NAG Toolbox for MATLAB

c06la

1 Purpose

c06la estimates values of the inverse Laplace transform of a given function using a Fourier series approximation. Real and imaginary parts of the function, and a bound on the exponential order of the inverse, are required.

2 Syntax

```
[valinv, errest, nterms, na, alow, ahigh, nfeval, ifail] = c06la(fun, t,
relerr, alphab, 'n', n, 'tfac', tfac, 'mxterm', mxterm)
```

3 Description

Given a function $F(p)$ defined for complex values of p, c06la estimates values of its inverse Laplace transform by Crump's method (see Crump 1976). (For a definition of the Laplace transform and its inverse, see the C06 Chapter Introduction.)

Crump's method applies the epsilon algorithm (see Wynn 1956) to the summation in Durbin's Fourier series approximation (see Durbin 1974)

$$f(t_j) \simeq \frac{e^{at_j}}{\tau}\left[\tfrac{1}{2}F(a) - \sum_{k=1}^{\infty}\left\{\mathrm{Re}\left(F\left(a+\frac{k\pi i}{\tau}\right)\right)\cos\frac{k\pi t_j}{\tau} - \mathrm{Im}\left(F\left(a+\frac{k\pi i}{\tau}\right)\right)\sin\frac{k\pi t_j}{\tau}\right\}\right],$$

for $j = 1, 2, \ldots, n$, by choosing a such that a prescribed relative error should be achieved. The method is modified slightly if $t = 0.0$ so that an estimate of $f(0.0)$ can be obtained when it has a finite value. τ is calculated as $t_{\mathrm{fac}} \times \max(0.01, t_j)$, where $t_{\mathrm{fac}} > 0.5$. You specify t_{fac} and α_b, an upper bound on the exponential order α of the inverse function $f(t)$. α has two alternative interpretations:

(i) α is the smallest number such that

$$|f(t)| \leq m \times \exp(\alpha t)$$

for large t,

(ii) α is the real part of the singularity of $F(p)$ with largest real part.

The method depends critically on the value of α. See Section 8 for further details. The function calculates at least two different values of the parameter a, such that $a > \alpha_b$, in an attempt to achieve the requested relative error and provide error estimates. The values of t_j, for $j = 1, 2, \ldots, n$, must be supplied in monotonically increasing order. The function calculates the values of the inverse function $f(t_j)$ in decreasing order of j.

4 References

Crump K S 1976 Numerical inversion of Laplace transforms using a Fourier series approximation *J. Assoc. Comput. Mach.* **23** 89–96

Durbin F 1974 Numerical inversion of Laplace transforms: An efficient improvement to Dubner and Abate's method *Comput. J.* **17** 371–376

Wynn P 1956 On a device for computing the $e_m(S_n)$ transformation *Math. Tables Aids Comput.* **10** 91–96

5 Parameters

5.1 Compulsory Input Parameters

1: **fun – string containing name of m-file**

fun must evaluate the real and imaginary parts of the function $F(p)$ for a given value of p.

Its specification is:

```
[fr, fi] = fun(pr, pi)
```

Input Parameters

1: **pr – double scalar**
2: **pi – double scalar**

The real and imaginary parts of the argument p.

Output Parameters

1: **fr – double scalar**
2: **fi – double scalar**

The real and imaginary parts of the value $F(p)$.

2: **t(n) – double array**

Each $\mathbf{t}(j)$ must specify a point at which the inverse Laplace transform is required, for $j = 1, 2, \ldots, n$.

Constraint: $0.0 \leq \mathbf{t}(1) < \mathbf{t}(2) < \cdots < \mathbf{t}(n)$.

3: **relerr – double scalar**

The required relative error in the values of the inverse Laplace transform. If the absolute value of the inverse is less than **relerr**, then absolute accuracy is used instead. **relerr** must be in the range $0.0 \leq \mathbf{relerr} < 1.0$. If **relerr** is set too small or to 0.0, then the function uses a value sufficiently larger than ***machine precision***.

4: **alphab – double scalar**

α_b, an upper bound for α (see Section 3). Usually, α_b should be specified equal to, or slightly larger than, the value of α. If $\alpha_b < \alpha$ then the prescribed accuracy may not be achieved or completely incorrect results may be obtained. If α_b is too large the function will be inefficient and convergence may not be achieved.

Note: it is as important to specify α_b correctly as it is to specify the correct function for inversion.

5.2 Optional Input Parameters

1: **n – int32 scalar**

Default: The dimension of the arrays **t**, **valinv**, **errest**. (An error is raised if these dimensions are not equal.)

n, the number of points at which the value of the inverse Laplace transform is required.

Constraint: $\mathbf{n} \geq 1$.

2: **tfac – double scalar**

t_{fac}, a factor to be used in calculating the parameter τ. Larger values (e.g., 5.0) may be specified for difficult problems, but these may require very large values of **mxterm**.

Suggested value: $\mathbf{tfac} = 0.8$.

Default: 0.8

Constraint: **tfac** > 0.5.

3: **mxterm – int32 scalar**

The maximum number of (complex) terms to be used in the evaluation of the Fourier series.

Suggested value: **mxterm** ≥ 100, except for very simple problems.

Default: 100

Constraint: **mxterm** ≥ 1.

5.3 Input Parameters Omitted from the MATLAB Interface

work

5.4 Output Parameters

1: **valinv**(**n**) **– double array**

An estimate of the value of the inverse Laplace transform at $t = \mathbf{t}(j)$, for $j = 1, 2, \ldots, n$.

2: **errest**(**n**) **– double array**

An estimate of the error in **valinv**(j). This is usually an estimate of relative error but, if **valinv**$(j) <$ **relerr**, **errest**(j) estimates the absolute error. **errest**(j) is unreliable when **valinv**(j) is small but slightly greater than **relerr**.

3: **nterms – int32 scalar**

The number of (complex) terms actually used.

4: **na – int32 scalar**

The number of values of a used by the function. See Section 8.

5: **alow – double scalar**

The smallest value of a used in the algorithm. This may be used for checking the value of **alphab** – see Section 8.

6: **ahigh – double scalar**

The largest value of a used in the algorithm. This may be used for checking the value of **alphab** – see Section 8.

7: **nfeval – int32 scalar**

The number of calls to user-supplied (sub)program **fun** made by the function.

8: **ifail – int32 scalar**

0 unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Note: c06la may return useful information for one or more of the following detected errors or warnings.

ifail $= 1$

On entry, **n** < 1,
or **mxterm** < 1,
or **relerr** < 0.0,

or **relerr** ≥ 1.0,
or **tfac** ≤ 0.5.

ifail $= 2$

On entry, $\mathbf{t}(1) < 0.0$,
or $\mathbf{t}(1), \mathbf{t}(2), \ldots, \mathbf{t}(\mathbf{n})$ are not in strictly increasing order.

ifail $= 3$

$\mathbf{t}(\mathbf{n})$ is too large for this value of **alphab**. If necessary, scale the problem as described in Section 8.

ifail $= 4$

The required accuracy cannot be obtained. It is possible that **alphab** is less than α. Alternatively, the problem may be especially difficult. Try increasing **tfac**, **alphab** or both.

ifail $= 5$

Convergence failure in the epsilon algorithm. Some values of **valinv**(j) may be calculated to the desired accuracy; this may be determined by examining the values of **errest**(j). Try reducing the range of **t** or increasing **mxterm**. If **ifail** $= 5$ still results, try reducing **tfac**.

ifail $= 6$

All values of **valinv**(j) have been calculated but not all are to the requested accuracy; the values of **errest**(j) should be examined carefully. Try reducing the range of t, or increasing **tfac**, **alphab** or both.

7 Accuracy

The error estimates are often very close to the true error but, because the error control depends on an asymptotic formula, the required error may not always be met. There are two principal causes of this: Gibbs' phenomena, and zero or small values of the inverse Laplace transform.

Gibbs' phenomena (see the C06 Chapter Introduction) are exhibited near $t = 0.0$ (due to the method) and around discontinuities in the inverse Laplace transform $f(t)$. If there is a discontinuity at $t = c$ then the method converges such that $f(c) \rightarrow (f(c-) + f(c+))/2$.

Apparent loss of accuracy, when $f(t)$ is small, may not be serious. Crump's method keeps control of relative error so that good approximations to small function values may appear to be very inaccurate. If $|f(t)|$ is estimated to be less than **relerr** then this function switches to absolute error estimation. However, when $|f(t)|$ is slightly larger than **relerr** the relative error estimates are likely to cause **ifail** $= 6$. If this is found inconvenient it can sometimes be avoided by adding k/p to the function $F(p)$, which shifts the inverse to $k + f(t)$.

Loss of accuracy may also occur for highly oscillatory functions.

More serious loss of accuracy can occur if α is unknown and is incorrectly estimated. See Section 8.

8 Further Comments

8.1 Timing

The value of n is less important in general than the value of **nterms**. Unless the user-supplied (sub)program **fun** is very inexpensive to compute, the timing is proportional to **na** $\times$ **nterms**. For simple problems **na** $= 2$ but in difficult problems **na** may be somewhat larger.

8.2 Precautions

You are referred to the C06 Chapter Introduction for advice on simplifying problems with particular difficulties, e.g., where the inverse is known to be a step function.

The method does not work well for large values of t when α is positive. It is advisable, especially if **ifail** $= 3$ is obtained, to scale the problem if $|\alpha|$ is much greater than 1.0. See the C06 Chapter Introduction.

The range of values of t specified for a particular call should not be greater than about 10 units. This is because the method uses parameters based on the value $\mathbf{t}(n)$ and these tend to be less appropriate as t becomes smaller. However, as the timing of the function is not especially dependent on n, it is usually far more efficient to evaluate the inverse for ranges of t than to make separate calls to the function for each value of t.

The most important parameter to specify correctly is **alphab**, an upper bound for α. If, on entry, **alphab** is sufficiently smaller than α then completely incorrect results will be obtained with **ifail** $= 0$. Unless α is known theoretically it is strongly advised that you should test any estimated value used. This may be done by specifying a single value of t (i.e $\mathbf{t}(n)$, $n = 1$) with two sets of suitable values of **tfac**, **relerr** and **mxterm**, and examining the resulting values of **alow** and **ahigh**. The value of $\mathbf{t}(1)$ should be chosen very carefully and the following points should be borne in mind:

(i) $\mathbf{t}(1)$ should be small but not too close to 0.0 because of Gibbs' phenomenon (see Section 7),

(ii) the larger the value of $\mathbf{t}(1)$, the smaller the range of values of a that will be used in the algorithm,

(iii) $\mathbf{t}(1)$ should ideally not be chosen such that $f(\mathbf{t}(1)) = 0.0$ or a very small value. For suitable problems $\mathbf{t}(1)$ might be chosen as, say, 0.1 or 1.0 depending on these factors. The function calculates **alow** from the formula

$$\mathbf{alow} = \mathbf{alphab} - \frac{\ln(0.1 \times \mathbf{relerr})}{2 \times \tau}.$$

Additional values of a are computed by adding $1/\tau$ to the previous value. As $\tau = \mathbf{tfac} \times \mathbf{t}(n)$, it will be seen that large values of **tfac** and **relerr** will test for a close to **alphab**. Small values of **tfac** and **relerr** will test for a large. If the result of both tests is **ifail** $= 0$, with comparable values for the inverse, then this gives some credibility to the chosen value of **alphab**. You should note that this test could be more computationally expensive than the calculation of the inverse itself. The example program (see Section 9) illustrates how such a test may be performed.

9 Example

c06la_fun.m

```
function [fr,fi] = c06la_fun(preal,pimag)
  z = complex(1,0)/complex(preal+0.5,pimag);
  fr = real(z);
  fi = imag(z);
```

```
t = [1];
relerr = 0.01;
alphab = -0.5;
[valinv, errest, nterms, na, alow, ahigh, nfeval, ifail] = ...
    c06la('c06la_fun', t, relerr, alphab, 'tfac', 7.5)
```

```
valinv =
    0.6071
errest =
    0.0037
nterms =
          18
na =
           2
alow =
   -0.0395
ahigh =
    0.0939
nfeval =
```

```
           36
ifail =
            0
```