A Quadratic Iteration and a 'PI' Challenge, from Mike Mudge.

Simple Iteration This is the process of calculating consecutive numbers from a formula of the type $x_{n+1} = F(x_n)$, where F is a given function, n takes the values 0, 1, 2... and x_0 is a given initial value (which is sometimes called the seed of the iteration).

Convergence For computational investigation, as distinct from a pure mathematical approach, a simple iteration will be deemed to have converged when two distinct x_i values agree to within a prescribed tolerance, eg +1 or -1 in the 12th decimal place.

Note If these two values are consecutive they are said to represent the limit (or limiting value) of the iteration. If, however, they are not consecutive, then we define a constant (or limiting) sequence to consist of one of these values together with all of the x_i which separate them.

Now, Fred Salt of Bridgend has considered the case where $F(x_n) = x_n - x_n^2 + u$, $x_0 = u$ and u is chosen from within the range 0.2 to the square root of 3.

Example Salt Iteration,

```
u = 0.318309886183791.
x1 = x0 - x0^2 + u =
     0.535298588725244,
x2 = x1 - x1^2 + u =
     0.567063895817797,
```

and so on until

```
x17 = x18 =.. =
     0.564189583547757,
```

the limiting value of the iteration.

Now, from $x_{n+1} = x_n - x_n^2 + u$ it is easily seen that the limiting value when $x_{n+1} = x_n$ is given by $x_n^2 = u$; ie, x = square root of u. The above example illustrating this, where u is approximately PI^{-1} and x_{17} etc are approximately $PI^{-1/2}$ as expected.

Experiments will show that for u less than 1 the limiting value is obtained in this way, the number of iterations required (for a given tolerance) increasing rapidly as u approaches 1.

Using a tolerance of +1 or -1 in the 15th decimal place and values of u = 0.999, 0.9999, 0.99999, 0.999998, the number of iterations are, respectively, 21389, 167464, 1214199 & 4483784. Fred executes one million iterations in about 3.5 seconds in Borland's Turbo C++ on a PC with an 80486 33MHz CPU; thus these investigaitons are quite realistic for the PC user.

There is a totally trivial result when u = 1! However, for u greater than 1 the fascination begins:

(A) If u lies between 1 and 1.48 the constant iteration sequence consists of two terms whose sum is 2, eg

```
u = 1.25 yields
 0.500000000000000 &
 1.500000000000000
```

(B) If u lies between 1.48 and 1.61 the constant iteration sequence has four terms, the sum of terms 1 and 2 and the sum of terms 3 and 4 both being 2.
eg u = 1.570796326794897 yields

```
1.820512356910358,
 0.077043442041949,
 1.641904076875175 &
0.516851406010752.
```

(C) If u lies between 1.61 and 1.639, the constant iteration sequence has eight terms and no particular sum pattern can be detected.

(D) If u lies between 1.639 and 1.6477, there are 16 terms in the iteration sequence.

Problem 1 Reproduce, extend and if possible explain the behaviour described above. (Consider the possibility of using other quadratic functions F(x), but only if this is going to assist in the understanding of the process... as the Salt Iteration alone poses a number of questions.)

Now, when the number of iterations required to obtain numerical convergence is recorded it is observed that, for certain values of u, the relationship between i, the number of iterations required to reach a constant sequence, and u, can be expressed either by: log(i) = c + m*u or by 1/i = c + m*u.

For example, in the range of u-values between 1.18 and 1.648 minima in the number of iterations required for convergence occur at several values of u. There is a maximum of 211 iterations at u = 1.632, with other maxima 1749285 at u = 1.6181 & 903037026 at u = 1.664046156.

Problem 2 Construct an in-depth study of the number of iterations required for 'numerical convergence', both as a function of u, the seed of the iterative process, and as a function of the degree of accuracy required before convergence is deemed to have occurred.

Problem 3 Consider what area of current pure (and empirical) mathematical research is bordered by this study. If possible, obtain references and/or make suggestions for further work in this area.

POST SCRIPT, a 'micro-programming' challenge involving 'PI'

Eric Adler of Bayswater, London, offers the following QBasic program for the calculation of PI; this generated 16 decimal places on a 386/387.

```
REM PROGRAM TO CALCULATE PI
B# = SQR (1/2): P# = 2*
                SQR(2)
DO WHILE B# < >1
PRINT P#
B# = SQR(1/2 + 1/2 * B#)
P# = P#/B#
LOOP
```

Problem 4 a) Explain the mathematical background to the above program.
b) Produce a shorter piece of code for the evaluation of PI.

Attempts to solve some, or all, of the above problems may be sent to Mike Mudge, 22 Gors Fach, Pwll-Trap, St Clears, Carmarthen, Dyfed SA33 4AQ, tel (0994) 231121, to arrive by 1 May 1993. Any communication arriving by the closing date will be judged, using suitable subjective criteria, and a prize will be awarded by *PCW* to the 'best' submission. Please note that material can only be returned if a stamped addressed envelope is provided.

Review, October 1992: Integer Polyhedra

Gareth Suggett observed that the problem with 'zero volume tetrahedra' is simply that of plane configurations of four points with integer separations and refers interested readers to Martin Gardener's *Mathematical Circus*, Penguin 1981 edition, page 65.

However, the very worthy prizewinner this month is Jim Duncan of 9 Ryeground Lane, Formby, Liverpool L37 7EG, using Lattice C on an Atari 1040 ST. Among the many interesting results, all accompanied by computer generated graphics, there are:

- Two distinct tetrahedra with edge lengths 3, 5, 7, 11, 13, 17, which are enantiomers, forming a mirror image pair.
- Four distinct trigonal bipyramids with edge lengths 3, 5, 7, 11, 13, 17, 19, 23, 29, forming two sets of mirror image pairs.
- Sixteen distinct rectangular base pyramids with edge lengths 3, 5, 7, 11, 13, 19, 23; eight sets of mirror image pairs etc.
- Four distinct octahedra with prime edges maximum 41 were found in about 40 minutes computing time. Details of coding are available from Jim if you wish to study his algorithm further.

Mike Mudge welcomes correspondence on any subject within the areas of number theory and computational mathematics. Particularly welcome are suggestions for future Numbers Count articles.