NUMERICAL RECIPES

Webnote No. 5, Rev. 1

Implementation of the Euler Transformation

There is an elegant and subtle implementation of Euler's transformation due to van Wijngaarden [1]: It incorporates the terms of the original alternating series one at a time, in order. For each incorporation it *either* increases p by 1, equivalent to computing one further difference (5.3.10), or else *retroactively* increases n by 1, without having to redo all the difference calculations based on the old n value! The decision as to which to increase, n or p, is taken in such a way as to make the convergence most rapid. Van Wijngaarden's technique requires only one vector of saved partial differences. Here is the algorithm:

series.h

```
struct Eulsum {
```

Convergence acceleration of an alternating series by the Euler transformation. Initialize by calling the constructor with arguments `nmax`, an upper bound on the number of terms to be summed, and `epss`, the desired accuracy. Then make successive calls to the function `next` (see below).

```
    VecDoub wksp;
    Int n,ncv;
    Bool cnvgd;
    Doub sum,eps,lastval,lasteps;

    Eulsum(Int nmax, Doub epss) : wksp(nmax), n(0), ncv(0),
        cnvgd(0), sum(0.), eps(epss), lastval(0.) {}

    Doub next(const Doub term)
```

Incorporates into `sum` the next term, with value `term`, of an alternating series. On each call `term` should have a sign opposite to that of the previous call. The flag `cnvgd` is set when convergence is detected.

```
    {
        Int j;
        Doub tmp,dum;
        if (n+1 > wksp.size()) throw("wksp too small in eulsum");
        if (n == 0) {                          Initialize:
            sum=0.5*(wksp[n++]=term);          Return first estimate.
        } else {
            tmp=wksp[0];
            wksp[0]=term;
            for (j=1;j<n;j++) {                Update saved quantities by van Wijn-
                dum=wksp[j];                       gaarden's algorithm.
                wksp[j]=0.5*(wksp[j-1]+tmp);
                tmp=dum;
            }
            wksp[n]=0.5*(wksp[n-1]+tmp);
            if (abs(wksp[n]) <= abs(wksp[n-1]))          Favorable to increase p,
                sum += (0.5*wksp[n++]);                  and the table becomes longer.
            else                                         Favorable to increase n,
                sum += wksp[n];                          the table doesn't become longer.
        }
        lasteps = abs(sum-lastval);
        if (lasteps <= eps) ncv++;
```

```
        if (ncv >= 2) cnvgd = 1;
        return (lastval = sum);
    }
};
```

Actually, Euler's transformation is a special case of a more general transformation of power series. Suppose that some known function $g(z)$ has the series

$$g(z) = \sum_{n=0}^{\infty} b_n z^n \tag{1}$$

and that you want to sum the new, unknown, series

$$f(z) = \sum_{n=0}^{\infty} c_n b_n z^n \tag{2}$$

Then it is not hard to show (see [2]) that equation (2) can be written as

$$f(z) = \sum_{n=0}^{\infty} [\Delta^{(n)} c_0] \frac{g^{(n)}}{n!} z^n \tag{3}$$

which often converges much more rapidly. Here $\Delta^{(n)} c_0$ is the nth finite-difference operator (equation 5.3.10), with $\Delta^{(0)} c_0 \equiv c_0$, and $g^{(n)}$ is the nth derivative of $g(z)$. The usual Euler transformation (equation 5.3.9 with $n = 0$) can be obtained, for example, by substituting

$$g(z) = \frac{1}{1+z} = 1 - z + z^2 - z^3 + \cdots \tag{4}$$

into equation (3), and then setting $z = 1$.

CITED REFERENCES AND FURTHER READING:

Goodwin, E.T. (ed.) 1961, *Modern Computing Methods*, 2nd ed. (New York: Philosophical Library), Chapter 13 [van Wijngaarden's transformations].[1]

Mathews, J., and Walker, R.L. 1970, *Mathematical Methods of Physics*, 2nd ed. (Reading, MA: W.A. Benjamin/Addison-Wesley), §2.3.[2]