4B−9

記号ラプラス変換

下地貞夫、小林　晃、○町田治夫

東京工科大学

1．はじめに

数式処理システムは汎用の優れたものが、いくつも発表され、使われているが、数式処理が各応用分野で十分に使われるためには、応用分野毎にきめ細かく配慮されたシステムが望ましい。そういうシステムが現れるためには、数値計算で行われているように、アルゴリズムやデータ構造やプログラミングの詳細が開示され、議論されるようになることが必要であると思われる。ここでは、数式処理技術開発の一例として、応用上重要なラプラス変換を記号的に行う試みについて報告したい。

記号ラプラス変換と逆変換は各関数に対する変換表が与えられれば、像関数が複素数の関数であることを意識する必要は全くなく、記号ラプラス変換は記号微分と、逆変換は有理関数の記号積分と同様の方法で行うことができる。

原関数	像関数
1	$\frac{1}{s}$
t^n	$\frac{n!}{s^{n+1}}$
e^{at}	$\frac{1}{s-a}$
$t^n e^{at}$	$\frac{n!}{(s-a)^{n+1}}$
$\sin\omega t$	$\frac{\omega}{s^2+\omega^2}$
$\cos\omega t$	$\frac{s}{s^2+\omega^2}$
$e^{at}\sin\omega t$	$\frac{\omega}{(s-a)^2+\omega^2}$
$e^{at}\cos\omega t$	$\frac{s-a}{(s-a)^2+\omega^2}$

表1　基本的な関数のラプラス変換

2．ラプラス変換

変換表は応用数学の教科書にあるとおりで、代表的なものを 表1 に示した。

これを次のように処理し、変換を行う。

(1) 原関数の内挿表現→前置表現への変換、

(2) 各関数に対するラプラス変換、

(3) 像関数の前置表現→内挿表現、

第(2)項は、各関数に対して原関数を表すリストと像関数を表すリストとの対応づけ、によって行う。例えば、正弦関数に対しては、(/ 1 (+ (^ s 2) 1) または (/ 1 (+ (^ s 2)(^ ω 2))) というリストを返す関数を用意して、'(sin x) か '(sin (ω t)) を入力したとき、この関数を起動するようにする。

どの関数が起動されるかは、数式リストの最初の要素によって定められる。演算子に関数定義を属性として与えておき、対応する関数を起動するようにした。

```
(mapc #'(lambda (op fun)；演算子に属性を与える。
  (setf (get op 'lapf)(symbol-function fun)))
   '(+ - * ^ sin cos exp)
   '(+lapf -lapf *lapf ^lapf slapf clapf elapf))
(defun lapf (f var1 var2)；ラプラス変換
  (cond ((atom f)
         (cond ((eq f var1)
                (list '/ 1 (list '^ var2 2)))
               (t (list '/ f var2))))
        (t (let ((lapf (get (car f) 'lapf)))
             (funcall lapf f var1 var2)))))
```

変換例：

```
>(laplc '((2 * t + 1) * (exp (a * t))) 't 's)
(2 / (S - A) ^ 2 + 1 / (S - A))

>(laplc '(t * (sin (w * t))) 't 's)
((W * 2 * S) / (S ^ 2 + W ^ 2) ^ 2)
```

2番目の例で、分子の記号の並び方が奇妙なのは、結果の整理が未だ十分でないためである。

Symbolic Laplace Transform

Sadao SHIMOJI, Akira KOBAYASHI, Haruo MACHIDA

Tokyo Engineering University

３．ラプラス逆変換

表1 を逆に用いる。これを次のように処理する。

(1) 像関数の内挿表現→前置表現への変換、
(2) 前置表現→Ａリストによる表現、
(3) 部分分数分解、
(4) 各関数に対するラプラス逆変換、
(5) 原関数の前置表現→内挿表現、

第(2)項は像関数の部分分数分解を行うためのもので、像関数の分母は因数分解された形で与えられるものとして、分母をＡリストのリストで表す。

第(3)項では、先ず像関数の分子を定数 "1" とした場合の各因数に対する分子を、拡張されたユークリッドの互除法よって求める。像関数の分子が定数でないときは、像関数の分子と求められた分子との積を部分分数の分母で割って余りを部分分数の分子とする。各分子はまとめてＡリストのリストで表すようにした。これらのリストを用いて、像関数の部分分数分解を前置形式で返す。

分子が定数の場合の変換例；
f1=(/ 1 (* S (+ S 2)))とする。

```
>(alist (caddr f1) 's)；分母の表現。
(((1 . 1)) ((1 . 1) (0 . 2)))

>(decomp (alist (caddr f1) 's) 's)；分子の値。
(((0 . 1/2)) ((0 . -1/2)))

>(parfs f1 's)；部分分数分解を行う。
(+ (/ 1/2 S) (/ -1/2 (+ S 2)))
```

分子が定数ではない場合の例；

```
>(parfs '(/ (+ S -1) (* S (+ S 2))) 's)
(+ (/ -1/2 S) (/ 3/2 (+ S 2)))
```

数は既約分数で表わされるので、互除法における中間膨張の心配はなかった。また、分母に重複因子が含まれる場合には、その部分に対して改めて部分分数分解を行うようにした。

ラプラス逆変換の例として、応用数学の教科書から代表的なものを選んで実行した結果を示す。

分母に重複因子を含まない場合

```
>(ilaplc '((11 * s - 12)
          / (s * ((s - 2) * (s + 3)))) 's)
(2 + (EXP (2 * T)) + -3 * (EXP (-3 * T)))
```

分母に１次式が重複因子として含まれる場合

```
>(ilaplc '((s + 2)
          / ((s - 1) ^ 2 * s ^ 3)) 's)
(-8 * (EXP T) + 3 * T * (EXP T)
              + 8 + 5 * T + T ^ 2)
```

分母が２次式の因子を含む場合

```
>(ilaplc '((s + 3)
          / (s ^ 2 - 2 * s + 2)) 's)
(4 * (EXP T) * (SIN T) + (EXP T) * (COS T))
```

４．むすび

記号ラプラス変換の試みについて概要を述べた。数式の認識は前置表現で、計算はこれをＡリストに変換して行い、応用数学の教科書に載っているものを例題として解いた。ここには示していないが、計算機処理なので、相当に大規模な問題も扱うことができる。

この応用を開拓するには、色々な問題があるので、それらを今後の研究課題としたいと、考えている。例えば、式の係数が数だけではなく、記号も含まれる場合には、部分分数分解の計算を適用しないことにしたが、これを部分的に緩めるなどの工夫も重要である。

最後に、山口昭子氏(日本電気)の昨年の卒業研究も本報告に寄与していることを明記して、謝意を表す。

参考文献

[1] 矢野健太郎、石原繁共著、基礎解析学、
裳華房、1981
[2] 後藤英一、一松信、広田良吾編、
数式処理のすすめ、共立出版、1986
[3] J. Moses, Symbolic Integration, MAC-TR-47,
Proj. MAC, MIT Press, 1967
[4] 下地貞夫、数式処理、森北出版、近刊