T H E S O F T W A R E H O U S E

N E W S L E T T E R # 1 0

April 1984

Aloha from Hawaii! The Soft Warehouse Newsletter provides you with information on new Soft Warehouse products, and software extensions or corrections to existing products. In addition, the newsletter is a medium for the exchange of ideas and application programs within the growing community of **muMATH** and **muLISP** users.

If you would like to subscribe, or extend your subscription to the Newsletter for three issues beyond the expiration number on your mailing label, please send $6 ($10 for orders from outside the U.S. or Canada) by check, VISA, or Master Card to The Soft Warehouse, P.O. Box 11174, Honolulu, Hawaii, 96828, U.S.A. A complete set of back issues is available on request for $15.

Recent Reviews and Articles

"Computer Algebra: I", William Squire, ACCESS[tm] The Journal of Microcomputer Applications in Engineering and Science, Volume 3 Number 1, January/February 1984, pp. 14-17. The first of a series of articles about muMATH "to acquaint the reader with the potential of computer algebra" and "to serve as a supplement to the (muMATH) manual in helping to use the system".

"Why Not LISP?", Gary Rader, PC Firing Line, Issue #1, P.O. Box 5503, North Hollywood, CA, 91616 (or phone Bill Salkin at 818 509-9002). PC Firing Line is a "magazine" distributed on a double sided IBM PC diskette. It contains several interesting articles by a number of authors on a wide range of topics. Gary Rader's article discusses the advantages of LISP over more conventional programming languages and presents some examples using muLISP.

"LISP For The PC", PC Tech Journal, Volume 1, Number 8, May 1984. A comparative review of muLISP, IQLISP, and TLC LISP.

"LISPing With Your PC", David T. McClellan, PC Magazine, Volume 2, Number 7, December 1983, pp. 517-528. A comparative review of muLISP and IQLISP.

"LISP For Your Personal Computer", David T. McClellan, Computers & Electronics, March 1984, p. 66. A short comparative review of muLISP and IQLISP.

"Adding Primitive I/O Functions to muLISP", Michael Carter, Dr. Dobbs Journal, forthcoming. A detailed article about how to write and link to machine language routines from the CP/M-80 version of muLISP-80. Implementing port I/O functions is used as an example.

A muMATH and muLISP User Groups

The following people have expressed an interest in joining or establishing informal user groups for the exchange of muMATH and/or muLISP application programs and programming tips:

muMATH:	Jack Mayes 5600 Kirkside Dr. #B Bakersfield, CA 93309	Heinz W. Sternberg 1100 Ptarmigan Dr. #6 Walnut Creek, CA 94595
muLISP:	Larry R. Yates The Problem Solving Center 417 Riverside Dr. New York, NY 10025	Glen Yoshimoto Machine Metamorphics, Inc. 15887 Ravine Rd. Los Gatos, CA 95030

Let the Newsletter editor know if you would like to have your name listed in the next Newsletter or if your group wishes to make an announcement in the Soft Warehouse Newsletter.

* * * * * * * **T h e m u M A T H e m a t i c i a n** * * * * * *

muMATH-83 Quick Reference Card

Jack Mayes, 5600 Kirkside Dr. #B
Bakersfield, California, 93309

To facilitate my own use of muMATH, I have begun making a quick reference sheet of muMATH commands, files, and special syntax structures. Once done, I believe this short summary could be helpful to other muMATH users. I would be willing to give a copy of it to any registered owner who would send me a self-addressed properly stamped envelope. -- Jack Mayes

Correction to STFORM Program

Daniel Bump sent us the following bug fixes to his STFORM program that was published in Newsletter #9. He writes: "The correction to the function STSPLIT2 prevents the occasional occurrence of infinite loops using STFORM. The correction to the function STFORM guarantees correct behavior when the argument is an atom or a monomial".

```
FUNCTION STFORM (EX1),
  WHEN SUM (EX1: EXPAND (EX1)),
    EX1: STUNFILT (STSORT (STFILT (REST (EX1)))),
    WHEN EMPTY (REST (EX1)), FIRST (EX1) EXIT,
    ADJOIN ('+, EX1) EXIT,
  WHEN ATOM (EX1) OR PRODUCT (EX1) OR POWER (EX1),
    STUNSPLT (FIRST (COEFLIST: LIST (STSPLIT1 (EX1, 1, STVAR)))) EXIT
  EX1
ENDFUN$
```

```
FUNCTION STSPLIT2 (EX1, EX2, EX3, EX4),
  WHEN DEN (EX4: EX1/EX3) = 1,
    STSPLIT2 (EX4, EX2*EX3, EX3) EXIT,
  EX2
ENDFUN$
```

Approximate Rational Arithmetic

Daniel Bump, 1421 Rosearden Drive
Forest Grove, Oregon, 97116

Even though muMATH-83 can display rational numbers using decimal and scientific notation, the internal computations are done exactly. Thus, numeric numerators and denominators can become quite large for lengthy sequences of arithmetic operations. When an exact answer is unnecessary, rounding intermediate results may save substantial time and space. Examples include numerical iterative or truncated series approximations and operations on large matrices of approximate numbers.

The following changes to the muMATH-83 version of ARITH.MUS and ALGEBRA.ARI provide optional automatic rounding of all intermediate fractional results that are not mere reciprocals. In file ARITH.MUS delete the function RATSUM and insert:

```
FUNCTION APPROX (EX1, EX2,  % Local: % EX3, EX4, EX5, EX6, EX7),
  WHEN NEGATIVE (EX2),
    APPROX (-EX1, -EX2) EXIT,
  WHEN EMPTY(EX3) AND ONE(EX1) OR MINUS1(EX1),
    ADJOIN (EX1, EX2) EXIT,
  BLOCK
    WHEN EMPTY (EX3),
      EX3: EX6: 1,
      EX4: EX5: 0 EXIT,
  ENDBLOCK,
  WHEN ZERO (EX2),
    ADJOIN (EX3, EX5) EXIT,
  EX1: DIVIDE (EX1, EX2),
  EX7: POP (EX1),
  EX6: EX5 * EX7 + EX6,
  WHEN EX6 > MAXDEN,
    ADJOIN (EX3, EX5) EXIT,
  APPROX (EX2, EX1, EX3 * EX7 + EX4, EX3, EX6, EX5),
ENDFUN$
```

Modify the functions MKRAT and ADDTERMS as follows:

```
FUNCTION MKRAT (EX1, EX2),
  EX2: APPROX (EX1, EX2), EX1: POP (EX2),      % insert this line %
  WHEN ONE (EX2), EX1 EXIT,
  ...
ENDFUN$
```

```
FUNCTION ADDTERMS (EX1, EX2),
  ...
  WHEN NUMBER (EX1)  AND  NUMBER (EX2),        % changed next line %
    MKRAT(NUM(EX1)*DEN(EX2)+NUM(EX2)*DEN(EX1),DEN(EX1)*DEN(EX2)) EXIT,
  WHEN APPLY (GET ('+, FIRST(EX1)), ADJOIN (EX2, ARGLIST(EX1))) EXIT,
  APPLY (GET('+,FIRST(EX2)), ADJOIN(EX1,ARGLIST(EX2)))
ENDFUN$
```

In ARITH.MUS and ALGEBRA.ARI change PROPERTY *, ^ as follows:

```
PROPERTY *, ^, FUNCTION (EX1, EX2, EX3),
  WHEN MINUS1 (EX3),
    WHEN INTEGER (EX2),
      WHEN INTEGER (EX1),
      WHEN EX1 EQ FIRST(EX2:APPROX(EX1,EX2)), FALSE EXIT, %changed%
        EX1: POP (EX2),                                   %changed%
      WHEN ONE (EX2), EX1 EXIT,
        ...
```

With these changes, when the control variable MAXDEN is a positive integer, all nonreciprocal numeric results are replaced by the closest reduced fraction whose denominator does not exceed MAXDEN. (For the underlying theory, see Theory of Numbers by Hardy and Wright.) For example, to limit denominators to 1000:

```
? MAXDEN: 1000$

? pi: 314159265/100000000;  % Assignment accurate to 8 digits %
@: 355/113                  % but denominator of result < 1000 %

? POINT: 7$

? pi;                       % pi is only accurate to 6 digits %
@: 3.1415929
```

The following definition for the EXPonential function should be called only when MAXDEM has a positive integer value:

```
? FUNCTION EXP (X, % Local: % SUM, TERM, N),
    TERM: X,  SUM: N: 1,
    LOOP
      WHEN SUM = (SUM: SUM + TERM), SUM EXIT,
      TERM: TERM * X / (N: N + 1)
    ENDLOOP
  ENDFUN$

? EXP (1);                  % This approximation to e is %
@: 2.7182835                % accurate to 5 digits. %

? NUMNUM: 30$

? EXP (2 pi #I);            % The final result is often exact %
@: 1                        % despite intermediate roundings! %
```

Laplace Transforms

Rob Frohne, 807 Elm Drive
West Lafayette, Indiana, 47906

Laplace transforms have many uses, including transfer functions and the solution of appropriate differential or integral equations. The file LAPLACE.DIF implements LAPLACE transforms, as illustrated by the following demonstration:

```
    ? RHS: K SIN T$

    ? LAPLACE (RHS);    % T is default time-domain independent var %
    @: K/(1 + S^2)      % S is default transform-domain indep. var %

    ? DEPENDS(U(T))$    % Make U implicitly depend on T %

    ? DIFVAR: T$        % Make U` denote DIF (U, T) %

    ? U1: 0$            % Establish initial value of 1st derivative %

    ? LAPLACE (U`` + U == RHS);  % A constant-coefficient linear ODE %
    @: S^2 LAPLACE (U, T, S) + LAPLACE (U, T, S) - UO S == K/(1 + S^2)

    ? SOLVE (@, LAPLACE (U));
    @: {LAPLACE (U, T, S) == (K + U0 S + U0 S^3) / (1 + 2 S^2 + S^4)}
```

```
% File:  LAPLACE.DIF              05/13/83              Rob Frohne %

% LAPLACE.DIF requires features provided by the muMATH-83 version of
  DIF.ALG.  In addition to DIF.ALG, the following packages may prove
  useful: INT.DIF + LIM.DIF, INTMORE.INT, LOG.ALG, TRG.ALG,
  EQN.ALG, SOLVE.EQN.

  USTEP (t) denotes the unit step function  (1 + SIGN (t))/2.

  #GAMMA denotes Euler's constant, 0.5772...

  For any name U depending on T,  LAPLACE (DIF (U, T, integer))  uses
  U0 for the initial value of U, U1 for the initial value of its first
  derivative, U2 for the initial value of its second derivative, etc. %

FUNCTION LAPLACE (EX1, % Optional: % T, S,
    % Local: % LOGEXPD, TRGEXPD),
  BLOCK
    WHEN T EXIT,
    T: 'T, ENDBLOCK,
  BLOCK
    WHEN S EXIT,
    S: 'S, ENDBLOCK,
  LOGEXPD: 30, TRGEXPD: -5,
  EX1: EVAL (EX1),
  WHEN EX1 = T, S^-2 EXIT,
```

```
  WHEN FREE (EX1, T), EX1/S EXIT,
  WHEN APPLY (GET ('LAPLACE, FIRST (EX1)), ARGEX (EX1)) EXIT,
  WHEN FREE (LOGEXPD: DEFINT (EX1 * #E^(-S*T), T, 0, PINF), 'DEFINT),
    LOGEXPD EXIT,
  LIST ('LAPLACE, EX1, T, S),
ENDFUN$

FUNCTION GAMMA (N),
  WHEN NUMBER (N),
    WHEN N > 0,
      WHEN INTEGER (N), (N-1)! EXIT,
      WHEN N = 1/2, #PI^N EXIT,
      WHEN N < 1, LIST ('GAMMA, N) EXIT,
      (N-1) * GAMMA (N-1) EXIT EXIT,
  LIST ('GAMMA, N),
ENDFUN$

PROPERTY LAPLACE, USTEP, FUNCTION (EX1),
  WHEN FREE (EX1: EX1 - T, T), #E^(EX1*S)/S EXIT,
ENDFUN$

PROPERTY LAPLACE, DIF, FUNCTION (EX2,
    % Local: % EX3, EX4, EX5),
    % Refers to outside vars EX1, T, S from LAPLACE %
  WHEN THIRD (EX1) = T,
    BLOCK
      WHEN EX4: THIRD (REST (EX1)) EXIT,
      EX4: 1 ENDBLOCK,
    EX3: S^EX4 * LIST ('LAPLACE, EX2, T, S),
    BLOCK
      WHEN NAME (EX2), EX2: EXPLODE (EX2) EXIT,
      EX2: EXPLODE (FIRST (EX2)) ENDBLOCK,
    EX5: -1,
    LOOP
      EX3: EX3 - FIRST (COMPRESS (APPEND (EX2, EXPLODE (EX4: EX4-1))))
        * S^(EX5:EX5+1),
      WHEN ZERO (EX4), EX3 EXIT,
    ENDLOOP EXIT,
ENDFUN$

PROPERTY LAPLACE, +, FUNCTION (EX1, EX2),
  LAPLACE (EX1, T, S) + LAPLACE (EX2, T, S),
ENDFUN$

PROPERTY LAPLACE, *, FUNCTION (EX1, EX2),
  WHEN FREE (EX1, T), EX1 * LAPLACE (EX2, T, S) EXIT,
  WHEN FREE (EX2, T), EX2 * LAPLACE (EX1, T, S) EXIT,
  WHEN EX1 = T,
    WHEN FREE (EX2: LAPLACE (EX2, T, S), 'LAPLACE),
      DIF (EX2, S) EXIT EXIT,
  WHEN POWER (EX1),
    WHEN BASE (EX1) = #E,
      WHEN FREE (EXPON (EX1)/T, T),
        EVSUB (LAPLACE (EX2, T, S), S, S - EXPON (EX1)/T) EXIT EXIT,
    WHEN BASE (EX1) = T,
```

```
      WHEN POSITIVE (EX1: EXPON (EX1)),
        (-1)^EX1 * DIF (LAPLACE (EX2, T, S), S, EX1) EXIT EXIT EXIT,
  WHEN POWER (EX2), "LAPLACE*" (EX2, EX1) EXIT,
  WHEN EX1 = USTEP (T), LAPLACE (EX2, T, S) EXIT,
  WHEN FIRST (EX1) EQ 'USTEP,
    WHEN FREE (EX1: SECOND (EX1) - T, T)
      #E^(EX1 * S) * LAPLACE (EVSUB (EX2, T, T - EX1), T, S) EXIT EXIT,
  WHEN FIRST (EX2) EQ 'USTEP, "LAPLACE*" (EX1, EX2) EXIT,
ENDFUN$

PROPERTY LAPLACE, SIN, FUNCTION (EX1),
  WHEN FREE (EX1: EX1/T, T), EX1 / (S^2 + EX1^2) EXIT,
ENDFUN$

PROPERTY LAPLACE, COS, FUNCTION (EX1),
  WHEN FREE (EX1: EX1/T, T), S / (S^2 + EX1^2) EXIT,
ENDFUN$

PROPERTY LAPLACE, INT, FUNCTION (EX1, EX2),
  WHEN EX2 = T, LAPLACE (EX1, T, S) / S EXIT,
ENDFUN$

PROPERTY LAPLACE, DEFINT, FUNCTION(EX1, EX2, EX3, EX4),
  WHEN ZERO (EX3) AND EX4 = T,
    LAPLACE (EVSUB (EX1, EX2, T)/S, T, S) EXIT,
ENDFUN$

PROPERTY LAPLACE, ^, FUNCTION (EX1, EX2),
  WHEN EX1 EQ #E,
    WHEN FREE (EX2: EX2/T), (S - EX2)^-1 EXIT EXIT,
  WHEN EX1 = T,
    WHEN NUMBER (EX2) AND EX2 > -1 OR FREE (EX2, T),
      GAMMA (EX2+1) * S^(-1-EX2) EXIT EXIT,
ENDFUN$

PROPERTY LAPLACE, LOG, FUNCTION (EX1, EX2),
  WHEN EX1 = T, (-#GAMMA - LN (S))/S EXIT,
ENDFUN$

PROPERTY LAPLACE, ERF, FUNCTION (EX1, % Local: % EX2),
  WHEN FREE (EX2: EX1*T^(1/2), T),
    (1 - #E^(-2*EX2*S^(1/2))) / S EXIT,
  WHEN FREE (EX2: EX1/T, T),
    #E^(S^2/(4*EX2^2)) * (1 - ERF (S/(2*EX2))) / S EXIT,
ENDFUN$

PROPERTY LAPLACE, "==", FUNCTION (EX1, EX2),
  LIST ('"==", LAPLACE (EX1, T, S), LAPLACE (EX2, T, S)),
ENDFUN$

RDS ()$
```

Fix for muMATH-83 Cubic Equation Solver

A bug in the 10/01/83 version of SOLVE.EQN prevents the solution of some cubic equations such as:

? **SOLVE (X^3 + X^2 + 1, X);**

It can be fixed by changing the line WHEN EX5^2=EX2^**2** in the function SOLEXP that occurs two lines before the comment line

% End optional lines for cubic and quartic equations %

to WHEN EX5^2=EX2^**3**. Change the date of the file to 02/25/84.

An improvement for matrix division

Without full expansion over a common denominator, the function "\" in the file MATRIX.ARR might not recognize a singular nonnumeric matrix. To force expansion over a common denominator in versions of MATRIX.ARR dated prior to 04/11/84, modify the first few lines of the function "\" to read

```
FUNCTION \ (EX1, EX2,
    % Local: % NUMNUM, DENDEN, DENNUM, PWREXPD,        % New %
                EX3, EX4, LEX1, LEX2, LEX3, LEX4),
  NUMNUM: DENDEN: 30, DENNUM: -30, PWREXPD: 6,         % New %
  WHEN (COLMAT(EX1) OR ROW(EX1)
      AND COLMAT(EX1: EX1.IDMAT(LENGTH(REST(EX1)))))
  ...
```

and change the date of the file to 04/11/84.

Fix for the Absolute Value Trig Identities

If the file TRG.ALG is dated prior to 01/09/84, replace the lines

```
     PROPERTY COS, ABS, IDENTITY $
     PROPERTY SEC, ABS, IDENTITY $
with
     PUT ('COS, 'ABS, 'COS) $
     PUT ('SEC, 'ABS, 'SEC) $
```

If the file HYPER.ALG is dated prior to 01/09/84, replace the lines

```
     PROPERTY SECH, ABS, IDENTITY $
     PROPERTY COSH, ABS, IDENTITY $
with
     PUT ('SECH, 'ABS, 'SECH) $
     PUT ('COSH, 'ABS, 'COSH) $
```

Change the date of both files to 01/09/84.

Upcoming LISP & AI Conferences

The University of Texas at Austin will host back-to-back conferences on LISP and AI in August. The LISP and Functional Programming Conference will be held from August 5 through 8 (for details contact Edward A. Schneider, Burroughs Corp., Austin Research Center, 12201 Technology Blvd., Austin, TX, 78727). The AAAI-84 Conference will be held from August 6 through 10 (for details contact the American Association for Artificial Intelligence, 445 Burgess Dr., Menlo Park, CA, 94025).

A Merge Sort Utility for Files

The Soft Warehouse needed a simple utility program to automate the merging of monthly name and address files into one file for the entire year. The resulting merge sort utility, perhaps with appropriate modifications for a particular application, may prove useful to our readers. Since the utility needs to have multiple open input files, muLISP-83 is required to run the program.

The input files can be any ASCII text files as long as the records are separated by a blank line and they are sorted based on the first line of each record. For example:

```
Smith, Dr. John
12345 Recursion Drive
Function City, MU  54321
```

```
; File: MERGE.LIB  (C)       04/11/84      The Soft Warehouse

; MERGE [file-list, target-file, file-type, src-drv, tar-drv]
; <file-list> is a list of files to be merged.
; <target-file> is the desired name of the target file.
; <file-type> is the file name extension of the files in <file-list>.
; <src-drv> (an optional argument) designates the source files' drive.
; <tar-drv> (an optional argument) designates the target file's drive.

(PUTD 'DEFUN '(NLAMBDA (NAM$ EXP$) (PUTD NAM$ EXP$) NAM$))

(DEFUN MERGE (LAMBDA (FILE-LIST TARGET-FILE FILE-TYPE SRC-DRV TAR-DRV
    % Local: % LST SORT-LIST FILE-NAME LINELENGTH RECORD)
  (LOOP                          ;Read first record from each file
    ((NULL FILE-LIST))
    (SETQ FILE-NAME (POP FILE-LIST))
    ( ((NULL (RDS FILE-NAME FILE-TYPE SRC-DRV)))
      (SETQ RECORD (READ-RECORD))
      ((NULL RECORD)  (RDS) )
      (PUSH (CONS FILE-NAME RECORD) SORT-LIST) ) )
  (WRS TARGET-FILE FILE-TYPE TAR-DRV)  (SETQ WRS)
  (LOOP                          ;Merge records loop
```

```
    ((NULL SORT-LIST))
    (SETQ LST SORT-LIST)
    (SETQ RECORD (POP LST))
    (LOOP                          ;Find next record to write
      ((NULL LST))
      ( ((COMPARE (UNPACK (CADR RECORD)) (UNPACK (CADAR LST))))
        (SETQ RECORD (CAR LST)) )
      (POP LST) )
    (WRITE-RECORD (CDR RECORD))    ;Write record to target file
    (SETQ FILE-NAME (CAR RECORD))
    (RDS FILE-NAME FILE-TYPE SRC-DRV)
    (SETQ SORT-LIST (REMBER RECORD SORT-LIST))
    (SETQ RECORD (READ-RECORD))    ;Read in next record from file
    ( ((NULL RECORD) (RDS) )
      (PUSH (CONS FILE-NAME RECORD) SORT-LIST) ) )
  (WRS) ))                         ;Close target file

(DEFUN READ-RECORD (LAMBDA (% Local: % LINE LST)
  (LOOP
    (SETQ LINE (READ-LINE))
    ((OR (NULL LINE) (EQ LINE '""))  (REVERSE LST) )
    (PUSH LINE LST) ) ))

(DEFUN READ-LINE (LAMBDA (% Local: % LST)
  (LOOP
    ((NOT (READP)) NIL)
    ((EQ (READCH) CR)  (PACK (REVERSE LST)) )
    ( ((EQ RATOM LF))
      (PUSH RATOM LST) ) ) ))

(DEFUN WRITE-RECORD (LAMBDA (LST % Local: % WRS)
  (SETQ WRS T)
  (LOOP
    ((NULL LST)  (TERPRI) )
    (PRINT (POP LST)) ) ))

(DEFUN COMPARE (LAMBDA (ATMLST1 ATMLST2)
  (LOOP
    ((NULL ATMLST1) ATMLST2)
    ((NULL ATMLST2) NIL)
    ((NOT (EQ (CAR ATMLST1) (CAR ATMLST2)))
      (LESSP (ASCIIX (CAR ATMLST1)) (ASCIIX (CAR ATMLST2))) )
    (POP ATMLST1)  (POP ATMLST2) ) ))

(DEFUN ASCIIX (LAMBDA (ATM)
  ((NUMBERP ATM) ATM)
  ((LESSP 96 (SETQ ATM (ASCII ATM)) 123)  (DIFFERENCE ATM 32) )
  ATM ))

(DEFUN REMBER (LAMBDA (OBJ LST)
  ((NULL LST) NIL)
  ((EQUAL OBJ (CAR LST))  (CDR LST) )
  (CONS (CAR LST) (REMBER OBJ (CDR LST))) ))

(SETQ CR (ASCII 13))  (SETQ LF (ASCII 10))  (RDS)
```