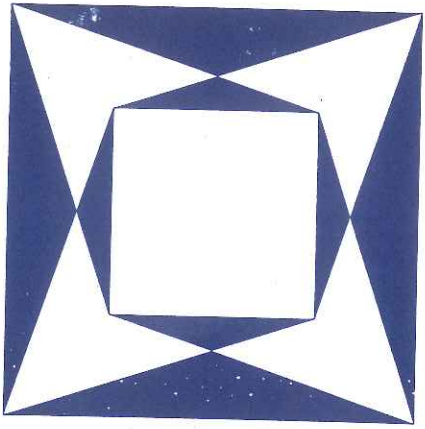
IRISH MATHEMATICAL SOCIETY



BULLETIN

NUMBER 30 MARCH 1993
ISSN 0791-5578

IRISH MATHEMATICAL SOCIETY
BULLETIN

EDITOR: Dr James Ward
ASSOCIATE EDITOR: Dr Rex Dark
BOOK REVIEW EDITOR: Dr Michael Tuite
PROBLEM PAGE EDITOR: Dr Phil Rippon
PRODUCTION MANAGER: Dr Mícheál Ó Searcóid

The aim of the Bulletin is to inform Society members about the activities of the Society and about items of general mathematical interest. It appears twice each year, in March and December. The Bulletin is supplied free of charge to members; it is sent abroad by surface mail. Libraries may subscribe to the Bulletin for IR£20.00 per annum.

The Bulletin seeks articles of mathematical interest written in an expository style. All areas of mathematics are welcome, pure and applied, old and new. The Bulletin is typeset using TEX. Authors are invited to submit their articles in the form of TEX input files if possible, in order to ensure speedier processing.

Correspondence concerning the Bulletin should be addressed to:

Irish Mathematical Society Bulletin
Department of Mathematics
University College
Galway
Ireland

Correspondence concerning the Problem Page should be sent directly to the Problem Page Editor at the following address:

Faculty of Mathematics
Open University
Milton Keynes, MK7 6AA
UK

The Irish Mathematical Society acknowledges the assistance of EOLAS, The Irish Science and Technology Agency, in the production of the Bulletin.

CONTENTS

THE IRISH MATHEMATICAL SOCIETY

Officers and Committee Members

President	Dr B. Goldsmith	Department of Mathematics Kevin Street College Dublin
Vice-President	Dr D. Hurley	Department of Mathematics University College Cork
Secretary	Dr G. Ellis	Department of Mathematics University College Galway
Treasurer	Dr D. A. Tipple	Department of Mathematics University College Dublin

Committee Members:

Dr R. Timoney, Dr E. Gath, Dr J. Pulé, Dr G. Lessells, Dr B. McCann, Dr P. Mellon, Dr C. Nash, Dr M. Ó Searcóid, Dr J. Ward.

Local Representatives

Cork	RTC	Mr D. Flannery
	UCC	Dr M. Stynes
Dublin	DIAS	Prof. J. Lewis
	Kevin St.	Dr B. Goldsmith
	DCU	Dr M. Clancy
	St. Patrick's	Dr J. Cosgrave
	TCD	Dr R. Timoney
	UCD	Dr F. Gaines
	Tallaght	Dr E. O'Riordan
Dundalk	RTC	Dr J. Harte
Galway	UCG	Dr R. Ryan
Limerick	MICE	Dr G. Enright
	UL	Dr R. Critchley
	Thomond	Mr J. Leahy
Maynooth		Prof. A. G. O'Farrell
Waterford	RTC	Mr T. Power
Belfast	QUB	Dr D. W. Armitage

NOTES ON APPLYING FOR I.M.S. MEMBERSHIP

1. The Irish Mathematical Society has reciprocity agreements with the American Mathematical Society and the Irish Mathematics Teachers Association.

2. The current subscription fees are given below.

Institutional member	IR£50.00
Ordinary member	IR£10.00
Student member	IR£4.00
I.M.T.A. reciprocity member	IR£5.00

   The subscription fees listed above should be paid in Irish pounds (puint) by means of a cheque drawn on a bank in the Irish Republic, a Eurocheque, or an international money-order.

3. The subscription fee for ordinary membership can also be paid in a currency other than Irish pounds using a cheque drawn on a foreign bank according to the following schedule:

   If paid in United States currency then the subscription fee is US$18.00.
   If paid in sterling then the subscription fee is £10.00 stg.
   If paid in any other currency then the subscription fee is the amount in that currency equivalent to US$18.00.

   The amounts given in the table above have been set for the current year to allow for bank charges and possible changes in exchange rates.

4. Any member with a bank account in the Irish Republic may pay his or her subscription by a bank standing order using the form supplied by the Society.

5. The subscription fee for reciprocity membership by members of the American Mathematical Society is US$10.00.

6. Subscriptions normally fall due on 1 February each year.

7. Cheques should be made payable to the Irish Mathematical Society. If a Eurocheque is used then the card number should be written on the back of the cheque.

8. Any application for membership must be presented to the Committee of the I.M.S. before it can be accepted. This Committee meets twice each year.

9. Please send the completed application form with one year's subscription fee to

The Treasurer, I.M.S.
Department of Mathematics
University College
Dublin
Ireland

Minutes of the Meeting of the Irish Mathematical Society

Ordinary Meeting

22nd December 1992

The Irish Mathematical Society held an ordinary meeting at 12.15 pm on Tuesday 22nd December at the DIAS, 10 Burlington Road. Fifteen members were present. The president, R. Timoney, was in the chair.

1. The **minutes** of the meeting of 16th April 1992 were approved and signed.

2. **Matters arising:** R. Timoney remarked that the Euromath project is progressing satisfactorily.

3. **Bulletin:** The March 1992 issue has been sent to members, and the December 1992 issue is at the printers. R. Dark and M. Ó Searcóid were congratulated on their good work. Contributors to the Bulletin are encouraged to use the format files produced by M. Ó Searcóid (details of which are given in the Bulletin).

4. **European Mathematical Society:** B. Goldsmith circulated a report of the EMS Council meeting held in July 1992. He has a more detailed version of the minutes which has not been circulated. S. Dineen was also present at the Council meeting. R. Timoney reported on the EMS Congress in Paris. He noted that the IMS's responses to various questionnaires appeared at the congress. The next EMS Congress is in Budapest in 1996. The IMS made three late and unsuccessful nominations for representatives to the Human Capital and Mobility scheme. The IMS pays an annual membership fee of ECU300 to the EMS. **The membership fee for an individual is £11 and should be sent to B. Goldsmith by 1st March 1993.**

5. **Treasurer's business:** Next year's Cork Operator Theory Conference and the Galway/St. Andrews Group Theory Conference will both receive £300 support from the IMS. The possibility of further funding for these conferences will be discussed at the March 1993 Committee meeting. Applications for support for conferences in 1994 must be received by 1st December 1993. This will be announced in the Bulletin. The treasurer will present a detailed report in March.

6. **Constitution:** D. Tipple and M. Ó Searcóid explained the need to change the constitution of the IMS, and circulated a new draft constitution. A copy of a modified version of this draft will be sent to all members of the Society at least one month before the next ordinary meeting, and will be put to the vote at that meeting.

7. **Elections:** The following were elected, unopposed, to the Committee (* denotes re-election):

Committee member	Proposer	Seconder
B. Goldsmith* (President)	R. Timoney	C. Nash
D. Hurley* (Vice-President)	M. Ó Searcóid	S. Dineen
G. Lessells	D. Hurley	G. Ellis
B. McCann*	B. Goldsmith	P. Mellon
M. Ó Searcóid*	B. Goldsmith	G. Ellis
J. Pulé	D. Tipple	M. Ó Searcóid
R. Timoney*	N. Buttimore	C. Nash

The following have one more year of office:
G. Ellis (Secretary), E. Gath, P. Mellon, C. Nash, D. Tipple (Treasurer).
The following have left the Committee:
F. Gaines, F. Holland.

8. **Leaving Certificate points:** The president had received a letter from the IMTA asking for discussions with the IMS regarding Leaving Certificate points. He replied, suggesting that the IMTA meet himself and D. Hurley. No meeting has yet taken place. It was felt that the matter should be pursued. More generally, it was felt that the Committee should try to encourage closer liaisons with the IMTA.

9. **AOB:** S. Dineen offered to organize the 1994 September Meeting at UCD.

The meeting closed at 1.00 pm.

Graham Ellis
University College
Galway

Conference Announcement

IRISH MATHEMATICAL SOCIETY 1993 MEETING

The annual Irish Mathematical Society Meeting will be held on 2-3 September 1993 at University College Cork. At this time, the following invited speakers have agreed to attend and the titles of their talks are appended:

J. W. Bruce, Department of Mathematics, University of Liverpool. "Whatever happened to calculus?"

Rod Gow, Department of Mathematics, University College Dublin. "Integral lattices and their automorphism groups"

Frank Hodnett, Department of Mathematics and Statistics, University of Limerick. "The thermocline equations and aspects of the dynamics of the North Atlantic Ocean"

Gerard J. Murphy, Department of Mathematics, University College Cork. "Toeplitz operators"

J. Philip O'Kane, Department of Civil and Environmental Engineering, University College Cork. "Mathematicians - do we need them?"

Philip J. Rippon, Department of Mathematics, Open University. "A Mandlebrot set for piecewise linear mappings"

J. Brian Twomey, Department of Mathematics, University College Cork. "The teaching and presentation of mathematics" (short lecture followed by discussion)

At a later stage we will advertise the full list of speakers and their talk titles over the MATHDEP@IRLEARN electronic bulletin board. As well as invited speakers, we also solicit a limited number of submitted talks, each of 20 minutes duration. If you are interested in presenting such a talk, please send a title and abstract (100-200 words) to the organizers by 31 May. We will let

you know by 15 June if we have been able to schedule your talk at the Meeting.

We plan to open the Meeting at approximately 1100 on Thursday 2 September and to close it at approximately 1600 on 3 September, so that participants travelling from other Irish institutions may only need to spend one night away from home.

Information on local accomodation is available from the organizers.

If you plan to attend this Meeting, please inform the organizers by 15 August, sending your name, affiliation and expected dates of arrival and departure.

Please communicate with either of us preferably by email. We can also be reached by post at Department of Mathematics, University College Cork, Cork, Ireland.

Telephone: + 353 - 21 - 276871 extension 2540 (Department secretary) Fax: + 353 - 21 - 272642 (include "Mathematics Department" in address page).

Pat Fitzpatrick (fitzpat@bureau.ucc.ie)
Martin Stynes (stynes@bureau.ucc.ie)

A THEORETICAL BASIS FOR PADÉ APPROXIMATION

Patrick Fitzpatrick

Abstract: The theory of Gröbner bases of polynomial ideals and modules has opened up new horizons in computational commutative algebra and algebraic geometry. We review this theory briefly and show how it leads to a new interpretation of the construction of (multivariable) Padé approximants as minimal elements in Gröbner bases. One of the more interesting aspects of this interpretation is its application to (1-variable) Padé approximation over a finite field, which is the key step in decoding the well-known classes of BCH and Goppa codes, normally carried out using the Berlekamp-Massey algorithm or the extended Euclidean algorithm. This leads to a new theoretical derivation for a decoding algorithm, which is—in its practical implementation—equivalent to that based on the extended Euclidean algorithm.

1. Introduction—Gröbner bases of ideals

The main difficulty in passing from the 1-variable polynomial ring $k[x]$ to the multivariable ring $k[x_1, \ldots, x_n]$ is that there is no longer a uniquely specified division algorithm. In fact, it is no longer clear what is meant by a quotient and a remainder and whether or not these are well defined. In $k[x]$, division is based on successive comparison of the leading term of the divisor with that of the dividend/remainder—it is clear what these leading terms are and we implicitly use an ordering of monomials based on degree. In $k[x_1, \ldots, x_n]$ many monomial orders (defined more

This article is the text of a lecture given by the author at the September meeting of the Irish Mathematical Society held at the Regional Technical College, Waterford, September 3–4, 1992.

precisely below) are possible and each has its own division algorithm.

For example, consider ordering the monomials using *lex* order, that is, lexicographically, and let us take $x > y > z$. Then dividing $x^3 + x^2y^2 + yz$ by $x + y$ (we work over $\mathbf{Q}$ unless otherwise stated) gives

$$x^3 + x^2y^2 + yz = (x + y)(x^2 + xy^2 - xy - y^3 + y^2) + y^4 - y^3 + yz.$$

On the other hand using *gradlex*—or *graduated lexicographic*—order, that is, using total degree first and ordering lexicographically the monomials of the same total degree, we obtain

$$x^2y^2 + x^3 + yz = (x + y)(xy^2 - y^3) + y^4 + x^3 + yz.$$

In both cases the algorithm stops because the leading monomial of the divisor does not divide the leading monomial of the remainder.

Remark. In the second case we could continue a little further by moving the y^4 term to the remainder and carrying out a further division based on comparison of the leading x of the divisor with the x^3 term of the remainder to give

$$(x + y)(xy^2 - y^3 + x^2 - xy + y^2) + y^4 - y^3 + yz.$$

This difficulty is intimately related to the ideal membership problem. In $k[x]$ each ideal I is principal, that is, it can be generated by a single element g say, written $I = (g)$. Thus the division algorithm solves the ideal membership problem: by a simple argument $f \in I$ if and only if the remainder on division of f by g is 0. In $k[x_1, \ldots, x_n]$ ideals are not usually principal (although by Hilbert's Basis Theorem they all have finite generating sets which we indicate by writing $I = (g_1, \ldots, g_r)$), and the monomial order plays a crucial role. For example, suppose to investigate the membership or otherwise of a polynomial f in the ideal I we divide successively by the generators g_j of I, determining the order of division by the leading monomial of g_j. Then we can derive seemingly contradictory equations as in the following example.

Example 1. In $\mathbf{Q}[x, y, z]$ we have

$$x^3 + 2xyz + xy + y = x(x^2 + yz) + y(xz + x + 1)$$
$$2xyz + x^3 + xy + y = 2x(yz + x^2) + 0(xz + x + 1)$$
$$-x^3 + xy + y$$

where the first equation—based on gradlex with $x > y > z$—indicates that the polynomial on the left is in the ideal $(x^2 + yz, xz + x + 1)$, while the second—based on gradlex with $z > y > x$—seems to imply that it is not.

These difficulties were resolved by B. Buchberger [1] by the introduction of what he called *Gröbner bases* of polynomial ideals (in honour of his supervisor W. Gröbner who had suggested to him the problem of finding constructively a multiplication table for the quotient ring $k[x_1, ..., x_n]/I$ and indicated a possible avenue of exploration). The existence of such bases—although not their construction—had already been discovered independently a year earlier by H. Hironaka [8] who called them *standard bases*. Since the early '70s their theory and applications have received wide attention and Gröbner basis routines are now implemented in all the major computer algebra packages.

Essentially, Buchberger focussed on the set of leading terms of the ideal I in question, where the leading term $\mathrm{Lt}(p)$ of a polynomial p is the greatest monomial of p under the chosen monomial order. This monomial order $<$ can be varied—and different Gröbner bases of I will result—but it must have certain properties, namely, it must be compatible with the multiplication so that if α, β, γ are monomials and $\alpha < \beta$ then $\alpha\gamma < \beta\gamma$, and also it must be a well-ordering (equivalently, $1 < \alpha$ for every monomial α). The set of leading terms of (non-zero) polynomials in I is denoted $\mathrm{Lt}(I)$ and it generates an ideal $(\mathrm{Lt}(I))$. The existence of a finite basis for $(\mathrm{Lt}(I))$ may be established using Dickson's Lemma (cf. [3]) so there exist $p_1, \ldots, p_s \in I$ such that $(\mathrm{Lt}(I)) = (\mathrm{Lt}(p_1), \ldots, \mathrm{Lt}(p_s))$. Now it is clear that if $\{g_1, \ldots, g_r\}$ is a basis of I then $(\mathrm{Lt}(g_1), \ldots, \mathrm{Lt}(g_r)) \subseteq (\mathrm{Lt}(I))$ but the reverse inclusion is not always true as the example above shows. There—using

gradlex with $x > y > z$—we have $I = (g_1, g_2) = (x^2 + yz, xz + x + 1)$ so $(\mathrm{Lt}(g_1), \mathrm{Lt}(g_2)) = (x^2, xz)$, whereas $(\mathrm{Lt}(I))$ contains $\mathrm{Lt}(zg_1 - xg_2) = \mathrm{Lt}(yz^2 - x^2 - x) = yz^2$ which is not in (x^2, xz). The definition of a Gröbner basis is precisely that this reverse inclusion should hold, that is, $\{g_1, \ldots, g_r\}$ is a *Gröbner basis* of I if $(\mathrm{Lt}(g_1), \ldots, \mathrm{Lt}(g_r)) = (\mathrm{Lt}(I))$. Moreover, it can be shown that if $\{g_1, \ldots, g_r\}$ is a subset of I such that $(\mathrm{Lt}(g_1), \ldots, \mathrm{Lt}(g_r)) = (\mathrm{Lt}(I))$ then indeed $\{g_1, \ldots, g_r\}$ is a basis—*a fortiori* a Gröbner basis—of I. (In this approach Hilbert's Basis Theorem is derived as a corollary of Dickson's Lemma.)

Henceforth we write GB for Gröbner basis. The ideal membership problem is solved completely by GBs: $f \in I$ if and only if f has remainder 0 under division by a GB of I. By division here we mean successive reduction of f by multiples of the generators based on comparison of the leading terms of the GB with the leading terms of the dividend/remainder. The defining property of the GB ensures that such a reduction is always possible when the remainder is in I. In the example above with gradlex and $z > y > x$, $(yz + x^2, xz + x + 1, x^3 - 2xy - y)$ is a GB for $(yz + x^2, xz + x + 1)$ and the division algorithm now gives

$$2xyz + x^3 + xy + y = 2x(yz + x2) + 0(xz + x + 1)$$
$$-1(x^3 - xy - y)$$

showing the polynomial on the left hand side to be in the ideal as required.

The construction of GBs—more about this later—is (unfortunately!) computationally complex in the general case and a great deal of research has gone into finding improvements to Buchberger's original algorithm, for example by studying the effects of changing the monomial order used. Examples are known however, that, no matter what refinements are introduced, will always take up large amounts of time and/or space because of expansion in the degrees of the polynomials in the basis or in the coefficients of the polynomials involved in the intermediate computations. This has not deterred the use of GBs in practice since it is believed that the constructions are "on the average" (and particularly when

only two or three variables are involved) much less complex than the worst case.

A number of other fundamental problems in commutative algebra and algebraic geometry may be solved algorithmically using GBs (cf. [2]). Among these are the determination of whether or not a system of polynomial equations has finitely or infinitely many solutions (or none at all) and the constructive evaluation of these solutions in the finite case, the construction of the elimination ideals $I \cap k[x_1, \ldots, x_j]$, $1 < j < n$, the implicitization problem (elimination of parameters), and the construction of bases of syzygy modules. It is this latter application that interests us here.

2. Gröbner bases of modules, syzygies and Padé approximants

We consider submodules M of the free module R^r where $R = k[x_1, \ldots, x_n]$. Each such module has a finite basis and the theory of GBs can be extended in a natural way. The set of *terms of length* r (replacing the monomials) is

$$T_r = \{(0, \ldots, 0, \alpha_j, 0, \ldots, 0) : \alpha_j \text{ is a monomial}\}.$$

If $<$ is a monomial order then we define a *term order* $<$ on T_r by $(0, \ldots, \alpha_j, \ldots, 0) < (0, \ldots, \alpha_l, \ldots, 0)$ if $\alpha_j < \alpha_l$ or if $\alpha_j = \alpha_l$ and $j < l$. In fact, we require something slightly more general, namely, let $w = (\psi_1, \ldots, \psi_r)$ be any *weight vector* where the ψ_j are monomials and let $<$ be a monomial order. Then the term order $<_w$ on T_r *induced* from $<$ and w is defined by the relation $(0, \ldots, \alpha_j, \ldots, 0) < (0, \ldots, \alpha_l, \ldots, 0)$ if $\psi_j\alpha_j < \psi_l\alpha_l$ or if $\psi_j\alpha_j = \psi_l\alpha_l$ and $j < l$. The terms form a vector space basis of R^r. Henceforth, for definiteness, we shall use gradlex with $x_1 < \ldots < x_n$ as our chosen monomial order.

We are particularly interested in modules of syzygies: given a set of polynomials $G = \{g_1, \ldots, g_r\}$, the module of syzygies of G is defined as

$$\mathrm{syz}(G) = \left\{ (h_1, \ldots, h_r) \subseteq R^r : \sum_{j=1}^{r} h_j g_j = 0 \right\}.$$

In fact the construction of a GB from a given basis

$$G = \{g_1, \ldots, g_r\}$$

of the ideal $I \subseteq R$ proceeds by calculating certain "S(yzygy)-polynomials" $\alpha g_i + \beta g_j$, namely, those that eliminate the leading terms of the pairs of polynomials g_i, g_j. These are then appended to the given basis and it was Buchberger's original contribution to prove that this procedure eventually terminates with a basis G' in which all these S-polynomials may be expressed with certain restrictions on the coefficients. This property is equivalent to the defining property of a GB given above and thus it turns out that the construction of the GB $G' = \{g_1, \ldots, g_r, g_{r+1}, \ldots, g_t\}$ for $I = (g_1, \ldots, g_r)$ actually produces, in addition, a GB for $\mathrm{syz}(G)$ under the term order *induced* from the monomial order in R and the weight vector $(\mathrm{Lt}(g_1), \ldots, \mathrm{Lt}(g_r))$. For more details see Möller and Mora [12].

We need one final piece of terminology: if ϕ is a monomial and I is an ideal then ϕ is said to be *reduced* modulo I if $\phi \notin (\mathrm{Lt}(I))$. Also, a polynomial p is reduced modulo I if each of its monomials is reduced modulo I. It is easy to see that if G is a GB for I then each polynomial $f \in R$ can be reduced using G to a polynomial p which is reduced modulo I. This is just the remainder on division of f by G provided that the division algorithm is extended—as in the remark at the beginning of section 1—to allow reduction *as far as possible* by every element g of G, by comparing the leading term of g with every monomial in the remainder rather than just the leading monomial.

Turning now to the problem of constructing Padé approximants we observe that this is a special case of solving for the pair (a, b) the congruence

$$a \equiv bh \bmod I \qquad (*)$$

where h is a given polynomial and I is a given ideal. For the purposes of this exposition we restrict to the case that I is a *monomial ideal* (that is, generated by monomials). The polynomial h is derived by various means (such as Taylor expansion) from some more or less known function u and the classical 1-variable

Padé approximation problem is to derive (a, b) such that the quotient a/b agrees with the expansion h of u as far as terms of a certain degree $m-1$ say, where restrictions are placed on $\deg(a)$ and $\deg(b)$ so that $\deg(a)+\deg(b) < m$. This may be interpreted as the solution of $(*)$ where $I = (x^m)$. In the classical theory a great deal of attention is (justifiably) paid to questions of convergence, but here we ignore such considerations altogether and deal only with the construction problem. One of the most interesting aspects of $(*)$ from our point of view is that in the 1-variable case it may be solved using the following theorem.

Theorem 1 (cf. McEliece [11], Theorem 8.5, p.177). *Let a, b, h be polynomials satisfying*

$$a \equiv bh \bmod x^m$$

and suppose that $\deg(a)+\deg(b) < m$. Then in the extended Euclidean algorithm applied to h and x^m giving a sequence of remainders r_j, two sequences of auxiliary polynomials u_j, v_j, and a sequence of equations

$$u_j h + v_j x^m = r_j$$

there is a unique index k and a polynomial c such that

$$a = cr_k, \quad b = cu_k.$$

Thus the construction of Padé approximants is completely solved in this case by the extended Euclidean algorithm. Of course, this does not make sense unless computations in the field k are exact—so, for example, it makes no sense to consider using this method for Padé approximation using a machine representation of the real numbers. (For example what is the degree of the polynomial $10^{-10}x+1$, if the computer only has 8 decimal places of precision?) However, in another case of interest, congruence $(*)$ arises in the context of decoding BCH, Reed-Solomon and Goppa error correcting codes: h is the *syndrome* polynomial, b is the *error locator* polynomial and a is the *error evaluator* polynomial

(for Goppa codes x^m is replaced by the Goppa polynomial), and there the computations—over a finite field—*are* exact.

An alternative to this method of solution in the 1-variable case is to use the Berlekamp-Massey algorithm (cf. [10]); for the relationships between the extended Euclidean algorithm, the Berlekamp-Massey algorithm and linear recurring sequences see [4], [6]). The Berlekamp-Massey algorithm was generalized to n variables by Sakata [13].

In [7], we gave a generalization of the Euclidean algorithm method by interpreting the solution of $(*)$—for arbitrary I—as a minimal element in a GB of a certain syzygy module. We outline this method in the next section, noting that because of the relative complexity of computing GBs this provides a *theoretical "basis"* for Padé approximation rather than a new practical method. However, in the 1-variable case our method turns out *in practice* to lead to an algorithm equivalent to that based on the extended Euclidean algorithm—we shall return to this point in Section 4. Moreover, in the context of multivariable codes and Goppa geometric codes, there are grounds for believing that our techniques may be valuable in the search for a general decoding algorithm alternative to that based on Sakata's extension of the Berlekamp-Massey algorithm.

3. Changing the term order

Further details for this section may be found in [7].

Let $\{g_1, \ldots, g_r\}$ be a GB for I and consider the set

$$F = \{-1, h, g_1, \ldots, g_r\}$$

which is clearly a GB for R (since it contains a scalar multiple of 1). We may assume that h is reduced modulo I. Each solution of $(*)$ corresponds to an equation

$$a(-1) + bh + \sum_{j=1}^{r} c_j g_j = 0,$$

in other words to a syzygy on F. The algorithm for constructing a GB (in this case verifying that the set is a GB) gives a

basis for $\mathrm{syz}(F) \subseteq R^{r+2}$ relative to the term order on T_{r+2} induced by the chosen monomial order $<$ and the weight vector $(1, \mathrm{Lt}(h), \mathrm{Lt}(g_1), \ldots, \mathrm{Lt}(g_r))$. This basis consists of the elements $\{(h, 1, 0, \ldots, 0), (g_j, 0, \ldots, 0, 1, 0, \ldots 0), 1 < j < r\}$, where the second vector has a 1 in the $j+2$ place. These are just the "obvious" syzygies that one would write down immediately; what is important is that they form a GB.

Now write M for the submodule of R^2 formed by the solutions (a, b) of $(*)$. Then by projection on the first two places we find that M has a GB $\{(h, 1), (g_j, 0), 1 < j < r\}$ under the term order on T_2 induced by $<$ and $w = (1, \mathrm{Lt}(h))$. Moreover, it can be shown that $(h, 1)$ is the unique element of least leading term (namely, $(0, 1)$) under this order. Again to simplify the exposition we now concentrate on the case that I is generated by all the monomials of total degree m. Thus $I = (x_1^m, x_1^{m-1}x_2, \ldots, x_{n-1}x_n^{m-1}, x_n^m)$ and we observe that the given basis is a GB of I. Let the total degree $\tau(p)$ of a polynomial p be defined as the maximum of the total degrees of its monomials. One example of the sort of restriction that may be placed on a, b is the following *total degree condition:*

$$\tau(a) < k,\ \tau(b) < l,$$

where k, l are non-negative integers and $k + l < m$. Then the following theorem is a special case of [7], Theorem 2.4.

Theorem 2. *Suppose that $(*)$ with I generated by monomials of total degree m has a reduced solution (a, b) with a, b relatively prime and satisfying the total degree condition above and let $w = (x_n^l, x_n^k)$. Then (a, b) is the minimal reduced solution relative to the term order induced by $<$ and w (uniquely defined up to a scalar multiple). A scalar multiple of (a, b) appears in any GB of M under this order.*

(Here a *reduced* solution is one in which both a and b are reduced modulo I and a *minimal* solution is one of least leading term. Thus to calculate the required solution (a, b) it is only necessary to convert the known GB $\{(h, 1), (g_j, 0)\}$ to a GB relative to the term order $<_w$ and pick out the minimal element.)

We end this section with two examples. The first is a 1-variable calculation derived from Knuth [9], Exercise 4, p. 515, while the second shows the method at work in $\mathbf{F}_2[x, y]$ where $\mathbf{F}_2$ is the field of 2 elements.

Example 2. Let $h = 7x^3 + 3x^2 + x + 1$ in $\mathbf{Q}[x]$. Then there are essentially four Padé approximants (a, b) to h modulo $I = (x^4)$, namely,

$$(h,\ 1),\quad (-2x^2 + 4x - 3,\ 7x - 3),\quad (x - 1,\ x^2 + 2x - 1),$$
$$(1,\ -2x^3 - 2x^2 - x + 1).$$

According to the Theorem these may be determined by finding minimal elements in GBs of the module M generated by $\{(h, 1), (x^4, 0)\}$ relative to the term orders adapted to the weight vectors $(1, x^3)$, (x, x^2) (equivalently $(1, x)$), (x^2, x) (equivalently $(x, 1)$) and $(x^3, 1)$, respectively.

Example 3. Let $h = xy^3 + x^4 + y^3 + xy^2 + x^3 + x^2 + y + x$ in $\mathbf{F}_2[x, y]$. Suppose that (a, b) exists as the Padé approximant for h relative to the ideal I generated by the monomials of total degree 5, where a and b are restricted to have total degree (at most) 2. Then we seek a GB for M under the term order $<_w$ induced from $<$ and $w = (y^2, y^2)$ (equivalently $(1, 1)$). This is just the well-known *term-order–position* order (note $x < y$)

$$(1, 0) <_w (0, 1) <_w (x, 0) <_w (0, x) <_w \ldots$$

Converting the basis $\{(h, 1), (g_j, 0), 1 < j < r\}$ to a basis relative to this order we obtain

$$\{(x^3y + x^2y + x^3,\ x^3 + x^2),\ (0,\ x^4),\ (x^4 + x^2y + x^3,\ x^2),$$
$$(xy + y + x,\ y^2 + x + 1),\ (xy^2 + x^3,\ x^3 + xy + x^2),\ (0,\ x^3y),$$
$$(y^3 + x^2y + xy + y + x,\ y^3 + x^2y + xy + x + 1)\}$$

in which it is clear that the fourth element is the desired minimum. Hence

$$\frac{xy + y + x}{y^2 + x + 1} \equiv xy^3 + x^4 + y^3 + xy^2 + x^3 + x^2 + y + x \bmod I.$$

4. Solution of the key equation

We return now to the 1-variable version of $(*)$ taking $I = (x^{2t})$. The congruence now takes the form of what Berlekamp called the "key equation" for decoding a t-error correcting BCH (or Reed-Solomon, or Goppa) code. By its construction the solution module M contains an element (ω, σ) where σ (the error evaluator polynomial) and ω (the error locator polynomial) are relatively prime and $\sigma(0) = 1$, $\delta\sigma \leq t$, $\delta\omega < \delta\sigma$. Since we have a total degree condition this element is just the minimal element (unique up to a scalar multiple) in a GB of M under the term order induced by $<$ (ordinary degree ordering among the monomials) and the weight vector (x^{2t}, x^{2t-1}) (equivalently, $(x, 1)$). The calculations to convert the known basis $\{(h, 1), (x^{2t}, 0)\}$ to a GB under this term order are identical to those which would be carried out in the Euclidean algorithm applied to h and x^{2t}, which means that we have derived a new theoretical foundation for this algorithm.

It is in fact possible (cf. [5]) to develop the theory in this 1-variable case, without using the full machinery of GBs and thus to derive a justification for the algorithmic solution of the key equation which is (in our opinion) more intuitive and natural than those based on the Berlekamp-Massey or extended Euclidean algorithm.

References

[1] B. Buchberger, An algorithm for finding a basis for the residue class ring of a zero-dimensional polynomial ideal (German). Ph. D. Thesis, Univ. Innsbruck (Austria), 1965.

[2] B. Buchberger, *Gröbner bases: an algorithmic method in polynomial ideal theory* in Multidimensional Systems Theory, N. K. Bose (ed.), Reidel: Dordrecht, 1985, 184–216.

[3] L. E. Dickson, *Finiteness of the odd perfect and primitive abundant numbers with n distinct prime factors,* Amer. J. of Math. **35** (1913), 413–426.

[4] J. L. Dornstetter, *On the equivalence between Berlekamp's and Euclid's algorithms,* IEEE Trans. Info. Thy. **IT-33** (1987), 428–431.

[5] P. Fitzpatrick, *A new algorithm for decoding BCH and Goppa codes using Gröbner bases of polynomial modules* (1992) (submitted for publication).

[6] P. Fitzpatrick and G. H. Norton, *The Berlekamp-Massey algorithm and linear recurring sequences over a factorial domain* (1990) (submitted for publication).

[7] P. Fitzpatrick and J. Flynn, *A Gröbner basis technique for Padé approximation,* J. Symbolic Computation **13** (1992), 133–138.

[8] H. Hironaka, *Resolution of singularities of an algebraic variety over a field of characteristic zero I, II,* Annals of Math. **79** (1964), 109–326.

[9] D. E. Knuth, The Art of Computer Programming: Vol. 2 Seminumerical Algorithms (2nd edn.). Addison-Wesley: Reading, Mass., 1981.

[10] S. Lin and D. J. Costello Jr., Error control coding: fundamentals and applications. Prentice-Hall: Englewood Cliffs, NJ, 1983.

[11] R. J. McEliece, The theory of information and coding. Addison-Wesley: Reading, Mass., 1987.

[12] H. M. Möller and F. Mora, *New constructive methods in classical ideal theory,* J. Algebra **100** (1986), 138–178.

[13] S. Sakata, *Extension of the Berlekamp-Massey algorithm to n dimensions,* Information and Computation **84** (1990), 207–239.

Patrick Fitzpatrick,
Department of Mathematics,
University College,
Cork.

FORMAL METHODS OF SOFTWARE DEVELOPMENT ADVANCES AND RETREATS

D. C. Ince

Abstract: This paper is about the use of discrete mathematics within software development. It describes, in outline, how discrete mathematics can be used to specify large computer systems, and how mathematical proof can be used to validate a system. This area of computer science is exceptionally promising, but is prevented by major problems from being adopted on industrial software projects. The paper examines one problem: that of data refinement and outlines one possible solution. It concludes by briefly examining the advances that have been made in formal methods of software development and also looking at where progress has been slow.

1. Introduction

Modern software development projects are normally organized on a phase-by-phase basis. One popular model is shown in Figure 1. Here the development process is split up into a number of separate activities, with each activity delivering a document which then forms the input into the next activity. The activities shown are:

Requirements analysis. This is the process of eliciting the requirements of a system from a customer. The requirements will be a mixture of functions: descriptions of what a system is intended to do, and constraints: statements which

This article is the text of an invited lecture given by the author at the September meeting of the Irish Mathematical Society held at the Regional Technical College, Waterford, September 3–4, 1992.

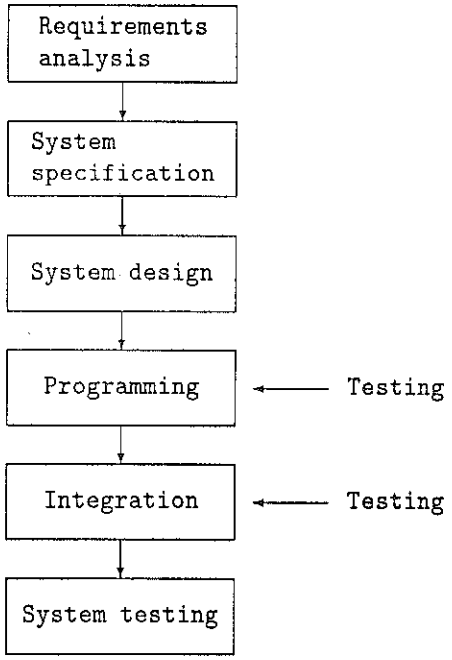


Figure 1: Conventional software development

constrain the system to be produced, or the process of developing the system. An example of the former is a constraint that a certain response time is required; while an example of the latter is the fact that the developer should use some particular programming language.

System specification. This is the process whereby the properties of the system which were discovered during the process of requirements analysis are written down. The document which is produced by this task is known as the *system specification*, although sometimes it is referred to as the *requirements specification*. Normally it is expressed in natural language.

System design. This is the activity in which the system specific-

ation is used to guide the process of deriving an overall system architecture. The architecture being expressed in terms of modules (subroutines, procedures, programs).

Programming. This is the process of taking the individual modules defined during system design and expressing them in some form of programming language.

Integration. The process of bringing together the programmed modules to form a final system.

Accompanying these development tasks is a set of parallel tasks which have the aim of validating the system: checking that user requirements are encapsulated in the system, and that individual software tasks such as integration have been carried out carefully. Examples of such tasks are: system testing, acceptance testing and module testing. This is the model of development that has been used for over twenty five years. However, the documents that are used for software development and which are generated by developers can be very flawed, leading to budget overruns, time overruns and even the cancellation of projects. In order to look at the problems which occur with these documents it is worth looking at the system specification.

2. Problems with the system specification

Although I am using the system specification as an example it is worth stressing at this juncture that similar problems occur with all the document produced by a software project. Life would be uncomplicated for the software developer if the system specification consisted of a series of sections marked

- Functional requirements;
- Non-functional requirements;
- Goals;
- Data requirements;
- Implementation and design directives;

each of which were consistent, unambiguous, and complete and where the text would be expressed in user terms. Unfortunately, this very rarely happens. The purpose of this section is to outline how reality deviates from the ideal.

In general, a system specification will be vague, contradictory, incomplete, and will contain functional requirements, constraints, and goals randomly mixed at different levels of abstraction. Often, it will either have a very naïve and over-ambitious view of the capabilities of a software system or a view which was current a few decades ago.

Vagueness

A system specification can be a very bulky document and to achieve a high level of precision consistently is an almost impossible task. At worst it leads to statements such as

> The interface to the system used by radar operators should be user friendly.
>
> The virtual interface shall be based on simple overall concepts which are straightforward to understand and use and which are few in number.

The former is at too high a level of abstraction and needs to be expanded to define requirements for help facilities, short versions of commands, and the text of user prompts. The latter is a platitude and should be removed from the specification.

Contradiction

A system specification will often contain functional and non-functional requirements which are at variance with each other. In effect they eliminate the solution space of possible systems. Typically, the sentences that make up the contradictions will be scattered throughout the document. An extreme example of such a contradiction is the statement

> The water levels for the past three months should be stored on magnetic tape.

(which may form part of the hardware requirements of a future system) and the statement

> The command PRINT-LEVEL prints out the average water levels for a specified day during the past three

> months. The response of the system should be no longer than three seconds.

Obviously, if a slow-storage medium such as magnetic tape is used then the response time will hardly be in the range of a few seconds.

A more subtle error occurs with the statements

> Data is deposited into the employee file by means of the WRITE command. This command takes as parameters: the name of the employee, the employee's department, and salary.
>
> The ENTRY-CHECK command will print on the remote printer the name of each employee together with the date on which the employee's details were entered in the employee file.

which are functional requirements together with the non-functional requirement

> The hardware on which the system will be implemented consists of: an IBM PC with 512k store, asynchronous I/O ports, keyboard, monitor, and 20 Mb hard disc.

Here the assumption made is that the employee file will contain an entry date for each employee. Unfortunately, the WRITE command does not take an entry date as a parameter, and the hardware specified does not include a description of a calendar/clock.

A system cannot be developed which satisfies contradictory requirements. If this were regarded as a pure example of a contradiction, then the ENTRY-CHECK command should be deleted. However, the contradiction could have arisen from a set of incomplete requirements. In this case the WRITE command should be amended to take the entry date as a parameter or the hardware requirement expanded to include a calendar/clock.

Incompleteness

One of the most common faults in a system specification is incompleteness. An example of this follows. It shows part of the functional requirements of a system to monitor chemical reactor temperatures.

> The system should maintain the hourly temperatures from sensors which are attached to functioning reactors. These values should be stored for the past three months.
>
> The function of the AVERAGE command is to display on a VDU the daily temperature of a reactor for a specified day.

These statements look correct. However, what happens if a user types in the AVERAGE command with a valid reactor name but for the current day? Should the system treat this as an error? Should it calculate the average temperature for the hours *up to* the hour during which the command was entered. Alternatively, should there be an hour threshold below which the command is treated as an error and, above which, the average temperature for the current day is displayed?

Mixed requirements

Very rarely will you find functional requirements partitioned neatly into functional requirements, non-functional requirements, and data requirements. Often statements about a system's function are intermixed with statements about data that is to be processed.

Naïveté

Another common failing of a system specification is that it will contain naïve views of what a computer system can achieve. This will be manifested in two ways. First, the statement of requirements will contain directives and statements which underestimate the power of the computer. The most frequent transgressors seem to be electronic engineers with little experience of software who insist on hardware requirements which could be easily satisfied by software at a much lower cost.

Another example of customer naïveté occurs in system specifications for systems which can never be built within budget. Such systems are normally specified because of the low technical expertise of the customer. The most common example of requirements for an impossible system is the specification of a particular hardware configuration and a set of functions which will never meet its performance requirements.

Another example of naïveté occurs when a customer suffers from a grossly ambitious view of what a system is capable of. One consequence of the recent rise in artificial intelligence has been a rash of system specifications which make the predictions of the wilder members of the artificial intelligence community seem almost sage-like.

Ambiguity

Specifications written in natural language will almost always contain ambiguities. Natural language is an ideal medium for novels and poetry; indeed, its success depends on the large number of meanings that can be ascribed to a phrase or a sentence. However, it is a very poor medium for specifying a computer system with precision. Some examples of imprecision are

> The operator identity consists of the operator name and password; the password consists of six digits. It should be displayed on the security VDU and deposited in the login file when an operator logs into the system.
>
> When an error on a reactor overload is detected the *error1* screen should be displayed on the master console and the *error2* screen should be displayed on the link console with the header line continuously blinking.

In the first statement does the word 'it' refer to the password or the operator identity? In the second statement should both consoles display a blinking header line or should it only be displayed on the link console?

Mixtures of levels of abstraction

A system specification will contain statements which are at different levels of detail. For example, the requirement

> The system should produce reports to management on the movement of all goods to and from all warehouses.

and the requirement

> The system should enable a manager to display, on a VDU, the cash value of all goods delivered from a specific warehouse on a particular day. The goods should be summarized into the categories described in section 2.6 of this document.

are at different levels of abstraction. The second requirement forms part of the first requirement. In a well-written statement of requirements the document should be organized into a hierarchy of paragraphs, subparagraphs, subsubparagraphs, etc. Each level of paragraph represents a refinement of the requirements embodied in the next higher level of paragraph. In a poorly written statement of requirements connected requirements will be spread randomly throughout the document.

3. Mathematics and the software project

The problems outlined above have prompted the software engineering community to look for better notations and methods for the main phases of the software project. The research that has been carried out has had two flavours. The first has involved the invention of graphical notations and software tools for such notations — tools known as analyst or designer workbenches. The second thrust has been in the area of developing mathematical notations. Good introductions to these notations can be found in [2] and [9].

The development of formal methods can be essentially seen as a reaction to the vagaries of natural language, and many of the proponents of such methods will cite the fact that the semantics of mathematics is exact. However, there is much more. My claim that formal methods has a part to play within software development is based on its modelling properties. System specifications

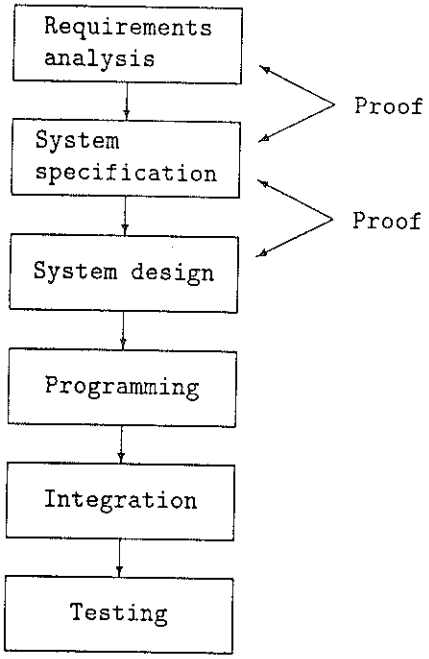


Figure 2: Formal software development

are notoriously cluttered documents, and mathematics enables all the clutter associated with the task of system specification to be removed. The way in which we use formal methods in a software project is shown in Figure 2. It closely mirrors the model put forward in Section 1 of this paper. Requirements analysis is an informal process so it is still carried out in the same way. The difference comes with system specification and design, where a mathematical notation is used to describe a system. Programming remains the same as before. Where the biggest difference is seen is in validation, where mathematical proof is used to check that the system design is a correct reflection of the system specification, and that the program code is a correct reflection of

the system design. Also, mathematical proof is used to explore the consistency of the system specification; for example, in the specification of an editor the analyst would demonstrate mathematically that when an insert command is followed by a delete command which removes the text added by the insert command, the document that is being edited returns to its original state.

It is worth pointing out that system and acceptance testing are still carried out within projects that use formal methods, however our limited experience with formal methods seems to suggest that the amount of reworking that occurs because of a failure of a system or acceptance test is drastically reduced, and the number of system and acceptance tests that fail is also reduced.

Before looking in a little detail at an example of a formal method it is worth stating some of the current problems:

- The size of the proofs that have to be carried out are very large. The mathematics that is produced is quite shallow, but there is quite a lot of it. For an example of the volume of mathematics that is generated see [1].
- There are few tools in existence that effectively support the formal development process. This is a serious problem given the amount of mathematics that has to be carried out.
- The customer has major problems understanding a formal specification.
- The mathematical abilities of many software development staff is not sufficiently sophisticated to use formal methods. To use discrete mathematics as a specification medium requires a high degree of facility in proof, and also the possession of modelling skills which many analysts, designers and programmers do not possess. I would regard this problem as the most serious, and the reason why, I suspect, formal methods will have limited use on the software projects of the future.

4. An introduction to mathematics on the software project

This section has a two aims First, it is a tutorial introduction to the use of mathematics on the software project. Second, it

provides a glimpse of some of the research that is being carried out into reducing the amount of proof that is required with formal methods. It describes a way of validating a design against a system specification which seems to be an improvement over previous methods — although it is still a research question as to how much an improvement can be achieved.

Before describing the mathematics it is worth stressing that I am describing one flavour of formal method known as a model-based method. There are other formal methods which are available, many of these are described in [2], however, model-based techniques have had the most industrial penetration.

4.1 The example

The example that I shall use is small, however it is rich enough to illustrate many of the principles of formal software development and some of the problems. It also represents a realistic piece of software which is used in a variety of systems.

The example is a symbol table handler. A symbol table is a collection of items which are stored and maintained in a computer system. Normally a symbol table will contain no duplicates and will have items added and removed from it during the operation of a system. Symbol tables are used everywhere in computing, for example, they are used in communications systems to keep track of calls, they are used in personnel systems to hold the names of staff employed by a company and they are used in computer operating systems to keep track of the users of the system.

I shall make a number of assumptions in writing down a specification of the symbol table:

- That four operations are required: an operation that adds an item to the symbol table, an operation that removes an item from a symbol table, an operation that returns with the number of items in the table, and an operation which checks that an identifier is stored in the symbol table.
- That the items in a symbol table will be called *identifiers*.
- That no more than $MaxIds$ identifiers are allowed in a symbol table.

- That no duplicates are allowed in the symbol table.

4.2 The specification

In writing down a model-based formal specification of the symbol table three pieces of mathematics are needed: *the state* a description of the stored data of the symbol table, a *data invariant* a predicate which describes the invariant properties of the state, and the four operations on the state.

The state is very simple. Since the only property of the symbol table is that no duplicates are allowed then a set can model the symbol table

$$SymTable : \mathbb{P}\ identifiers$$

where $\mathbb{P}$ is the power set operator. All this states is that $SymTable$ will be a set which contains identifiers. The data invariant is also quite simple. The only property that can be referred to in the data invariant is that the symbol table will contain no more than $MaxIds$ identifiers.

$$\#SymTable \leq MaxIds$$

where $\#$ is the set cardinality operator. The four operations are described by a pre-condition and a post-condition. A *pre-condition* is a predicate which must be true for an operation to be defined. A *post-condition* is a predicate which describes what happens when an operation is completed. An example of the use of these predicates is shown below in the specification of the operation $AddIdent$ which adds an identifier to the symbol table.

$$AddIdent(s : identifiers)$$
$$\text{pre}\quad s \notin SymTable \wedge \#SymTable < MaxIds$$
$$\text{post}\quad SymTable' = SymTable \cup \{s\}$$

The pre-condition states that the identifier s which is to be added to the symbol table must not already be in the table, and that there is room for the identifier in the table. The post-condition

shows the addition of s to the table, $SymTable'$ stands for the value of the symbol table after the operation has been completed. This, then, is the formal specification of the *AddIdent* operator in terms of mathematical structures which can be reasoned about. The specifications for the remaining operations are shown below, *RemoveIdent* removes an identifier from the symbol table, *NumIdent* returns with the number of items in the symbol table and *InTable* returns true if an identifier is in the symbol table.

$RemoveIdent(s : identifiers)$
pre $s \in SymTable$
post $SymTable' = SymTable \setminus \{s\}$

$NumIdent(s : identifiers)n : \mathbb{N}$
pre true
post $n = \#SymTable \wedge SymTable' = SymTable$

$InTable(s : identifiers)b : Boolean$
pre true
post $b \equiv s \in SymTable \wedge SymTable' = SymTable$

$\setminus$ stands for set subtraction and N is the set of natural numbers. The pre-condition for *NumIdent* is true since the operation is defined for all values of the state. The post-condition specifies that the symbol table is unaffected by the operation. The pre-condition for *InTable* is similarly true.

4.3 The development

The specification in the previous subsection represents an exact specification of a symbol table uncluttered with the noise that is so often found in industrial specifications. The next step is to transform the specification into program code. The technique I will use is known as *program calculation*. This method of development takes a specification and then uses a series of programming laws to transform that specification into program code [7]. The proponents of the method claim that it mirrors the process of algebraic manipulation used by mathematicians.

The first step in the development process is to select some computer data structure to model the symbol table. I shall use a single-dimension array with a fixed number of $MaxIds + 1$ locations. This will contain the identifiers with the last location holding a special value known as a *sentinel*. The reason for the sentinel will become clearer in the next section. This will formally be modelled by a total function $SymTableD$ which has a domain of consecutive integers from 1 to $MaxIds + 1$ and which always contains $MaxIds + 1$ elements in its range; a natural number $NumIds$ will be used to hold the number of identifiers currently represented in the state. $SymTableD$ models a single-dimensional array with bounds $1 \,..\, MaxIds+1$ that contains positive integers. The convention that I have used here is that the state which forms the design of the system is postfixed by a capital d. The data invariant for the state which is made up of both the array and $NumIds$ is

$$NumIds \leq MaxIds \wedge$$
$$\mathrm{dom}\, SymTableD = 1 \,..\, MaxIds + 1 \wedge$$
$$\#\,\mathrm{dom}\, SymTableD = MaxIds + 1$$

All the invariant states is the fact that the array will contain no more than $MaxIds$ elements, will range from 1 to $MaxIds$, and will have a fixed number of $MaxIds$ locations for identifiers.

Given this new design state how do you relate the specifications in the previous subsection to equivalent specifications in the design state? The answer is a predicate known as the *coupling invariant*. This is a predicate which characterizes the relationship between the state used in a specification and a design state. Whatever happens to the values in the specification state and the design state the coupling invariant will always hold. In the example used in this paper the coupling invariant is

$$SymTable = \mathrm{ran}(SymTableD \lhd (1 \,..\, NumIds))$$

where $\lhd$ is the domain restriction operator which forms a function by taking its first argument and restricting it to those elements which have their first element contained in the second operator.

Once the coupling invariant has been specified the next step is to use it to transform the specifications detailed in the previous subsection so that they refer to the design state. The only operation specification I shall consider is

$$
\begin{array}{l}
InTable(s : identifiers) \\
\textsf{pre} \quad \textsf{true} \\
\textsf{post} \; b \equiv s \in SymTable \land SymTable' = SymTable
\end{array}
$$

The manipulations on the other operations are roughly similar. We can use the coupling invariant to transform the post-condition to form a new operation $InTableD$ which operates on the design state.

$$
\begin{array}{l}
InTableD(s : identifiers) \\
\textsf{pre} \quad \textsf{true} \\
\textsf{post} \; b \equiv s \in \mathrm{ran}(SymTableD \lhd (1 \,..\, NumIds)) \land \\
\mathrm{ran}(SymTableD' \lhd (1 \,..\, NumIds)) \\
= \mathrm{ran}(SymTableD \lhd (1 \,..\, NumIds))
\end{array}
$$

We can now start making some more design decisions about the operation. I shall assume that the customer for the software has stated that for 98% of the time only a relatively few identifiers are examined when the $InTable$ operation is invoked. I shall assume that the program code which will eventually be produced will involve a linear search of the array $SymTableD$: the first element of the array will be examined, then the next, and so on, until either the end of the array has been reached or the identifier that is to be searched for has been found. Given this linear search strategy an efficient manipulation that can be made is that when an identifier has been found, it is moved to the first element of the array and all the remaining elements are shifted down by one. With this form of organization the most popular elements in the array for retrieval will usually be found near to the start of the array; in this way, the linear search will usually only involve a small number of elements.

In order to reflect the algorithm that will be used a number of transformations need to be applied to the post-condition of the

$InTableD$ operation, each transformation will preserve correctness. The first just uses a simple law of predicate calculus which reorganizes the equivalence.

$$
\begin{array}{l}
b \equiv s \in \mathrm{ran}(SymTableD \lhd (1 \,..\, NumIds)) \land \\
(s \in \mathrm{ran}(SymTableD \lhd (1 \,..\, NumIds)) \land \\
\mathrm{ran}(SymTableD' \lhd (1 \,..\, NumIds)) \\
= \mathrm{ran}(SymTableD \lhd (1 \,..\, NumIds))) \\
\quad \lor \\
(s \notin \mathrm{ran}(SymTableD \lhd (1 \,..\, NumIds)) \land \\
\mathrm{ran}(SymTableD' \lhd (1 \,..\, NumIds)) \\
= \mathrm{ran}(SymTableD \lhd (1 \,..\, NumIds)))
\end{array}
$$

The second disjunct in the predicate can be transformed into

$$
\begin{array}{l}
s \notin \mathrm{ran}(SymTableD \lhd (1 \,..\, NumIds)) \land \\
SymTableD' = SymTableD
\end{array}
$$

Then using a variable l which ranges in value from 1 to $NumIds$ the first disjunct can be transformed into

$$
\begin{array}{l}
s \in \mathrm{ran}(SymTableD \lhd (1 \,..\, NumIds)) \land \\
SymTableD(l) = s \land SymTableD'(1) = s \land \\
\forall i : 1 \,..\, l-1 \bullet SymTableD'(i+1) = SymTableD(i)
\end{array}
$$

since the predicate does not alter the range of $SymTableD$.

The post-condition of $InTableD$ has hence been transformed to

$$
\begin{array}{l}
b \equiv s \in \mathrm{ran}(SymTableD \lhd (1 \,..\, NumIds)) \land \\
(s \in \mathrm{ran}(SymTableD \lhd (1 \,..\, NumIds)) \land \\
SymTableD(l) = s \land SymTableD'(1) = s \land \\
\forall i : 1 \,..\, l-1 \bullet SymTableD'(i+1) = SymTableD(i)) \\
\quad \lor \\
s \notin \mathrm{ran}(SymTableD \lhd (1 \,..\, NumIds)) \land \\
SymTableD' = SymTableD
\end{array}
$$

The structure of the eventual software can now be discerned. It will consist of code which checks whether s is in the array. If s is in the array then it is moved to the front and the remainder of the array shifted back one; however if s is not in the array then the array remains unchanged.

4.4 Programming

The next stage is to transform the design specification for *InTableD* into program code. I shall use a simple programming language due to Dijkstra to express the code [3]. The structure of the program to carry out the search and possible adjustment of the array will reflect the structure of the post-condition:

Carry out search for s.
if s is in the array $\rightarrow$ adjust array
[] s is not in array $\rightarrow$ $SymTableD := SymTableD$
fi

This can be simplified to

Carry out search for s.
if s is in the array $\rightarrow$ adjust array
fi

By a process of refinement we can gradually aim towards the target of an implementation. The first part of the code: that of discovering s in the array requires a loop. It can be dealt with first. The technique that is used for this is to identify a loop invariant: a predicate which is true during the execution of the loop and which, when the loop terminates, will imply the post-condition which is required. The post-condition that we wish to satisfy is that connected with the search for s

$$b \equiv s \in \text{ran}(SymTableD \lhd (1\,..\,NumIds))$$

In order to satisfy this post-condition the first thing we do is to insert the identifier that is to be looked for in

$$SymTableD(NumIds + 1).$$

This identifier acts as a sentinel which cuts a search short. We can then develop code which establishes the post-condition

$$SymTableD(l) = s \wedge 1 \leq l \leq NumIds + 1 \wedge$$
$$\forall i : 1\,..\,l-1 \bullet SymTableD(l) \neq s$$

The code which established this post-condition will find the first occurrence of s inside the array $SymTableD$. A loop can be used for this with a loop invariant

$$1 \leq l \leq NumIds + 1 \wedge \forall i : 1\,..\,l-1 \bullet SymTableD(l) \neq s$$

The condition that has to be conjoined to the loop invariant to imply the post-condition is

$$SymTableD(l) = s$$

If we have a while loop which terminates when the condition in the while loop is false, then the loop condition is the negation of the above

$$SymTableD(l) \neq s$$

The structure of the program code now looks like

$SymTableD(NumIds + 1) := s$;
initialization for the loop
do $SymTableD(l) \neq s \rightarrow$
  loop body
od;
if s is in the array $\rightarrow$ adjust array
fi

Before the loop starts executing the loop invariant must be true. This can be achieved by initializing l to one. The loop must be driven to termination and this is achieved by having a statement $l := l + 1$ inside the loop. This code does not violate the loop invariant. Finally the predicate $b \equiv SymTableD(l) = s$ can be established by observing that since

$$l \leq NumIds \equiv s \in \text{ran}(SymTableD \lhd (1\,..\,NumIds))$$

and

$$b \equiv s \in \text{ran}(SymTableD \lhd (1\,..\,NumIds))$$

then

$b \equiv l \leq NumIds$

which can be established by the statement $b := l \leq NumIds$. The program can now be expressed as:

$SymTableD(NumIds + 1) := s;$
$l := 1;$
do $SymTableD(l) \neq s \rightarrow$
  $l := l + 1$
od;
$b := l \leq NumIds;$
if s is in the array $\rightarrow$ adjust array
fi

The incrementation of the loop does not violate the loop invariant so no more statements are required in the loop. The final part of the program code can now be derived. Since the loop invariant is true when the loop finishes we can say that the element s is in the array if b holds. This becomes the condition in the *if* statement.

The adjustment of the array requires that the post-condition

$$SymTableD(l) = s \wedge SymTableD'(1) = s \wedge$$
$$\forall i : 1 \,..\, l - 1 \bullet SymTableD'(i + 1) = SymTableD(i)$$

is established. Since the first conjunct has already been established all that is required is to satisfy the remaining two conjuncts. A loop is used for the third conjunct. A possible loop invariant involving a loop counter j is

$$1 \leq j \leq l \wedge$$
$$\forall i : 1 \,..\, j - 1 \bullet SymTableD'(l - i + 1) = SymTableD(l - i)$$

The condition which must be true in order to imply the post-condition is that $j = l$. Thus, if the loop is a while loop, the condition on the loop will be the negation $j \neq l$. The invariant is established by setting the variable j to be 1. The loop is driven to termination by incrementing j by 1. This means that

$$1 \leq j \leq l \wedge$$
$$\forall i : 1 \,..\, j \bullet SymTableD'(l - i + 1) = SymTableD(l - i)$$

$SymTableD(NumIds + 1) := s;$
$l := 1;$
do $SymTableD(l) \neq s \rightarrow$
  $l := l + 1$
od;
$b := l \leq NumIds;$
if $b \rightarrow$
  $j := 1;$
  do $j \neq l \rightarrow$
    $SymTableD(l - j + 1) := SymTableD(l - j);$
    $j := j + 1$
  od;
  $SymTable(1) := s$
fi

Figure 3: The final program

must be true for the invariant to hold after the incrementing of j. Thus, in order to re-establish the loop invariant what is required is that the statement $SymTableD(l - j + 1) := SymTableD(l - j)$ is executed. The second conjunction in the post-condition $SymTableD'(1) = s$ can be established by means of the statement $SymTable(1) := s$. Hence the code for the whole program will be that shown in Figure 3. The most obvious observation one can make about the mathematics displayed in the previous section concerns the volume. A large number of lines of set theory and predicate calculus were generated in order to derive a correct program. The nearest analogue in mathematics that I can think of is the calculation of the derivative of complicated functions from first principles. In defence a number of things can be said. First, that many of the steps that I described could be telescoped, some of the length of the development came about for didactic reasons. Second, a large number of the steps are quite simple, where it is obvious to the developer when a mistake is made. Third software tools are becoming available which enable the developer to check each step and partially automate the effort. Fourth, research on development using discrete mathematics is still in its

early stages, and the use of program calculation as a technique is still in its comparative infancy. Fifth, some very challenging algorithms have been developed using the technique described above. For example, Gries [4] has described an efficient binary fraction to decimal conversion routine, Morris has described the derivation of a pattern matching algorithm [8], and van Gasteren has described the development of a space efficient cyclic permutation algorithm [10]. Kaldewaij has collected together a number of program calculations in an advanced undergraduate textbook [5]. A comparison of the technique described in this paper and other mathematical development methods can be found in [6].

5. Advances and Retreats

It is clear that there are still many years to go before mathematical methods of software development will be used on even a relatively small proportion of our projects. However, they offer the hope of software with a very low level of faults and also offer tantalizing research problems to both the computer scientist and the mathematician. There have been many advances:

- There are now well-designed notations such as VDM [1] which are able to describe industrial software systems.
- Formal methods of software development have a secure place in the syllabus of the vast majority of British university computing degree courses.
- Some software development areas such as the safety-critical area are now beginning to realize the potential of formal methods of software development.
- The last three years has seen some excellent teaching books produced, for example [9].

However, to balance these advances there are a number of failures or areas where advance has been painfully slow:

- The penetration of formal methods of software development in the computing industry is minimal. I would estimate it as less than 1%.

- There is a lack of tools for software developers who use PC level computers. Those tools that have been developed are mainly confined to very powerful workstations.
- The tools that are available are experimental, and tend not to scale up to industrial size systems. Many such tools tend to be not very powerful theorem provers.
- Formal design is still a void. One solution has been described in this paper. However, although it seems to offer quite an improvement over current formal design techniques, it still requires quite a large amount of rather shallow mathematics to be generated.
- There is still a lack of integration of formal methods with other activities on the software project. For example, we know very little about the development of system tests from formal specifications.

References

[1] D. Andrews and D. Ince, Practical Formal Methods with VDM. McGraw-Hill, 1991.

[2] B. Cohen, W. T. Harwood and M. I. Jackson, The Specification of Complex Systems. Addison-Wesley, 1986.

[3] E. W. Dijkstra, A Discipline of Programming. Prentice-Hall, 1976.

[4] D. Gries, *Binary to decimal, one more time* in Beauty is our Busines., W. H. J. Feijen et al. (ed.), Springer Verlag, 1990.

[5] A. Kaldewaij, Programming: the Derivation of Algorithms. Prentice-Hall, 1990.

[6] H. J. Littek and P. J. L. Wallis, *Refinement methods and refinement calculi*, Software Engineering Journal (1992), 219–229.

[7] C. Morgan, Programming from Specifications. Prentice-Hall, 1990.

[8] J. M. Morris, *Programming by expression refinement* in Beauty is our Business, W. H. J. Feijen et al. (ed.), Springer Verlag, 1990.

[9] B. Potter, J. Sinclair and D. Till, Introduction to Formal Specification and Z. Prentice-Hall, 1991.

[10] A. J. M. van Gasteren, *Experimenting with a refinement calculus* in Beauty is our Business, W. H. J. Feijen et al. (ed.), Springer Verlag, 1990.

D. C. Ince,
Department of Computing,
The Open University,
Walton Hall,
Milton Keynes MK7 6AA,
England.

NUMERICAL SOLUTION OF CONVECTION-DIFFUSION PROBLEMS

Martin Stynes

Abstract: An overview is given of the nature of convection-diffusion problems and of some methods commonly used to solve these problems.

1. Introduction

Think of a still pond. At a point in this pond you pour a small amount of liquid dye. Approximately what shape will the dye stain take on the surface of the water as time passes? I think that we would all agree that the answer is a disc of slowly increasing radius, as the dye *diffuses* outwards from the initial point.

Consider next a more complicated situation: suppose that I replace the still pond above by a river which is flowing strongly and smoothly. What now is the shape of the dye stain?

The answer is a long thin curved wedge. This shape is the result of two physical processes: there is as before a tendency for the dye to diffuse slowly through the water, but the dominant mechanism present is the swift movement of the water, which rapidly sweeps (this is *convection*) the dye downstream. Convection alone would carry the dye along a (one-dimensional) curve on the surface; diffusion gradually spreads that curve, resulting in a wedge shape.

Physical situations such as this, where convection and diffusion are both present but convection dominates, are known

This article is the text of an invited lecture given by the author at the September meeting of the Irish Mathematical Society held at the Regional Technical College, Waterford, September 3–4, 1992.

as *convection-diffusion problems*. Convection-diffusion problems arise when modelling airflow over cars, aircraft wings and through jet engines, in weather forecasting, in the modelling of electrical currents in semiconductor devices and in many other applications. Consequently there is great interest in their analysis and numerical solution.

In this article we shall begin by discussing the nature of solutions to convection-diffusion problems. Then we move on to the construction of accurate numerical methods for the solution of these problems. Finally we outline the main tools which are used to analyse such numerical methods.

2. Structure of convection-diffusion solutions

The simplest mathematical model of a convection-diffusion problem is a two-point boundary value problem of the following form:

$$\varepsilon u''(x) + a(x)u'(x) + b(x)u(x) = f(x) \qquad \text{for} \quad 0 < x < 1, \quad (1)$$

with $u(0)$ and $u(1)$ given, where ε is a small positive parameter and a, b and f are some given functions. Here the term u'' corresponds to diffusion and its coefficient ε is small. The term u' represents convection, while u and f play the rôles of a source and driving term respectively. (For an explanation of why diffusion and convection should be modelled by second and first order derivatives respectively, see for example Spriet & Vansteenkiste [10].)

Problems of this type, where the highest order derivative has a small coefficient, are *singularly perturbed* differential equations. We begin by considering a single generic example in detail.

Example 1 Suppose that

$$-\varepsilon u''(x) + u'(x) = 1 \qquad \text{for} \quad 0 < x < 1, \qquad (2)$$

with $u(0) = u(1) = 0$ and $0 < \varepsilon << 1$. Here we have taken $-\varepsilon$ rather than ε as the coefficient of u'', since this turns out later to be more convenient; one can clearly move from either formulation to the other by multiplying the differential equation by -1.

The solution to this boundary value problem is easily seen to be

$$u(x) = x + \frac{e^{-1/\varepsilon} - e^{-(1-x)/\varepsilon}}{1 - e^{-1/\varepsilon}}.$$

It is more revealing if we write this as

$$u(x) = x - e^{-(1-x)/\varepsilon} + O(e^{-1/\varepsilon}). \qquad (3)$$

These three terms should be interpreted in the following way. The first, x, is the solution of the *initial* value problem

$$u'(x) = 1 \quad \text{on } (0,1) \quad \text{subject to } u(0) = 0. \qquad (4)$$

(This problem is obtained by formally setting ε equal to zero in (2) and taking one of the original boundary conditions.) The second term in (3) has a negligible influence on the solution when x is not near 1 (recall that ε is positive and small). It is essentially a correction to the solution of (4) which is required in order that the other boundary condition $u(1) = 0$ of the original problem be satisfied. The last term in (3) is of negligible size.

Thus from (3) we can see that a graph of $u = u(x)$ will closely approximate the straight line $u = x$ on almost all of [0,1]. When x approaches 1, the graph (while, of course, remaining continuous) suddenly departs from this straight line and plunges downwards to satisfy the condition $u(1) = 0$. We say that the graph has a *boundary layer* at $x = 1$.

This behaviour may be summarized as follows. Except on a narrow region near one of the boundaries, the solution of the original boundary value problem closely approximates the solution of an associated initial value problem.

Several further examples of this type are given in O'Riordan [8].

In two dimensions the situation is similar, as we now show.

Example 2 Consider the second order elliptic convection-diffusion problem

$$-\varepsilon \Delta u + u_x + u = f \quad \text{on } \Omega, \qquad (5)$$

with $u = 0$ on $\partial\Omega$, the boundary of Ω. Here to avoid any technical complications we assume that Ω is a bounded strictly convex region in $\mathbf{R}^2$ with smooth boundary $\partial\Omega$. We also assume that $f \in L^2(\Omega)$. As before, we take ε to be a small positive parameter. These hypotheses imply that (5) has a unique solution $u(x, y)$.

Write $\vec{\imath}$ for the unit vector in the direction of the positive x-axis and $\vec{n}$ for the outward pointing unit normal to $\partial\Omega$. Set

$$\partial^-\Omega = \{p \in \partial\Omega : \vec{\imath}.\vec{n} < 0 \text{ at } p\}$$

and

$$\partial^+\Omega = \{p \in \partial\Omega : \vec{\imath}.\vec{n} > 0 \text{ at } p\}.$$

Then, analogously to Example 1, the solution u on all of Ω, except close to $\partial^+\Omega$, is equal (modulo a little diffusion) to the solution v of the first order hyperbolic problem

$$v_x + v = f \quad \text{on } \Omega, \tag{6}$$

with initial data $v = 0$ on $\partial^-\Omega$. At $\partial^+\Omega$ the function u will have a boundary layer, i.e., close to $\partial^+\Omega$ the solution u changes rapidly in order to satisfy the boundary condition $u = 0$ on $\partial^+\Omega$.

This example in two dimensions is related to our earlier "dye in the river" problem. Think of the direction in which (6) propagates (the positive x-axis) as the direction of flow of the river. Then the first part of the previous paragraph states that the solution at any point (i.e., the presence or absence of dye at any point) *depends only on what happens almost directly upstream of that point*, which is what we observed when we stated that the dye spread as a long thin wedge.

This identification of the direction of propagation of a first order hyperbolic problem with the direction of flow of a fluid dynamics problem is often tacitly made in convection-diffusion terminology.

For further examples in two dimensions (and some graphs) see Johnson [1]. A feature which may occur in two dimensions is that a discontinuity in the boundary data on $\partial^-\Omega$ will in general cause

an *internal layer* in the solution; this is a narrow region, centred on one of the characteristics of the first order hyperbolic problem (6) - i.e, following the direction of flow - in which the solution changes rapidly. For a pictorial example of this see Johnson [1].

3. Numerical solution

In this Section we discuss some numerical methods which are commonly used to compute approximate solutions for convection-diffusion problems.

3.1 Why standard numerical methods fail

It is not immediately evident why convection-diffusion problems merit special attention from numerical analysts. After all, a problem such as Example 1 of the previous Section is a linear two-point boundary value problem. The average undergraduate numerical analysis textbook will give several methods applicable to this class of problems. (We shall refer to such standard textbook methods as *classical* methods, in order to distinguish them from methods which are designed specifically for convection-diffusion problems.) However if you try any classical method on Example 1, you will probably find that your computed solution displays wild oscillations and yields a very poor approximation of the true solution. What has gone wrong?

The answer may be found by a careful inspection of the convergence analysis of the classical method. This reveals that the accuracy of the method depends in general on the size of the greatest lower bound for the absolute value of the coefficient of the highest order derivative in the differential equation. In many problems this lower bound is not close to zero and then classical methods are often satisfactory. In the case of Example 1 however, this coefficient is $-\varepsilon$. When ε is close to zero, classical methods tend to be destabilized, resulting in the oscillations mentioned above.

We now investigate this phenomenon in more detail, in order to see how to devise methods which will not misbehave so badly.

Consider again Example 1. We shall attempt to generate an approximate numerical solution by means of a standard finite

difference method. First partition [0,1] by a uniform mesh of $N+1$ points, where N is some positive integer. That is, we set $x_i = i/N$ for $i = 0, \ldots, N$. Put $h = 1/N$.

A typical classical finite difference approach would begin by approximating

$$u''(x_i) \quad \text{by} \quad \frac{u(x_{i+1}) - 2u(x_i) + u(x_{i-1})}{h^2}$$

and

$$u'(x_i) \quad \text{by} \quad \frac{u(x_{i+1}) - u(x_{i-1})}{2h}. \tag{7}$$

(If these expressions are unfamiliar, use Taylor expansions to see that each approximation is $O(h^2)$ accurate; as h is small in practice - perhaps 0.1 at most - this approximation is sufficiently accurate for most purposes.)

For each i, we write $u^h(x_i)$ for the solution which we will compute at x_i. Then based on the above difference approximations and the differential equation (2), we compute the $u^h(x_i)$ from the following linear system of equations:

$$-\varepsilon \frac{u^h(x_{i+1}) - 2u^h(x_i) + u^h(x_{i-1})}{h^2} + \frac{u^h(x_{i+1}) - u^h(x_{i-1})}{2h} = 1,$$

for $i = 1, \ldots, N-1$, where $u^h(x_0) = u^h(x_N) = 0$. This set of equations comprises our *difference scheme*.

Writing this system of equations in matrix-vector form, it is easy to see that we obtain a tridiagonal square matrix whose i th row is

$$0 \ldots 0 \quad \left(-\frac{\varepsilon}{h^2} - \frac{1}{2h}\right) \quad \frac{2\varepsilon}{h^2} \quad \left(-\frac{\varepsilon}{h^2} + \frac{1}{2h}\right) \quad 0 \ldots 0,$$

for $i = 1, \ldots, N-1$. Its first ($i = 0$) and last ($i = N$) rows, which correspond to the boundary conditions, are $1\ 0 \ldots 0$ and $0 \ldots 0\ 1$ respectively.

At this point we introduce the reader to M-matrices. This class of matrices is frequently encountered in numerical analysis. We say that a matrix $A = (A_{ij})$ is an M-matrix iff

$$A_{ij} \leq 0\ \forall i \neq j,\ A^{-1} \textit{ exists and } (A^{-1})_{ij} \geq 0\ \forall i, j.$$

The significance of M-matrices in finite difference methods is that, loosely speaking, methods which give rise to M-matrices are stable and well-behaved.

Using some well-known results for M-matrices (see, e.g., Ortega & Rheinboldt [9]), one can quickly see that our difference scheme matrix above is an M-matrix if the nonzero off-diagonal entries are negative. This is equivalent to requiring that

$$-\frac{\varepsilon}{h^2} + \frac{1}{2h} < 0,$$

i.e., that

$$h < 2\varepsilon. \tag{8}$$

When (8) is satisfied, one expects the finite difference method to be stable and to yield an accurate approximation to the true solution of Example 1. In practice this is what happens. Also, if in practice $h >> 2\varepsilon$, then one's computed solution oscillates wildly and is useless.

Note here that if $\varepsilon = 1$ (i.e., if we no longer have a convection-diffusion problem), then (8) obviously holds, so one obtains an M-matrix and hence a stable numerical method. This is why classical methods are satisfactory for problems which are not of convection-diffusion type.

One might consider the above analysis as merely indicating that classical methods may be satisfactorily employed to solve convection-diffusion problems, provided only that the mesh is chosen so that some inequality such as (8) holds. Theoretically this is so, but from a practical viewpoint (8) asks too much. For in reality one wishes to solve two- or three-dimensional problems with, say, $\varepsilon = 0.0001$ (in fact ε is smaller in many applications). With this value of ε, a problem in two dimensions for which (8) is satisfied will in its finite difference formulation (which uses a square grid with $(N+1)^2$ mesh points) have about 25,000,000 unknowns! By usual computing standards, this is an absurdly large number. In three dimensions the situation is substantially worse (exercise: about 1.25×10^{11} unknowns).

The message here is that to get a classical method to work satisfactorily, one has to provide it with an unacceptably large number of mesh points. This restriction can be avoided by constructing methods which are specially suited to convection-diffusion problems. We now show how this may be done.

3.2 Upwinding

Our troubles above with large numbers of mesh points stemmed from the fact that our matrix became an M-matrix only when h was roughly the same size as ε, i.e., only when h was small. Looking at how the entries in our matrix are related to our chosen difference approximations to u'' and u', we are led to make the following modification to our method: instead of approximating

$$u'(x_i) \qquad \text{by} \qquad \frac{u(x_{i+1}) - u(x_{i-1})}{2h},$$

approximate it by

$$\frac{u(x_i) - u(x_{i-1})}{h}. \tag{9}$$

The motivation for this alteration is that it leads to a tridiagonal difference scheme matrix whose i th row is

$$0 \ldots 0 \quad \left(-\frac{\varepsilon}{h^2} - \frac{1}{h}\right) \quad \left(\frac{2\varepsilon}{h^2} + \frac{1}{h}\right) \quad -\frac{\varepsilon}{h^2} \quad 0 \ldots 0,$$

for $i = 1, \ldots, N-1$. As the nonzero off-diagonal terms are negative, it can be shown that this is an M-matrix, irrespective of the relative sizes of h and ε. Thus when a reasonable number of mesh points is used, this difference scheme will be stable. Consequently its computed approximation will be much closer than that of our original difference scheme to the true solution.

However in stabilizing the scheme we have paid a certain price in accuracy. As we mentioned previously, (7) is an $O(h^2)$ approximation. A similar Taylor expansion reveals that (9) is only an $O(h)$ approximation. Consequently our computed solution is not expected to be extremely accurate. It turns out to be moderately accurate outside the boundary layer but inaccurate inside this layer.

In fact to obtain accuracy *inside* boundary layers requires the construction of more complicated difference schemes. We refer the reader to O'Riordan [8] for an introduction to this topic.

The technique we described above, which consisted of replacing a centred difference approximation to a first derivative by a one-sided difference approximation, is known in the research literature as *upwinding*. This name comes from the fact that stability is achieved by taking this one-sided approximation in the upstream direction (recall the discussion after Example 2 earlier). If for example one tries instead a one-sided difference approximation in the downstream direction, this does not yield stability.

Upwinding has certain drawbacks, one of which is its mediocre degree of accuracy, as we described above. Another is the difficulty of generalizing it in a satisfactory way to problems in two or three dimensions. For this reason we now consider an alternative way of stabilizing our original difference scheme.

3.3 Artificial diffusion

We return once more to the differential equation

$$-\varepsilon u''(x) + u'(x) = 1$$

of Example 1. Suppose that we have a uniform mesh with the same notation as before. We generate a difference scheme by the following two-step procedure:

(*i*) *change* ε *to* $\varepsilon + \dfrac{h}{2}$

(*ii*) *apply our original method (i.e., centred difference approximation) to this modified differential equation.*

That is, we first modify the differential equation then apply a classical difference method. Due to step (*i*), this approach is known as the *artificial diffusion* method.

On working through the details of this procedure, one finds that it yields precisely the same difference scheme matrix as upwinding! We thus have two superficially different approaches

which turn out to have identical outcomes for our one-dimensional problem. However, unlike upwinding, (a variant of) the artificial diffusion method can readily be generalized in a satisfactory manner to two or more dimensions, as we shall see in subsection 3.4.

Now recall the differential equation of Example 2:

$$-\varepsilon(u_{xx} + u_{yy}) + u_x + u = f. \tag{10}$$

Working with a square mesh of diameter h in two dimensions, the obvious generalization of our one-dimensional artificial diffusion method would be to replace ε in (10) by $\varepsilon + h/2$ then to apply a classical method to the modified differential equation. This will give a stable method, but as we describe below, it does not cope successfully with internal layers if these are present.

Recall that internal layers are narrow regions in the interior of the domain where the solution changes rapidly. If we visualize the surface $u = u(x, y)$, then an internal layer is a steep, almost sheer cliff forming part of this surface and running in the direction of the flow across the domain Ω from $\partial^-\Omega$ to $\partial^+\Omega$.

When the artificial diffusion method is applied to a problem with an internal layer, the computed solution will not include an almost sheer cliff. Instead, a moderately steep slope will be generated (this is often described by saying that the layer has been "smeared out"). The basic reason is that the method adds diffusion in all directions, including the direction perpendicular to the internal layer, so the cliff is diffused in this direction.

3.4 Streamline diffusion

The artificial diffusion method is considerably improved if the added diffusion is confined to act only in the direction of flow and not perpendicular to this direction. This idea is the basis for the *streamline diffusion* method, which is fully described in Johnson [1]. When this method is applied to (10), the diffusion term $-\varepsilon(u_{xx}+u_{yy})$ is essentially modified to $-hu_{xx}-\varepsilon u_{yy}$. The method computes reasonably sharp internal layers.

The streamline diffusion method has a further advantage over the artificial diffusion method. As we indicated in subsection 3.3,

the artificial diffusion method is somehow closely related to upwinding, while upwinding is based on an $O(h)$ approximation to the true solution. Consequently both upwinding and artificial diffusion can at best be $O(h)$ accurate, even in parts of the solution which are distant from layers. Now, using a finite element approach, the streamline diffusion method can be generated in a manner which yields better than $O(h)$ accuracy away from layers. The analysis of this higher order accuracy is discussed later.

3.5 Cell vertex finite volume method

We close our list by briefly describing one class of *finite volume* methods. Finite volume methods are a standard tool in the aerospace industry, where extremely complex numerical problems (such as modelling the airflow over an entire aircraft) are commonplace.

Our concern here is with the *cell vertex* finite volume method. To apply this method, one first divides the domain of the differential equation into many small pieces or "cells" (intervals in one dimension, rectangles in two dimensions). Here let's consider the two-dimensional formulation. One seeks a computed solution, u^h say, which is a continuous piecewise bilinear function (i.e., bilinear on each cell). The unknowns in the problem are the values of u^h at the cell corners. One generates a system of equations in these unknowns in the following way. Let C be a typical cell. In the differential equation replace u at each occurrence by u^h, then integrate the resulting equation over C using Gauss' divergence theorem. This entails computations such as

$$\int_C u^h_x dx\, dy = \int_C \nabla \cdot (u^h, 0) dx\, dy = \int_{\partial C} u^h dy,$$
$$\int_C u^h_{yy} dx\, dy = \int_C \nabla \cdot (0, u^h_y) dx\, dy = -\int_{\partial C} u^h_y dx.$$

The resulting integrals over ∂C can be expressed in terms of the values of u^h at the cell corners; this is clear for the first such integral above, while for the second some form of differencing yields a reasonable interpretation of $\int_{\partial C} u^h_y dx$. See Mackenzie & Morton [3] for details.

The above integration over C yields one equation. You might expect *a priori* that one performs this computation over each cell, but it is a curious feature of the method that this does not in general yield the correct number of equations (i.e., the number of equations may then not match the number of unknowns). One can obtain the correct number of equations by discarding or dividing cells as needed, as described in Morton [5].

3.6 Summary

In the above subsections we have given short descriptions of some numerical methods which are suited to convection-diffusion problems. Our list is by no means exhaustive (for other approaches see, e.g., Miller [4]). No panacea currently exists; for each method, one can exhibit examples to which an application of the method yields disappointingly inaccurate results.

Any proposed method for convection-diffusion problems, if it is to have any chance of success, must somehow mimic the behaviour of the true solution discussed in Section 2. That is, away from layers its computed solution at each point should depend only on what happens in a narrow region directly upstream of that point. Each of our methods has this property to a greater or lesser extent.

4. Numerical analysis

In Section 3 we described some standard numerical methods which are suited to convection-diffusion problems. We now indicate briefly the techniques which are used to prove that such methods do indeed yield accurate numerical approximations to the true solutions of these problems.

For finite difference methods, the basic ideas are those of *inverse monotonicity* and *barrier functions*. These can be described quite simply. Suppose that the discrete linear system of equations is

$$L^h u^h = f^h,$$

where L^h is the matrix arising from the difference scheme and u^h will be our computed solution. The right hand vector f^h is known. Suppose (this is inverse monotonicity) that

$$(L^h)^{-1} \geq 0,$$

where the inequality holds for each entry in the matrix. It's of course reasonable to assume that $(L^h)^{-1}$ exists (otherwise we could not compute a solution u^h); the essential feature here is that all its entries are non-negative.

Let $\tilde{u}$ denote the restriction of the true solution u to the mesh points. The consistency error $L^h(u^h - \tilde{u})$ can be estimated by Taylor expansions. Warning: this calculation is tedious and involves a lot of careful estimation of terms!

Using this consistency error estimate, one next tries to construct a discrete function w^h which satisfies the vector inequalities $w^h \geq 0$ and $L^h w^h \geq |L^h(u^h - \tilde{u})|$. This is often not as difficult as it looks; one chooses w^h to mimic certain properties that one expects in the true solution u. The function w^h is known as a barrier function.

Finally, combining the last inequality with the inverse monotonicity property, we deduce that

$$|u^h - \tilde{u}| \leq w^h,$$

which is a satisfactory result provided that w^h is small.

A rather famous example of the use of this technique is provided by Kellogg & Tsan [2].

The main drawback to this method of analysis is the assumption of inverse monotonicity. In the context of convection-diffusion problems, this property often holds for the matrices arising from ordinary differential equations but is less frequently true for problems in two and three dimensions. Thus other analytical techniques are needed.

For the streamline diffusion method in two dimensions, various global error estimates have been proven. However these are not satisfactory since they are expressed in terms of Sobolev norms of the true solution u, which become excessively large when ϵ is small. The best local estimates available have been obtained by

Niijima [7]. He shows that, away from all layers, $O(h^{11/8}\ln(1/h))$ accuracy is achieved when ε is small. (Here h denotes the mesh diameter, just as in our one-dimensional investigations.) His approach uses finite element techniques to obtain local bounds on a discrete Green's function (this function is basically the inverse of the difference scheme matrix). Hence one can readily deduce convergence of the streamline diffusion method in regions that are not close to any layers. This analysis is very technical but it works.

Analysis of convergence of the cell vertex finite volume method has lagged far behind the application of the method. Up to now, no fully satisfactory analysis of this method has been published. The best estimates available are in Morton & Stynes [6], where a sharp convergence result for the one-dimensional case is obtained in a weighted discrete Sobolev H^1 norm. This bound is obtained by using techniques from finite element analysis. At present we are working on an extension of this result to the two-dimensional case.

References

[1] C. Johnson, Numerical solution of partial differential equations by the finite element method. Cambridge University Press: U.K., 1988.

[2] R. B. Kellogg and A. Tsan, *Analysis of some difference approximations for a singular perturbation problem without turning points*, Math. Comp. **32** (1976), 1025–1039.

[3] J. A. Mackenzie and K. W. Morton, *Finite volume solutions of convection-diffusion test problems*, Math. Comp. (to appear, Jan. 1993).

[4] J. J. H. Miller (ed.), Computational methods for boundary and interior layers in several dimensions. Boole Press: Dublin, 1991.

[5] K. W. Morton, *Finite volume methods and their analysis* in The mathematics of finite elements and applications VII. Proceedings of MAFELAP 1990, Brunel, April 1990, J. R. Whiteman (ed.), Academic Press: London, 1991, 189–214.

[6] K. W. Morton and M. Stynes, *An analysis of the cell vertex scheme* (1992) (Report Number 91/7, Oxford University Computing Laboratory).

[7] K. Niijima, *Pointwise error estimates for a streamline diffusion finite element scheme*, Numer. Math. **56** (1990), 707–719.

[8] E. O'Riordan, *Numerical methods for singularly perturbed differential equations*, Bull. Irish Math. Soc. **27** (1986), 14–24.

[9] J. M. Ortega and W. C. Rheinboldt, Iterative solution of nonlinear equations in several variables. Academic Press: New York, 1970.

[10] J. A. Spriet and G. C. Vansteenkiste, Computer-aided modelling and simulation. Academic Press: London, 1982.

Martin Stynes,
Department of Mathematics,
University College,
Cork.

GROUP PROJECT WORK AT SUB-DEGREE LEVEL

Neville T. Neill

Introduction

At a recent conference on "The Teaching of Mathematics at Third Level in Ireland" the notion of student projects was briefly discussed. The Department of Mathematics at the University of Ulster has been involved with both individual and group-based projects for a number of years and this paper attempts to summarize our experiences with students on the Higher National Diploma in Mathematics, Statistics and Computing where the regulating body, the Business and Technician Education Council (BTEC), insists on the completion of a group-based project as an integral part of the course.

Philosophy

Diplomates i.e. technicians will, at least during the initial years of employment, work as a member of a team often undertaking well defined individual tasks under a fairly rigid time-scale. The success of the team clearly depends on the motivation and effort of each individual and hence HND students must be well prepared in all aspects of group work. Since the project is the main vehicle for refinement of the various skills they have hopefully acquired during the course, it is necessary to put it into context within the overall course structure.

Structure of the Higher National Diploma

A course at Higher National level under the auspices of BTEC entails a minimum of two years full-time study. A sandwich element is encouraged and indeed the HND in Computer Studies at the Jordanstown campus incorporates a year long placement making the course of three years duration. The HND in Mathematics, Statistics and Computing with which this paper is dealing has no placement aspect and hence this course must attempt to simulate the workplace experience as closely as possible.

All BTEC courses place great emphasis on the acquisition and development of "Common Skills" [1,2] and at present the seven defined Common Skills break down into eighteen competences as given in Appendix I.

Clearly, therefore, any discussion of the project in isolation would be inappropriate and the following sections attempt to show how it becomes the natural culmination of both the academic and inter-personal skills developed during the course.

Year I

During the first year of the HND all students take a unit entitled 'Workshop'. This unit serves several purposes:

(i) It introduces the cohort to the hardware and associated operating systems they will use throughout the course. Included are terminal clusters linked to the Vax mainframe, networked 386 based DEC workstations, networked 286 based PS/2's, a NIMBUS network and various stand-alone PC's. Many students have done little or no computing prior to joining the course and it is essential that their initial anxieties at being faced with such a daunting array of new technologies are quickly allayed. During this acclimatization phase students are encouraged to learn from each other and the initial atmosphere of individual uncertainty is soon replaced by group confidence and cohesion.

(ii) It challenges the students' concepts of learning by making them responsible for the pace and depth at which they acquire knowledge. From the outset is is explained that the theme of student-centred learning is central to the whole course and that,

for approximately 25% of the entire programme, they themselves will determine their rate of progress. This is a new concept for students recently arrived from the formalized teacher/pupil relationship which exists in secondary education but, almost without exception, they relish the freedom and responsibility it brings. Computer-based tutorials with associated self-assessment tests on such topics as MS-DOS, word processing, spreadsheets and databases are worked through systematically and, while these topics are not examined explicitly, it is made clear that their usage will be assumed throughout the course.

(iii) It provides students with instruction and practice in the areas of oral communication, presentation skills and group work. The last of these is clearly crucial to the whole structure of the course and is introduced in the following manner;

(a) Each individual completes a problem-solving/decision-making exercise. This involves deciding the order of priority to be given to various actions in a difficult situation. This year the EARTHQUAKE simulation exercise [3] was used and participants must decide on their immediate and long-term actions when trapped in a building damaged in an earthquake. Groups of 4 to 6 persons are then formed and often after heated debate, a group ranking is achieved. The solutions, as provided by the experts who produced EARTHQUAKE, are then given out and individual scores are compared against the group decisions. Almost inevitably the group produces better results than the individuals who comprise it indeed this year only one student out of 26 produced a score lower than that of their group. (Note that a final score is formed as the sum of the absolute differences of the true results and the stated results. Thus the lower the score the better the correlation between the views of the experts and those of the person simulating the earthquake scenario.) This illustrates graphically the advantages of group effort, and also provides a valuable insight into the difficulties which can arise when attempting to reach a consensus decision.

(b) The students are then split into those groups with which they will work for the remainder of the term. As a preamble to

the main investigation each group was asked to produce a solution to a real-life problem. This year the problem was how to organize the Christmas party for a medium-sized shirt manufacturing company, given some personal and financial constraints. As no single solution to such a problem exists, the debate to convince other groups of the validity of a particular solution is often lively and stimulating. The point of this exercise is however not so much the solution itself but the means by which it was obtained. After the orals each group is asked to write down all situations which can prevent a group from functioning properly, with up to 10 points often being noted. Larger groups are then formed which comprise one member of each of the smaller groups. These large groups pool their comments on how groups can be disrupted thus ensuring that all possibilities are discussed. The original groups now reassemble and have, in addition to their previously agreed points, those points which they may have overlooked and have been identified by their counterparts. All groups now have a similar list of potential pitfalls.

(c) Finally each group is asked to draw up a set of rules which they themselves would adopt to ensure that any problem undertaken is tackled in a fair and structured manner. They then write down these agreed rules and sign their names to the document. This form of "learning contract" provides the framework for both the operation and ultimate assessment of the group.

At periodic intervals the group is asked to refer to its agreed rules and check whether in fact they are being followed.

(iv) The Programme of Integrative Assignments (P. I. A.) This takes up most of the time allocated to the Workshop in the second term. As the name implies the P. I. A. offers the students the opportunity to undertake one or more group-based investigations, each of which requires them to use their recently acquired knowledge from at least two of the fields of Mathematics, Statistics and Computing. Each report is word processed, bound and accompanied by an oral presentation at which all Year I students and staff of the Department are present.

In previous years two Integrative Assignments have been un-

dertaken with the groups being rearranged for the second investigation. Experience has shown however that approximately five weeks is insufficient time for the task briefing, the analysis and solution of the problem and the production of a structured report and visual aids for the oral. This year, for the first time, only one Integrative Assignment was given. The groups were given approximately eight weeks to submit their report, one week to prepare for the orals and one week for their actual delivery. This less frenzied approach has proven to be much more successful with each team now having the time to produce a more substantial solution.

The topics undertaken this year were

(a) A user guide for Statistical applications of Lotus 1-2-3.
(b) Numerical Methods via Lotus 1-2-3 and DERIVE.
(c) Production of a software package to assist with central heating installation.
(d) An investigation of random numbers.
(e) A user guide to SECMATHS and CALMAT for non-mainstream mathematics students.
(f) Student attitudes to 'The Learning Process'.

Year II

The notion of investigation is again central to the practical aspects of many of the Year II units. The unit "Mathematical Modelling via Mechanics" relies almost entirely on this form of assessment while the Numerical Methods and Statistics units also include at least one group-based piece of course work.

The Project

At the beginning of the academic year the students are informed of their groupings together with their project title and its associated supervisor. The Department has tried to effect this linkage in two quite different ways:

(i) By giving each student a list of proposed projects and asking them to be ranked in descending order of preference. As far as possible students are then assigned to their first choice project.
(ii) By forming groups on the basis of their Year I results. Each group thus contains a range of abilities and is assigned a project by the Senior Course Tutor.

The pros and cons of these approaches are self-evident namely:-

Method (i)

pro – student motivation is maximized due to their involvement in the selection of the project to be undertaken.
con – a group can consist entirely of academically quite weak students and will thus require close supervision throughout the year. The standard of such a project may well be significantly lower than that anticipated at the outset.

Method (ii)

pro – the mixture of abilities within the group means that the weaker students can learn from their more gifted peers.
con – students may be forced to undertake a project in an area in which they have little interest and hence total commitment may be somewhat lacking.

An amalgam of these two approaches seems best namely:

(a) an element of choice being given to individual students
(b) groups being organized in such a way that a balance between preferred project and academic ability is achieved.

Group size

Experience has shown that the optimum group size is four. Projects have run with as many as five students and as few as three in a group (often due to one of the original members leaving the course) but neither of these formulations is particularly suitable. Many projects have sub-tasks inherent within them and subgroups of two students tackling such tasks and reporting back to the group as a whole seems the most efficient way of addressing a given problem.

Our initially intuitive and then experiential thoughts on optimum group size have since been confirmed at conferences at which group teaching methods are discussed.

Project organization

Before commencing the project the group agree on the rules under which they will operate and sign this "learning contract". The experience gained from the Programme of Integrative Assignments in Year I is clearly invaluable in drawing up these rules and the students themselves now realize the importance of adhering to them throughout the year.

In an attempt to simulate how a similar task would be undertaken in the workplace and indeed to emphasize the importance of the project within the course some supervisors have organized their group meetings on a formal basis with each of the students in turn assuming the role of secretary or chairperson. They produce an agenda, take and then produce minutes etc. and hence monitor the progress or otherwise of the project as a whole. Meetings are normally arranged on a fortnightly basis for the first term then on a three weekly cycle as the project gets fully underway. The minutes of the meetings form an appendix of the final document.

Project titles

These have fallen mainly into four categories:

(i) Statistics — Simulation
(ii) Statistics — Data Analysis
(iii) Numerical Analysis
(iv) Package Investigation

Sample titles for each of the above are;

(i) Factors affecting the accuracy of a golf putt. Sampling frequencies to minimize economic loss. Simulation of acceptance sampling for BS6001.
(ii) Causes of muscular dystrophy using multiple linear regression analysis. The Black-Scholes method applied to traded option

forecasting. The application of AR techniques in the analysis of the relationship between stock market performance and company turnover. An analysis of customer complaints received by a software house.
(iii) Multivariate optimization with constraints using the NAG library. Development of a software package for the solution of linear simultaneous equations. Methods for unconstrained optimization. Euclid's Algorithm.
(iv) The FAMULUS package. Applications of LOGO. Computer-Assisted learning via SYMBOLATOR. LATEX user guide and applications.

Project assessment

In the early years of the project most supervisors had no experience in the assessment of group work and it was this aspect which caused most of the initial misgivings. It soon became obvious, however, that the assigning of individual marks was not a particularly difficult task due to the fact that, at the end of a twenty five week period, the supervisor was well acquainted with both the group members and their commitment to the overall project.

The formal assessment breakdown for the project is given in Appendix II.

As well as the final seminar mentioned in Appendix II, an interim presentation is given at the end of the first term. This not only informs the members of the supervisory panel of the progress made to date but also gives the students the opportunity to practice for the seminar in a real-life situation.

Self and peer assessment have not been introduced and indeed are unlikely to be included in the foreseeable future. This form of assessment is less relevant to a situation in which a member of staff has worked closely with a group and can confidently determine the respective weightings for each member of the group.

The completed project is read by both the supervisor and a second marker from within the Department. The Moderator for the HND will also examine each document to ensure comparability in standards.

Recent innovations

Many of the projects, especially those in the field of Package Investigation, are essentially practical in nature and their assessment must reflect this fact. If the objective has been to produce a user-friendly guide to a particular package then the best way to verify the success of the project is to test the guide on non-specialists and obtain their reaction. The fact that their end-product will be used in this way acts as a tremendous motivating influence on the group members who normally act as facilitators during the test session.

Examples

(i) The group working with LOGO used a subset of their final document to provide Key Stage 2 pupils from a local secondary school with a basic introduction to turtle geometry. The pupils spent a day at the University under the supervision of the group members and the event proved stimulating and rewarding for both parties. The replacement of the formal seminar presentation by such a practical session allowed the students to demonstrate both their academic and interpersonal skills in a realistic environment and give real meaning to the project as a whole.

(ii) As noted above the ultimate test of a user-guide is whether or not it enables its reader to access a particular package quickly and painlessly. These were the criteria against which the user-guide to the computer algebra package SYMBOLATOR was to be assessed and hence the formal oral presentation was replaced by a laboratory session in which engineering students were required to check their solutions to a given tutorial via SYMBOLATOR. The mathematics students acted as demonstrators and the engineering students submitted their written comments the following day. Once again the need for concise, accurate and understandable instructions, both in written and oral form, was brought home to the group all of whom found the experience very useful.

(iii) The ideas outlined above were also applied to the LATEX user-guide produced by one of this year's groups. The standard computer services documentation was somewhat less than helpful and the group was set the task of providing not

only a detailed investigation into LATEX and its applications but also a short checklist which would enable non-specialists to edit, compile, preview and print a simple LATEX program. Once again a number of "guinea-pigs" were asked to use the checklist and see whether it achieved these objectives. The students soon realized that no detail could be excluded from such a listing and it illustrated vividly the pitfalls which exist should an author be so familiar with the subject matter that the same degree of familiarity is assumed in the reader.

Summary

Group based projects have now been running at sub-degree level within the Department of Mathematics for seven years. Initial misgivings have been dispelled and they now form an integral, timetabled part of the work of the Department.

Student motivation has been, in general, excellent with most groups taking a genuine interest in their work.

The assessment process has become well established and innovative methods of assessment are being introduced whenever possible.

Appendix I

Common Skill	*Competence*
Managing and Developing Self	1. Manage own rôles and responsibilities
	2. Manage own time in achieving objectives
	3. Undertake personal and career development
	4. Transfer skills gained to new and changing situations and contexts
Working with and Relating to Others	5. Treat others' values, beliefs and opinions with respect
	6. Relate to and interact effectively with individuals and groups
	7. Work effectively as a member of a team
Communicating	8. Receive and respond to a variety of information

	9. Present information in a variety of visual forms
	10. Communicate in writing
	11. Participate in oral and non-verbal communication
Managing Tasks and Solving Problems	12. Use information sources
	13. Deal with a combination of routine and non-routine tasks
	14. Identify and solve routine and non-routine problems
Applying Numeracy	15. Apply numerical skills and techniques
Applying Technology	16. Use a range of technological equipment and systems
Applying Design and Creativity	17. Apply a range of skills and techniques to develop a variety of ideas in the creation of new/modified products, services or situations
	18. Use a range of thought processes

Appendix II: Project assessment

(a) The project must normally be submitted by the first week of the third term.

(b) The project will be assessed by
  (i) the supervisor
  (ii) another suitably qualified member of staff.
  In addition each project group will be expected to give a short seminar upon the conclusion of their project and the performance of each member will be assessed by the project supervisory panel.

(c) The project will normally be assessed in accordance with the following marks allocation;

A. Presentation and organization		30%
(i) written presentation, including layout, aim, and purpose outlined, bibliography, and index.	10%	
(ii) clarity of written project including use of English, style, spelling and punctuation etc.	10%	
(iii) oral presentation, i.e. the ability to inform non-specialists in the project area on its content.	10%	
B. Contents and Results		25%
(i) evidence that the subject has been investigated in some depth.	10%	
(ii) results, including how far the aims have been realized.	10%	
(iii) conclusions, including suggestions as to possible extensions to the project.	5%	
C. Student Understanding and Motivation		45%
(i) student initiative in obtaining and analysing relevant material.	15%	
(ii) contribution to the project as a whole.	20%	
(iii) understanding of the techniques and concepts encountered in the project.	10%	
TOTAL		100%

(d) The seminar will normally provide the mark for (iii) in part A above.

References

[1] Common Skills, Consultative Pack 1, Business and Technician Education Council, 1991.

[2] Common Skills, Consultative Pack 2, Business and Technician Education Council, 1991.

[3] EARTHQUAKE, A Team Building Simulation, Orion International Limited, 1990.

Neville T. Neill,
Department of Mathematics,
University of Ulster,
Jordanstown.

Research Announcement

THE τ_w TOPOLOGY ON SPACES OF HOLOMORPHIC FUNCTIONS

Seán Dineen

If U is a domain in a locally convex space over $\mathbf{C}$ then a seminorm p on $H(U)$, the space of complex valued holomorphic functions on U, is said to be τ_w-continuous if there exists a compact subset K of U such that for each V open, $K \subset V \subset U$, there exists $C(V) > 0$ such that

$$p(f) \leq C(V)\|f\|_V$$

for $f \in H(U)$.

The τ_w topology is the topology generated by all τ_w continuous seminorms. The τ_w topology was originally motivated by properties of analytic functions which can be represented by Borel measures supported by every neighbourhood of compact set but not by the compact set itself and at the linear level is related to the inductive dual of a locally convex space.

The τ_w topology is thus defined by a set of inequalities and in many cases it is of interest to find an explicit set of semi-norms which generated this topology. Explicit sets are known for balanced domains in Banach spaces and in Fréchet-Montel spaces where it is known that τ_w coincides with the compact open topology. Here we give an explicit set of semi-norms for a collection of Fréchet spaces which includes all Banach spaces, all Fréchet spaces with unconditional basis of type (T) and the Köthe echelon spaces.

Definition 1. *An unconditional Schauder decomposition, $\{E_n\}_n$ of a Fréchet space E is a T-Schauder decomposition if there exists a fundamental system of semi-norms for E, $(\|\cdot\|_k)_{k\in N}$ such that*

(i) $\quad \|P_J(x)\|_k \leq \|x\|_k \quad$ *all* $J \subset N, \quad k \in N$ *and* $x \in E$

(ii) for every sequence $\alpha = (\alpha_k)_k$, $0 < \alpha_k \leq 1$, there exists a partition $J_\alpha = (J_{\alpha,k})_k$ of N such that if $P_{\alpha,k} := P_{J_{\alpha,k}}$ then $\|P_{\alpha,k}(x)\|_{k-1} \leq \alpha_k \|P_{\alpha,k}(x)\|_k$ for all $x \in E$ and all $k \geq 2$.
(iii) $(\|\cdot\|_k)$ defines the topology induced by E on $P_{\alpha,k}(E)$ for all α and all k.

Fréchet spaces with a T-Schauder decomposition are a slight modification of the spaces introduced in [1] and the spaces in [1] appeared as a result of developments arising form positive solutions to Grothendieck's "Problème des topologies".

Theorem 2. *If the Fréchet spaces E has a T-Schauder decomposition then the τ_w topology on $H(E)$ is generated by all semi-norms of the form*

$$p(f) = \sum_{n=0}^{\infty} \left\| \frac{\hat{d}^n f(0)}{n!} \right\|_{B_n}$$

where $(B_n)_n$ is a sequence of compact subsets of E which converges to a compact subset of E.

We have written $\sum_{n=0}^{\infty} \frac{\hat{d}^n f(0)}{n!}$ as the Taylor series expansion of f at the origin.

Reference

[1] J. Bonet and J. C. Diaz, *The Problem of topologies of Grothendieck and the class of Fréchet T-spaces*, Math. Nachr. **150** (1991), 109–118.

Seán Dineen,
Department of Mathematics,
University College Dublin,
Belfield,
Dublin 4.

Book Review

Patterns and Waves

The Theory and Applications of Reaction-Diffusion Equations

Peter Grindrod
Clarendon Press, Oxford, 1991
ISBN 0 19 959692 8
Paperback, St£17.50

Reviewed by Martin Stynes

In Don DeLillo's entertaining novel *White Noise* [1], a central event occurs when the narrator's small U.S. town is threatened by a toxic chemical cloud. Many inhabitants of the town flee. We read [1] "We joined ... the traffic flow into the main route out of town ... the traffic moved in fits and starts".

The traffic moved in fits and starts. We all recognize this phenomenon. As in DeLillo's novel, it occurs even in the absence of traffic lights and stop signs. Apparently all that is needed to trigger the effect is a sufficient density of traffic. Why does it happen? Why does heavy traffic never seem to flow smoothly at constant (albeit low) speed, but instead is subject to speeding up, slowing down and intermittent halting?

We can answer this question after a little analysis. Suppose that the cars are travelling in the direction of the positive x-axis on an infinitely long one-dimensional road. Let $u(x,t)$ denote the car density, which depends both on position x and time t. Here $u = 0$ corresponds to an empty road and $u = 1$ corresponds to maximum congestion.

Let I be any closed and bounded interval on the x-axis. Then the car population of I is $\int_I u\,dx$. The rate at which this population increases in time is given by $\frac{\partial}{\partial t}\left(\int_I u\,dx\right)$. We assume that u is smooth, so this expression equals $\int_I u_t\,dx$.

Assume that our road has neither entrances nor exits. Then changes in the car population of I can result only from cars entering and leaving I along the road. Denoting the car velocity by $v(x,t)$, the net number entering $I = [a,b]$ is given by $-(uv)(b) + (uv)(a)$. Assuming that v is smooth, this equals $-\int_I (uv)_x\,dx$.

Equating that with our earlier formula,

$$\int_I u_t\,dx = -\int_I (uv)_x\,dx.$$

Since I was arbitrary, we conclude that

$$u_t + (uv)_x = 0.$$

It's clear that the velocity v must depend on u. We now make a plausible simplifying assumption, viz., that $v = 1 - u$. Then our differential equation above becomes

$$u_t + (u(1-u))_x = 0,$$

i.e,

$$u_t + (1-2u)u_x = 0.$$

The characteristics of this hyperbolic equation satisfy

$$\frac{dt}{dx} = \frac{1}{1-2u}.$$

The solution u is constant on each characteristic. We see that, as time passes, regions of high car density ($1/2 < u \leq 1$) will move in the direction of the negative x-axis. Furthermore, this rate of movement is least for u near $1/2$ and greatest for u near 1. Thus, as regions of higher density move backwards towards regions of lower density, the traffic tends to bunch together and a stop-go regime develops, instead of all cars proceeding at some uniform speed.

The above discussion is a partial answer to Exercise 1.5 on page 63 of the book under review. It illustrates several features of this book: its concern with the understanding and solution

of evolutionary nonlinear partial differential equations, their use in modelling phenomena from the real world, and in particular the revelation that smooth initial data can in finite time generate nonsmooth solutions in such models.

In fact, unlike our example, the book (as can be inferred from its title) deals predominately with parabolic partial differential equations. These have the form

$$u_t = \Delta u + f(u, \nabla u, \vec{x}, t).$$

Here t is time, $\vec{x}$ is position in R^n, Δu denotes the Laplacian of u with respect to the variables $\vec{x}$ and f is some nonlinear function.

The author provides a sustained gradual development of concepts and analytical solution techniques. These are introduced fairly painlessly by means of a detailed examination of examples. He succeeds in finding the middle ground between excessive detail and inadequate explanations. Nevertheless, the reader is clearly expected to use pen and paper to verify various claims. I did not check many calculations in detail, but, for example, the analysis on p. 216 contains several typographical errors. This seems to have been a momentary atypical lapse.

The presentation is nicely structured. Technical difficulties are hived off to clearly marked subsections, so the overall flow of the book is not impeded. Internal cross-referencing, both forwards and backwards, is of an exceptionally high standard. Chapter 1, which occupies about one-quarter of the book, introduces the basic ideas and techniques. The other four Chapters deal (in order of increasing complexity) with observable phenomena in the solutions of nonlinear evolutionary partial differential equations. Their titles are: 2 – Pattern formation, 3 – Plane waves, 4 – A geometrical theory for waves, 5 – Nonlinear dispersal mechanisms. Applications, mostly from physiology, biology and chemistry, are scattered throughout these pages. For example, the Belousov-Zhabotinsky cyclic chemical reaction with its spectacular spiral wave patterns is the main example discussed in Chapter 4.

The author resists the temptation to give excessive attention to side issues. Instead, adequate references are given for further study of specific topics. A significant omission is the almost

complete absence of discussion and references for the numerical solution of the problems examined in the book. This is surprising since several Figures in the book have been produced by numerical computation of approximate solutions; if the author believes (by implication) that such computed solutions are instructive, then he should make some attempt to provide guidelines to the reader who wishes to perform numerical experiments. As the differential equations under consideration are nonlinear and consequently can be difficult to analyse, a computational approach may in practice often be an attractive option.

The book is suitable for beginning postgraduate students in applied mathematics, but pays more than lip service to basic theoretical issues such as local existence criteria. It is more accessible than Smoller's "purer" text [2]. I would welcome the opportunity to teach a course with Grindrod's book as the main text, supplemented by a little material from numerical analysis.

References

[1] D. DeLillo, White Noise (Picador Edition). Pan Books: London, 1986.

[2] J. A. Smoller, Shock Waves and Reaction Diffusion Equations. Springer-Verlag: New York, 1983.

Martin Stynes,
Department of Mathematics,
University College,
Cork.

Book Review

A FIRST COURSE IN NONCOMMUTATIVE RINGS

(Graduate Texts in Mathematics)

T.Y.Lam

Springer-Verlag, 1991, xv+397 pp.
ISBN 0-387-97523-3

Reviewed by Mark Leeney

Lam's book is a welcome addition to the literature on noncommutative rings. The first book of an intended two-volume introduction to ring theory, it is a fleshed-out version of the material covered in a one-semester graduate course designed and delivered to second year graduate students at Berkeley. In the Preface, Lam convincingly pleads the case for ring theory as 'an indispensable part of the education for any fledgling algebraist', citing the connections between it and group theory, functional analysis, algebraic geometry, to name but a few. Given the ubiquitous nature of noncommutative rings it is unfortunate that the number of books written at this introductory level is so few.

As per the origins of the subject, the book opens with the Wedderburn-Artin theory of semi-simple rings and in short order presents an impressive collection of examples. These include the quaternions, free rings, rings with generators and relations, group rings, Laurent series rings, twisted polynomial rings, differential polynomial rings, the tensor algebra of a finite dimensional vector space and triangular rings. These examples form the bedrock for motivation and serve to provide the author with a presumed knowledge on the part of the student which otherwise might not be the case. This is a lot of information to be assimilated by a student and I must admit to questioning the

efficacy of introducing the newcomer to so many new 'faces' at once. Chapter 2 continues with the Jacobson radical and group rings and the J-semisimplicity problem. Modules over finite dimensional algebras, representations of groups and linear groups are considered in Chapter 3. Chapter 4 introduces the prime radical and prime/semiprime rings. The structure of primitive rings, the Density theorem, subdirect products and the commutativity theorems are also presented there. Chapter 5 embarks on division rings; along the way proving Wedderburn's, Frobenius's and Cartan-Brauer-Hua's theorems. Explicit constructions of division rings further increases the stock of examples. Tensor products, maximal subfields and polynomials over division rings are also treated. Chapter 6 follows the generalization of the Artin-Schrier criterion for the existence of orderings on fields to rings without zero-divisors. As a special case, ordering structures on division rings are considered. Chapter 7 focuses on local/semilocal rings and idempotents. Standard results on lifting idempotents modulo an ideal are developed and used to study the Krull-Schmidt decomposition of modules. The final chapter deals with perfect and semiperfect rings and homological characterizations thereof. The notion of the basic ring of a semi-perfect ring is considered and is brought to bear on right artinian rings, culminating with examples of principal indecomposable modules and Cartan matrices.

As is customary with the series, the text is well bound, uncluttered in appearance and conventional in notation; typos are conspicuous only by their absence. The style is fairly informal without being chatty. Each chapter begins with a useful introduction to the material about to be presented; how the material developed historically, its power and limitations, and the goals of the chapter.

Sets of exercises appear at the end of each section of each chapter, thereby reinforcing the material on a regular basis. Exercises are, generally speaking, straightforward tests of understanding of the concepts introduced in the text. Trickier ones are supplied with hints and serve as a useful complement to the theory and examples in the text. An especially endearing aspect of the book is Lam's occasional excursions into 'open problem' territ-

ory. At such locations, time is spent in discussing the current state of affairs relating to a problem, partial answers and possible approaches : a stimulating insight into the process of doing research-level mathematics is given.

The book, generally, has a good balance of theory and examples. However, the initial stockpile of twenty-six examples in the first twenty-three pages may prove a bit intimidating. For reference purposes (and revisits are quite likely) the convenience of having these examples en bloc will be welcome - if one gets past page twenty-three with the desire to continue still intact. The no-nonsense, go-ahead style of the book is in keeping with the rest of the series and its setting of material in historical context together with the tantalizing references and discussion of open questions should have the desired effect of exciting and encouraging the student. However, I do wonder how plausible it would be to try to cover all this material in one semester. Of the many sins a teacher can commit, two spring to mind. Of these two, the worse is to underestimate the ability of one's students - this smacks of condescension and pride. The lesser evil is to overstretch a class - this can be construed as overindulgence, nay even one-upmanship. If Lam is guilty of such a transgression, at least he has chosen the lesser evil.

Any serious mathematical library should have this book in its collection. It could be used to good purpose as the basis for a one-year excursion into the realms of noncommutative rings or, as a text for self study by mathematically mature students. Like a good meal, this first course should serve as a satisfying entrée, leaving one wanting more. It does.

Mark Leeney,
Department of Computing,
RTC Letterkenny,
Donegal.

INSTRUCTIONS TO AUTHORS

The Bulletin is typeset with TEX. Authors should if possible submit articles to the Bulletin as TEX input files; if this is not possible typescripts will be accepted. Manuscripts are not acceptable.

Articles prepared with TEX

Though authors may use other versions of TEX, It is preferred that they write plain TEX files using the standard IMS layout files. These files can be received by sending an e-mail message to `listserv@irlearn.ucd.ie`. The body of the message should contain the three lines:

```
get imsform tex
get mistress tex
get original syn
```

Instructions on the use of these is contained in the article on *Directions in Typesetting* in issue number 27, December 1991.

The TEX file should be accompanied by any non-standard style or input files which have been used. Private macros, reference input files and both METAFONT and TEX source files for diagrams should also accompany submissions.

The input files can be transmitted to the Editor either on an IBM or Macintosh diskette, or by electronic mail to the following Bitnet or EARN address:

MATWARD@BODKIN.UCG.IE

Two printed copies of the article should also be sent to the Editor.

Other Articles

Authors who prepare their articles with word processors can expedite the typesetting of their articles by submitting an ASCII input file as well as the printed copies of the article.

Typed manuscripts should be double-spaced, with wide margins, on numbered pages. Commencement of paragraphs should be clearly indicated. Hand-written symbols should be clear and unambiguous. Illustrations should be carefully prepared on separate sheets in black ink. Two copies of each illustration should be submitted: one with lettering added, the other without lettering. Two copies of the manuscript should be sent to the Editor.