau^3 \|y'''(t)\| + 0(\tau^4)\,,\ C = (3\gamma^2-4\gamma+2)/(12(\gamma-2))\,.$$

This is estimated by

$$EST = \frac{3\gamma^2-4\gamma+2}{6\gamma(\gamma-1)}\, \|2y^n + \tau f(t_n,y^n) - 2y^{n+1} + \tau f(t_{n+1}, y^{n+1})\|$$

and we obtain a new stepsize by

$$\tau_{\text{new}} = q([TOL/EST]^{1/3})\tau\,,$$

with q as before. The other aspects of the implementation are the same as with the backward Euler method.

Appendix 3. On the accuracy of the total charge values $C(t)$

Consider equation (3.3). From this equation the total charge values are obtained after substitution of the approximations P_j^n, i.e.,

$$C^n = \sum_{j=1}^{m} \left(\frac{h_j + h_{j-1}}{2}\right) P_j^n . \tag{A3.1}$$

Suppose that P_j^n is computed by integrating (2.9) with the backward Euler rule (same arguments apply to the other integration formulas). Then, like in the derivation of (3.4), (3.8) from (3.3), we find

$$\begin{aligned} \frac{C^{n+1} - C^n}{\tau} &= \sum_{j=1}^{m} \left(\frac{h_j + h_{j-1}}{2}\right) \frac{P_j^{n+1} - P_j^n}{\tau} \\ &= \sum_{j=1}^{m} \left(A_{jj-1} P_{j-1}^{n+1} + A_{jj} P_j^{n+1} + A_{jj+1} P_{j+1}^{n+1}\right) \\ &= -\frac{\mu}{\alpha h_0} \left(P_1^{n+1} - P_0^{n+1}\right) , \end{aligned} \tag{A3.2}$$

which is the backward Euler discretization of the differential equation (3.8). Consequently, one can say that even when (A3.1) is used for output, that the error in C has two sources, namely the time error in P_1 and the discretization error of the backward Euler rule for the differential equation (3.8). Apparently, both errors are small in the experiments corresponding to the Tables 5.6 - 5.9.