Chapter 8: Using The System

8.1 Introduction

This chapter demonstrates the system working. Transcripts of the CLI interaction are included, along with anotations explaining what is being demonstrated.

8.2 Using Graph Manager

This test demonstrates graph spec list manipulation.

```
Commands Available:
     newspec <spec_name>
     delspec <spec_name>
     specs
     resetmanager
     return
     help


Graph Manager> newspec xgraph                                    // Add new graph spec
Horizontal:
Scale Division? 1
Inter-Divisions? 1
Range Min? 0
Range Max? 10
Vertical:
Scale Division? 1
Inter-Divisions? 1
Range Min? 0
Range Max? 10

Graph Manager> specs                                             // Output all graph specs
xgraph-> Graph Spec:
     Horiz: range (min=0, max=10), scale(scale=1, divs=1)
     Vert: range (min=0, max=10), scale(scale=1, divs=1)


Graph Manager> newspec anothergraph                              // Add new graph spec
Horizontal:
Scale Division? 10
Inter-Divisions? 5
Range Min? 0
Range Max? 100
Vertical:
Scale Division? 30
Inter-Divisions? 10
Range Min? 0
Range Max? 100

Graph Manager> specs                                             // Output all graph specs
anothergraph-> Graph Spec:
     Horiz: range (min=0, max=100), scale(scale=10, divs=5)
     Vert: range (min=0, max=100), scale(scale=30, divs=10)
 xgraph-> Graph Spec:
     Horiz: range (min=0, max=10), scale(scale=1, divs=1)
     Vert: range (min=0, max=10), scale(scale=1, divs=1)

Graph Manager> newspec xgraph                    // Attempt add with non-unique name
```

```
Horizontal:
Scale Division? 30
Inter-Divisions? 20
Range Min? 10
Range Max? 30
Vertical:
Scale Division? 40
Inter-Divisions? 50
Range Min? 60
Range Max? 70
ERROR: graph spec name must be unique to this list

Graph Manager> delspec ygraph                    // Attempt delete of non-existent spec
ERROR: no such spec exists

Graph Manager> delspec xgraph                                    // Delete existing spec

Graph Manager> specs                                             // Output all specs
anothergraph-> Graph Spec:
     Horiz: range (min=0, max=100), scale(scale=10, divs=5)
     Vert: range (min=0, max=100), scale(scale=30, divs=10)


Graph Manager> delspec xgraph                    // Attempt delete of non-existent spec
ERROR: no such spec exists

Graph Manager> delspec anothergraph              // Delete existing spec

Graph Manager> specs                             // Output all specs
This u-list is empty

Graph Manager>
```

8.3 Using Data Manager

```
System Manager> dataman
Commands Available:
     newrecord <record_name> <field_name>...
     delrecord <record_name>
     records
     newdataset <data_name> <record_name> <length>
     deldataset <data_name>
     datas
     newinput <input_name> lower_index upper_index start_value step
     delinput <input_name>
     inputs
     newmap <map_name> <input_field_name>... (newline) <output_field_name>...
     delmap <map_name>
     maps
     resetmanager
     help

Data Manager> newrecord record1 column1 column2 column3          // Add new record
name is:record1, records are  column1 column2 column3

Data Manager> newrecord record0 x y z                            // Add new record
name is:record0, records are  x y z

Data Manager> records                                            // Output all records
record0-> 1 x 2 y 3 z
 record1-> 1 column1 2 column2 3 column3

Data Manager> delrecord record3                          // Delete non-existent record
ERROR: record name not found

Data Manager> delrecord record0                          // Delete existing record
Data Manager> delrecord record1                          // Delete existing record
Data Manager> records                                    // Output all records
This u-list is empty

Data Manager> newrecord record1 x y z                    // Add new record
name is:record1, records are  x y z

Data Manager> newdataset data1 record1 10                // Add new data set
Data Manager> datas                                      // Output all data set
data1-> Record Base: record1
Dimensions: 10x3
Field Names: (3)
        x   y   z
i=0 ) 0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j
i= 1) 0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j
i= 2) 0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j
i= 3) 0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j
i= 4) 0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j
i= 5) 0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j
i= 6) 0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j
i= 7) 0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j
i= 8) 0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j
i= 9) 0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j   0.000000e+00+0.000000e+00j


Data Manager> deldataset data1                           // Delete data set
Data Manager> datas                                      // Output all data sets
This u-list is empty
```

```
Data Manager> newinput input1 0 10 100 10                // Add new set input

Data Manager> inputs                                     // Output all set inputs
input1-> lower index=0
upper index=10
start value=1.000000e+02
incrementer=1.000000e+01

Data Manager> newinput input0 10 20 30 40                // Add new set input
Data Manager> inputs                                     // Output all set inputs
input0-> lower index=10
upper index=20
start value=3.000000e+01
incrementer=4.000000e+01

 input1-> lower index=0
upper index=10
start value=1.000000e+02
incrementer=1.000000e+01

Data Manager> delinput input1                            // Delete set input
Data Manager> inputs                                     // Output all set inputs
input0-> lower index=10
upper index=20
start value=3.000000e+01
incrementer=4.000000e+01

Data Manager> delinput input0                            // Delete set input
Data Manager> inputs                                     // Output all set inputs
This u-list is empty

Data Manager> newmap map1 x y                            // Add new map
now enter output fields
z

Data Manager> maps                                       // Output all maps
map1->        Input Fields: x, y,
         Output Fields: z,

Data Manager> newmap map1 o p                            // Add new map - non-unique name
now enter output fields
l
ERROR: name is not unique to this list

Data Manager> delmap map1                                // Delete map
Data Manager> maps                                       // Output map list
This u-list is empty

Data Manager>
```

8.4 Using Calculator Manager

```
Commands Available:
     add <name>
     remove <name>
     set_current <name>
     storage
     errors
     clear_errors
     clear_memory
     clear <name>
     resetmanager
     verify
     auto_verify_on
     auto_verify_off
     dump
     return
     order <expression> | <assignment> | <equation>
     help


Calculator Manager.NULL> add calc1                              // Add calculator to list
Calculator Manager.calc1> add calc2                             // Add calculator to list
Calculator Manager.calc2> dump                                  // Output all calculators
Calculator Manager List:
Calculator 'calc1' contains:
Variable List:
 This u-list is empty
EQUATION List:
 This u-list is empty

Calculator 'calc2' contains:
Variable List:
 This u-list is empty
EQUATION List:
 This u-list is empty

Calculator Manager.calc2> set_current calc1                 // Set current to calc1
Calculator Manager.calc1> remove calc1                      // Remove existent calculator
Calculator Manager.calc2> remove calc2                      // Remove existent calculator
Calculator Manager.NULL> remove iop                         // Remove non-existent calc.
Manager ERROR - iop calculator was not removed because 'iop' calculator doesn't exist

Calculator Manager.NULL> add calc1                              // Add calculator to list

Calculator Manager.calc1> order 10+10                           // Simple addition
result is: '2.000000e+01+0.000000e+00j'

Calculator Manager.calc1> order 10+20*30                        // Precedence check
result is: '6.100000e+02+0.000000e+00j'

Calculator Manager.calc1> order 2^2                             // Real power
result is: '4.000000e+00+0.000000e+00j'

Calculator Manager.calc1> order sqrt(-2)                        // Complex square root
result is: '-0.000000e+00+1.414214e+00j'

Calculator Manager.calc1> order e^(j)                           // Polar to rectangular
result is: '5.403023e-01+8.414710e-01j'

Calculator Manager.calc1> order e^(pi*j)                        // Polar to rectangular
```

```
result is: '-1.000000e+00+-0.000000e+00j'

Calculator Manager.calc1> order (10+20)*30                    // Parentheses check
result is: '9.000000e+02+0.000000e+00j'

Calculator Manager.calc1> order 10!                           // Factorial check
result is: '3.628800e+06+0.000000e+00j'

Calculator Manager.calc1> order 10.5!                         // invalid operand check
Number_of_errors was 1
Errors are present in 'calc1'
result is: '0.000000e+00+0.000000e+00j'

Calculator Manager.calc1> errors                              // View error report
error: invalid FACTORIAL operand: non-integer / non-real

Calculator Manager.calc1> clear_errors                        // Clear error report

Calculator Manager.calc1> order SUM(n,0,10) n                 // Summation n, 0->10
result is: '5.500000e+01+0.000000e+00j'

Calculator Manager.calc1> order 1+2+3+4+5+6+7+8+9+10          // Verify Summation
result is: '5.500000e+01+0.000000e+00j'

Calculator Manager.calc1> order SUM(n,10) n                   // Summation n, 0->10
result is: '5.500000e+01+0.000000e+00j'

Calculator Manager.calc1> order SUM(n,10) n+2                 // Sum operates on term
result is: '5.700000e+01+0.000000e+00j'

Calculator Manager.calc1> order SUM(n,10) (n+2)          // Sum operates on expression
result is: '7.700000e+01+0.000000e+00j'

Calculator Manager.calc1> order sin 10                   // Sine, no parentheses
result is: '-5.440211e-01+0.000000e+00j'

Calculator Manager.calc1> order sin (10*20+30)           // Sine of expression
result is: '-6.160642e-01+0.000000e+00j'

Calculator Manager.calc1> order sin (pi-pi)              // Use constants
result is: '0.000000e+00+0.000000e+00j'

Calculator Manager.calc1> order tan (pi/2)               // illegal operand
Number_of_errors was 1
Errors are present in 'calc1'
result is: '0.000000e+00+0.000000e+00j'

Calculator Manager.calc1> errors                         // view error log
error: invalid
SQRT/CBRT/SIN/COS/TAN/SINH/COSH/TANH/ASIN/ACOS/ATAN/ASINH/ACOSH/ATANH/LN/LOG10/LOG2
operand

Calculator Manager.calc1> clear_errors                   // clear error log

Calculator Manager.calc1> errors                         // view error log
0 errors
Calculator Manager.calc1> order 1/0                      // Illegal divide by 0
Number_of_errors was 1
Errors are present in 'calc1'
result is: '0.000000e+00+0.000000e+00j'

Calculator Manager.calc1> errors                         // view error log
error: divide by 0
```

```
Calculator Manager.calc1> order 10MEGA                  // Engineering Multiplier
Number_of_errors was 1
Errors are present in 'calc1'
result is: '1.000000e+07+0.000000e+00j'

Calculator Manager.calc1> errors                        // Error report remains
error: divide by 0

Calculator Manager.calc1> order 10FEMTO                 // Engineering Multiplier
Number_of_errors was 1
Errors are present in 'calc1'
result is: '1.000000e-14+0.000000e+00j'

Calculator Manager.calc1> clear_errors

Calculator Manager.calc1> order (1+j)/(1-j)             // Complex division
result is: '0.000000e+00+1.000000e+00j'

Calculator Manager.calc1> order (1+j)(1+j)              // Must not leave out *
Number_of_errors was 1
Errors are present in 'calc1'
result is: '1.000000e+00+1.000000e+00j'

Calculator Manager.calc1> order (1+j)*(1+j)             // Complex Multiplication
Number_of_errors was 1
Errors are present in 'calc1'
result is: '0.000000e+00+2.000000e+00j'

Calculator Manager.calc1> errors                        // View error report
error: Illegal Terminating Token

Calculator Manager.calc1> clear_errors

Calculator Manager.calc1> order j                       // Complex power
result is: '0.000000e+00+1.000000e+00j'

Calculator Manager.calc1> order j^2                     // Complex power
result is: '-1.000000e+00+-0.000000e+00j'

Calculator Manager.calc1> order j^3                     // Complex power
result is: '0.000000e+00+-1.000000e+00j'

Calculator Manager.calc1> order j^4                     // Complex power
result is: '1.000000e+00+0.000000e+00j'

Calculator Manager.calc1> order (1+j)^0.5               // Complex power
result is: '1.098684e+00+4.550899e-01j'

Calculator Manager.calc1> order 0^0                     // Illegal operands
Number_of_errors was 1
Errors are present in 'calc1'
result is: '0.000000e+00+0.000000e+00j'

Calculator Manager.calc1> errors                        // View error report
error: invalid POWER operands

Calculator Manager.calc1> clear_errors

Calculator Manager.calc1> order e^(1+2*j)               // Complex power
result is: '-1.131204e+00+2.471727e+00j'

Calculator Manager.calc1> order e * e^(2*j)             // Complex power
```

```
result is: '-1.131204e+00+2.471727e+00j'

Calculator Manager.calc1> order 2^(10-8)                    // Power expression
result is: '4.000000e+00+0.000000e+00j'

Calculator Manager.calc1> order 2^0                         // Raise to power 0
result is: '1.000000e+00+0.000000e+00j'

Calculator Manager.calc1> order 2^0.33                      // Fractional power
result is: '1.257013e+00+0.000000e+00j'

Calculator Manager.calc1> order [-10]                       // Modulus
result is: '1.000000e+01+0.000000e+00j'

Calculator Manager.calc1> order [-10+j]                     // Complex modulus
result is: '1.004988e+01+0.000000e+00j'

Calculator Manager.calc1> order arg(1+j)                    // Complex argument
result is: '7.853982e-01+0.000000e+00j'

Calculator Manager.calc1> order re(1+3*j)                   // Extract real component
result is: '1.000000e+00+0.000000e+00j'

Calculator Manager.calc1> order im(1+3*j)                   // Extract imag component
result is: '3.000000e+00+0.000000e+00j'

Calculator Manager.calc1> order 2 logx 128                  // Log base 2 of 128
result is: '7.000000e+00+0.000000e+00j'

Calculator Manager.calc1> order log2 128                    // Log base 2 of 128
result is: '7.000000e+00+0.000000e+00j'

Calculator Manager.calc1> order 2 rootx 100                 // Square root of 100
result is: '1.000000e+01+0.000000e+00j'

Calculator Manager.calc1> order a=10                        // Assignment to variable
result is: '1.000000e+01+0.000000e+00j'

Calculator Manager.calc1> storage                           // Output current calculator
Calculator 'calc1' contains:
Variable List:
 a-> 1.000000e+01+0.000000e+00j

EQUATION List:
 This u-list is empty

Calculator Manager.calc1> order a:20                        // Define equation,invalid name
Number_of_errors was 1
Errors are present in 'calc1'
result is: '2.000000e+01+0.000000e+00j'

Calculator Manager.calc1> errors                            // view errors
error: Illegal tagging of variable name to equation


Calculator Manager.calc1> clear_errors

Calculator Manager.calc1> order a:a                         // Direct Circular Definition
Number_of_errors was 1
Errors are present in 'calc1'
result is: '1.000000e+01+0.000000e+00j'

Calculator Manager.calc1> errors
```

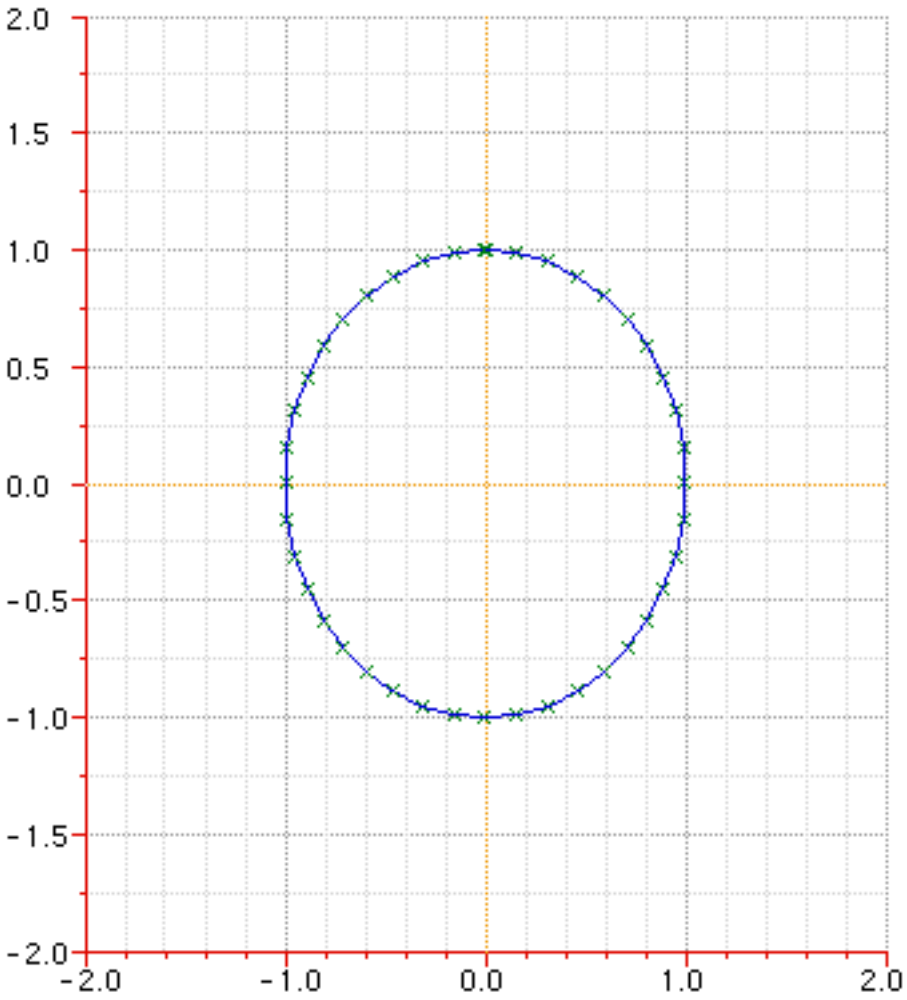
```
error: Illegal tagging of variable name to equation


Calculator Manager.calc1> clear_memory                    // Erase equation a

Calculator Manager.calc1> clear_errors

Calculator Manager.calc1> order a:b                       // Set up equation a
result is: '0.000000e+00+0.000000e+00j'

Calculator Manager.calc1> order b:a               // Set up indirect circular definition
Number_of_errors was 1
Errors are present in 'calc1'
result is: '0.000000e+00+0.000000e+00j'

Calculator Manager.calc1> errors                                  // View error report
error: Circular equation not stored - error trace 'b a b '

Calculator Manager.calc1> storage                         // View storage of current calc
Calculator 'calc1' contains:
Variable List:
 This u-list is empty
EQUATION List:
 a-> EQUATION=b

Calculator Manager.calc1> clear_errors

Calculator Manager.calc1> order a:b
result is: '0.000000e+00+0.000000e+00j'

Calculator Manager.calc1> order b:c
result is: '0.000000e+00+0.000000e+00j'

Calculator Manager.calc1> order c:d
result is: '0.000000e+00+0.000000e+00j'

Calculator Manager.calc1> order d:b                       // Indirect Circular definition
Number_of_errors was 1
Errors are present in 'calc1'
result is: '0.000000e+00+0.000000e+00j'

Calculator Manager.calc1> errors
error: Circular equation not stored - error trace 'b c d b '

Calculator Manager.calc1> clear_errors

Calculator Manager.calc1> order a:b+c
result is: '0.000000e+00+0.000000e+00j'

Calculator Manager.calc1> order d:a+k                     // Indirect Circular definition
Number_of_errors was 1
Errors are present in 'calc1'
result is: '0.000000e+00+0.000000e+00j'

Calculator Manager.calc1> errors
error: Circular equation not stored - error trace 'b c d a b '

Calculator Manager.calc1> clear_errors
Calculator Manager.calc1> clear_memory                    // Clear all vars and eqns

Calculator Manager.calc1> order a:10+m            // Reference variable in equation
result is: '0.000000e+00+0.000000e+00j'
```

```
Calculator Manager.calc1> order a                // Evaluate a
Number_of_errors was 1
Number_of_errors was 1
Errors are present in 'calc1'
result is: '1.000000e+01+0.000000e+00j'

Calculator Manager.calc1> errors                 // View error report
error: 'm'  Name not found

Calculator Manager.calc1> clear_errors

Calculator Manager.calc1> order m=5              // Assign m
result is: '5.000000e+00+0.000000e+00j'

Calculator Manager.calc1> order a                // Evaluate a
result is: '1.500000e+01+0.000000e+00j'

Calculator Manager.calc1> order m=10             // Reassign m
result is: '1.000000e+01+0.000000e+00j'

Calculator Manager.calc1> order a                // Re-evaluate a
result is: '2.000000e+01+0.000000e+00j'

Calculator Manager.calc1> order b:a+30           // Equation b references equation a
result is: '0.000000e+00+0.000000e+00j'

Calculator Manager.calc1> order b                // Evaluate equation b
result is: '5.000000e+01+0.000000e+00j'

Calculator Manager.calc1> order d:a+b            // More equation nesting
result is: '0.000000e+00+0.000000e+00j'

Calculator Manager.calc1> order d                // Evaluate d
result is: '7.000000e+01+0.000000e+00j'
```

8.5 Using System Manager 1

The following script generates the graph below:

```
add calc1
order xgraph:sin(var)
order ygraph:cos(var)

add graph
order xmin=0
order xmin=-2
order xmax=2
order xscale=1
order xdiv=4
order ymin=-2
order ymax=2
order yscale=0.5
order ydiv=1
order varmin=0
order varmax=2*pi
order res=pi/20
set_current calc1
return
```

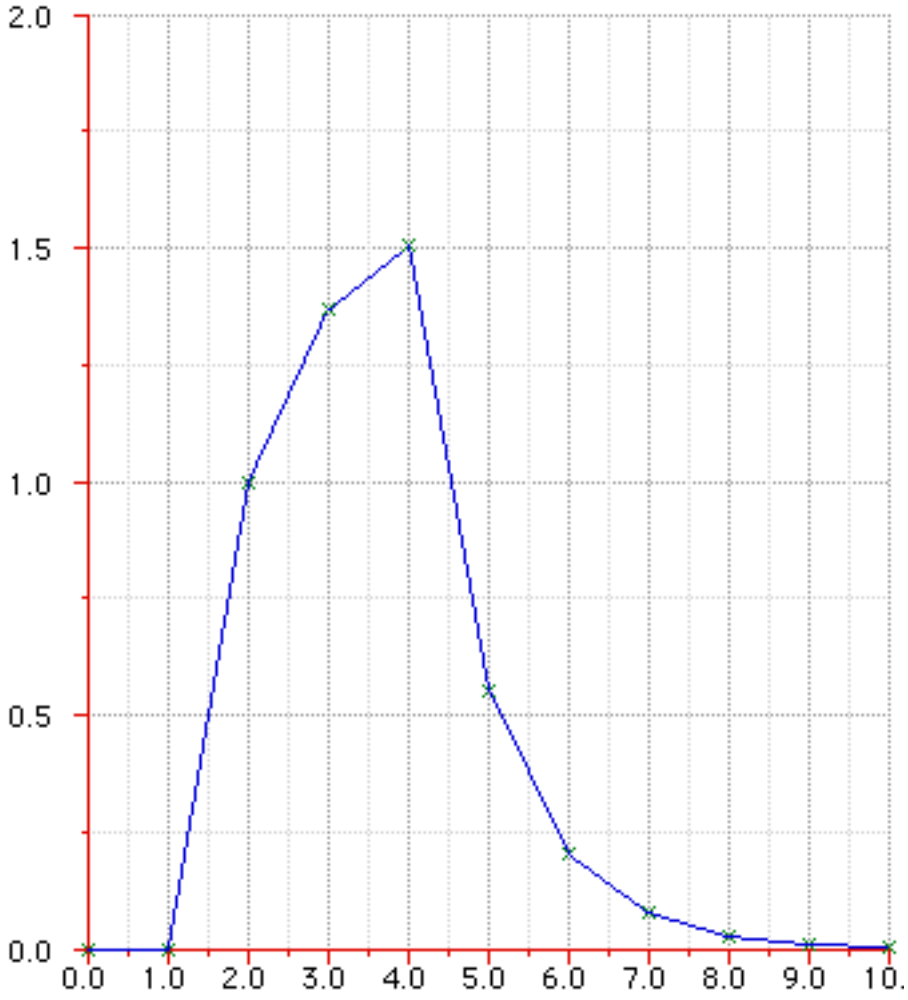


◇ **Figure 16 : System Manager 1 test 1**

The following script performs a convolution between a first order filter and a train of impulses at time=2,3,4. (Time along x-axis.)

```
add convolution
order N=10
order conv:SUM(m,N)(x*h)
order h:wind(t,0,20)*e^(-t)
order t:n-m
order x:wind(m,2,4)
order n:var
order xgraph:var
order ygraph:conv
add graph
set_current graph
order res=1
order xdiv=1
order xmax=10
order xmin=0
order xscale=1
order ymin=0
order ymax=2
order yscale=0.5
order ydiv=1
order varmin=0
order varmax=10
set_current convolution
return
```

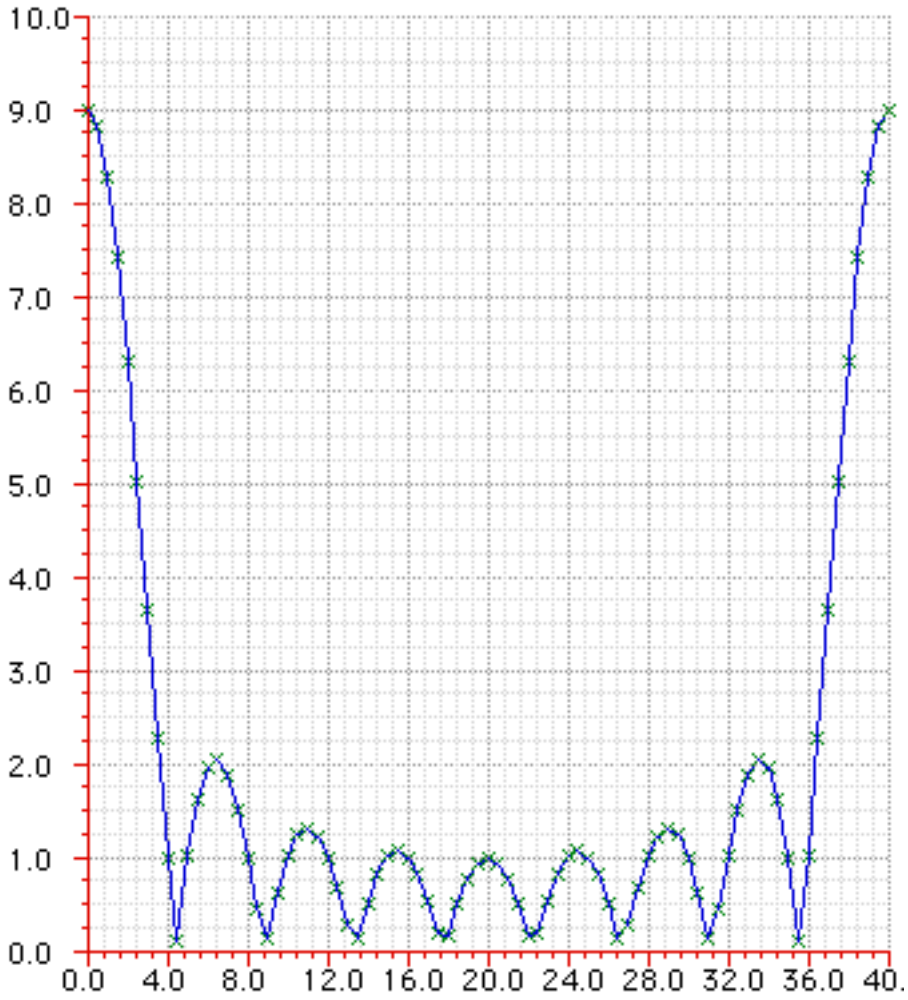
Figure 17 : System Manager 1 test 2



The following script performs the discrete time fourier transform of a rectanfular pulse from time =-4 to time =+4. Magnitude of response is plotted on the y-axis, and array element (which maps to the range 0->2π when considered as frequency) in dtft is plotted on x-axis. This set of equations is derived from Reference 2.

```
add fourier
order dft:SUM(n,-(N-1),N-1)(x*e^(-j*(2*pi/N)*n*k))
order k:var
order x:wind(n,-4,4)
order N=40
order xgraph:var
order ygraph:[dft]
add graph
set_current graph
order xmin=0
order xmax=40
order xscale=4
order xdiv=4
order ymin=0
order ymax=10
order yscale=1
order ydiv=3
order varmin=0
order varmax=40
order res=0.5
set_current fourier
return
```

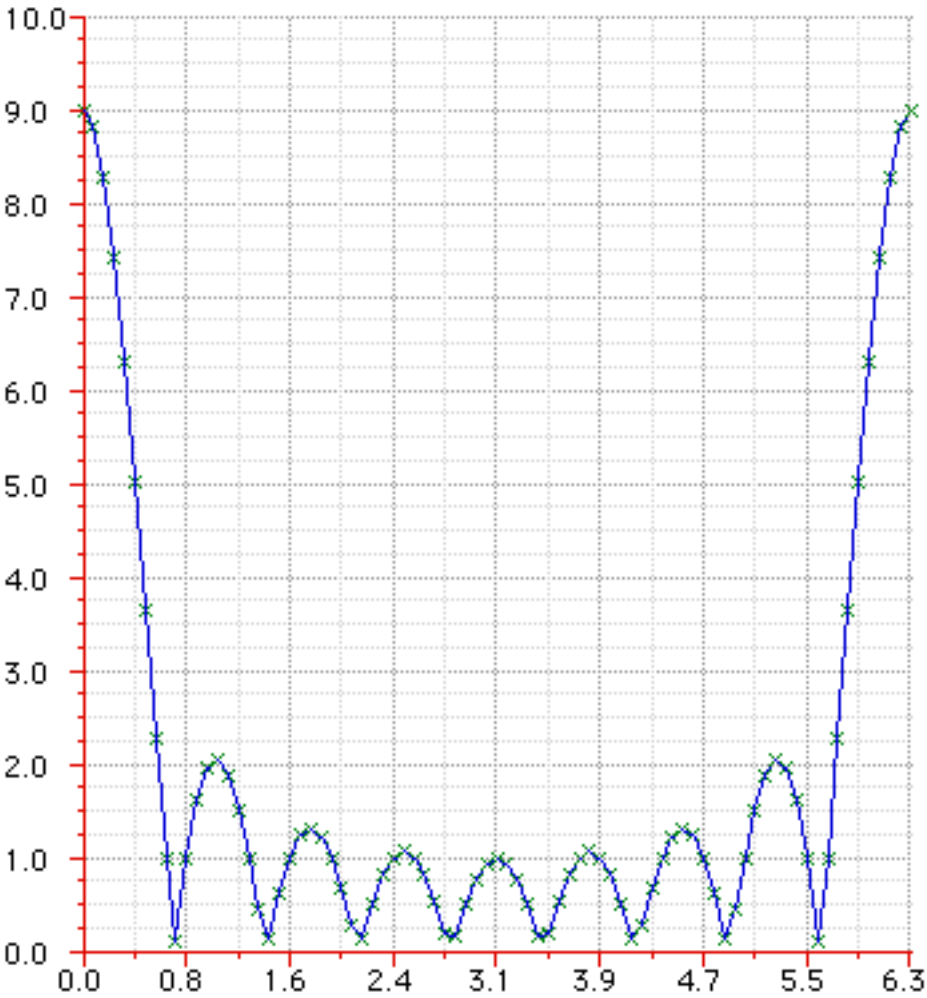
Figure 18 : System Manager 1 test 3



8.6 Using System Manager 2

The following script performs the discrete time fourier transform of a pulse from time =-4 to time =+4. Quicker execution of process than System Manager 1. Y-axis is response magnitude, X-axis is frequency. This set of equations is derived from Reference 2.

```
calcman
add fourier
order dft:SUM(n,-(N-1),N-1)(x*e^(-j*(2*pi/N)*n*k))
order x:wind(n,-4,4)
order N=40
order xgraph:k*pi/20
order ygraph:[dft]
return
dataman
newrecord record1 k xgraph ygraph
newdataset fourier1 record1 81
newinput input1 0 80 0 0.5
newmap map1 k
xgraph ygraph
return
graphman
newspec fourier
0.785398163
1
0
6.283185307
1
3
0
10
return
setprocess c fourier
setprocess d fourier1
setprocess g fourier
setprocess s input1
k
setprocess m map1
doprocess
graphprocess rect xgraph ygraph 0 80
```

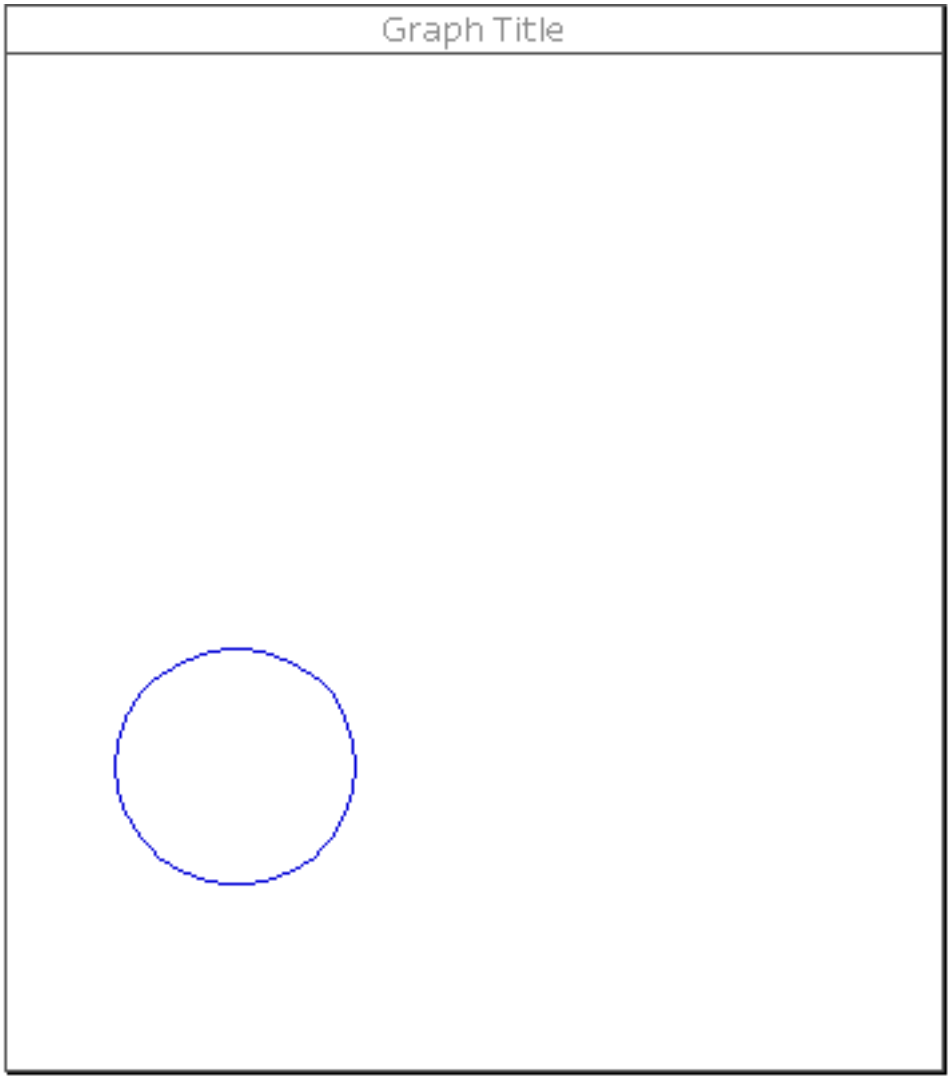


◇ **Figure 19 : System Manager 2 test 1**

The following script draws a complex vector graph. It draw a circle by plotting a sequence of vectors, as opposed to the System Manager 1 example where points were being plotted parametrically.

```
calcman
add circletest
order vector:e^(j*theta)
return
dataman
newrecord record1 theta vector
newdataset circle1 record1 30
newinput input1 0 29 0 0.20943951
newmap map1 theta
vector
return
graphman
newspec circle1
10
5
0
0
10
5
0
0
return
```

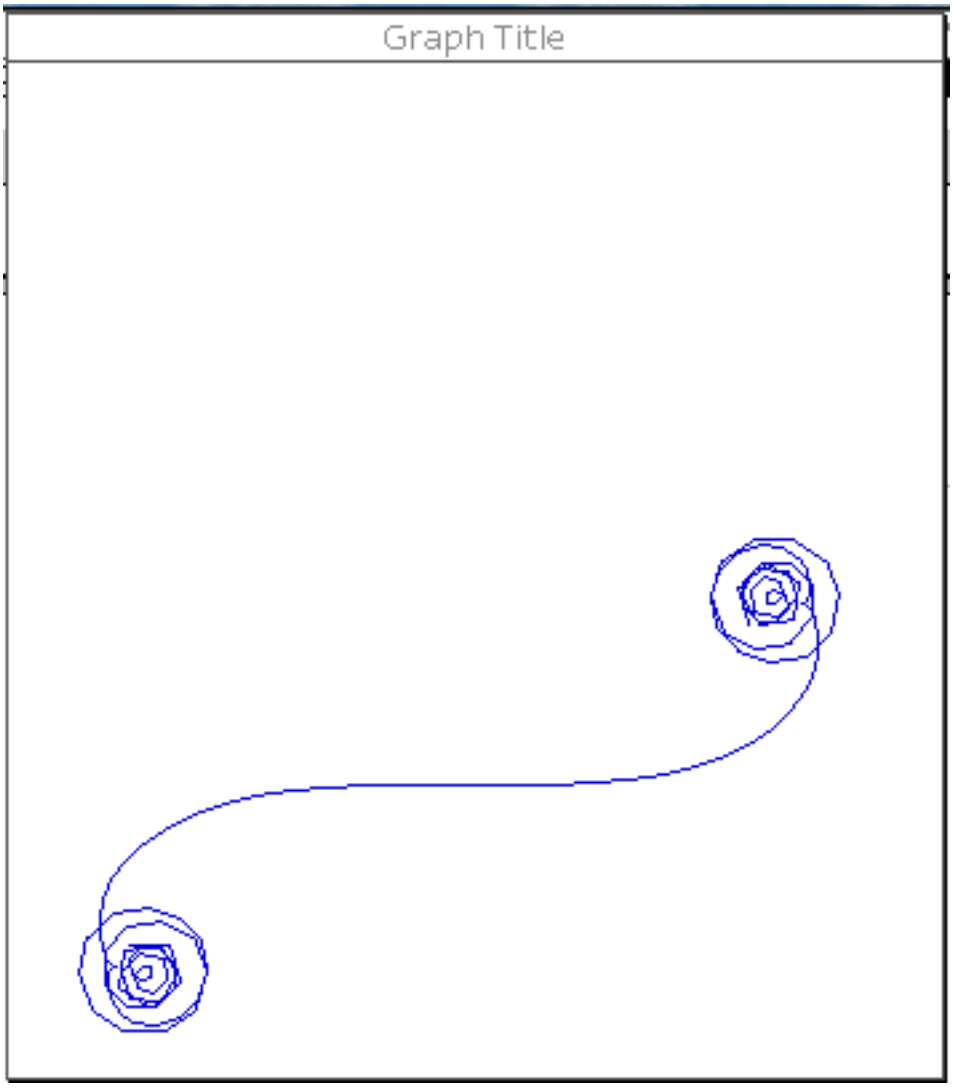
```
setprocess c circletest
setprocess d circle1
setprocess g circle1
setprocess s input1
theta
setprocess m map1
doprocess
graphprocess cornu vector 0 29
```



◇ **Figure 20 : System Manager 2 test 2**

The following script demonstrates how the cornu spiral from the SAW filter analysis is drawn. f0 is the fundamental frequency, set at 100 MHz. The system is excited by frequencies in the range -300 MHz to +300Mhz, in 5MHz steps.

```
calcman
add sawcalc

order x:[(f-f0)/f0]
order sawresponse:sin(pi*f/(f0*2))*e^(j*x^2*pi)
order f0=100 MEGA
return

dataman
newrecord sawcolumns f sawresponse
newdataset sawdata sawcolumns 160
newinput freqvalues 0 159 -300000000 5000000
newmap cornuspiralmap f
sawresponse
return

graphman
newspec cornuspiralgraph
15
1
0
0

15
2
0
0
return
setprocess c sawcalc
setprocess d sawdata
setprocess g cornuspiralgraph
setprocess s freqvalues
f
setprocess m cornuspiralmap
doprocess
graphprocess cornu sawresponse 0 159
```



◇ **Figure 21 : System Manager 2 test 3 - cornu spiral**

Chapter 9: Conclusions

9.1 Evaluation of the Finished System

Developing this system has been a very tough assignment, but rewarding in that it works as intended. The system performs well at what it sets out to do, although there are several areas where optimisation is required. The use of linked lists for all my data structures slows down the program significantly when many objects are present in the lists. This is due to the linear search routines. However, they strike a good compromise as they do not possess the complexity a similarly specified binary tree would have.

A method for parsing an equation once and storing it in a postfix form which does not require recursion to evaluate would help efficiency. There is a particular problem when a summation or prodation is performed, as the expression is repeatedly parsed for each variable value. Removing some of this redundancy would be beneficial, but would make an already complex class even more so.

Speed limitations were most apparent when carrying out the discrete fourier transform tests, due to the presence of the summation.

The use of recusive methods in three areas of the system (equation validator, expression parsing and equation evaluation) has proved to be the most challenging and interesting area of the work. Efficiency was most important in the expression parsing within the calculator. For this reason I deliberately made sure that recursive functions did not use function parameters. A common calculator symbol object was used which was encapsulated inside the calculator class as a data member - so universal access from within the class was guaranteed.

I would have preferred to have spent more time investigating the SAW fiter background material, but due to the sheer size of the system and the accompanying documentation this was not viable. I did state in my introduction that I intended the SAW filter background to provide a context for the project and that role is met satisfactorily.

The main problems encountered with this project were due to complexity. Conceptualising the recursive functionality (especially validator and equation evaluation) and theSystem Manager proved to be most difficult.

Code problems were limited mainly to instances where classes were self-referencing. There really isn’t a satisfactory way to deal with this in C++, and as a result some data members had to be pointers to classes instead of instantiations.

9.2 Further Work

As just mentioned, optimisation is required before the system is developed further. Controls for precision and more flexible graphing would be useful. A graphical user interface would be a substantial development, and due to the very compartmentalised nature of the system would be straightforward to conceptualise and integrate within the existing system.

Beyond this, provision for matrix and statistical operations would be most useful. Establishing a method of communication between calculator objects in the Calculator Manager would also prove interesting.

9.3 Bibliography

9.3 Bibliography

Scientific Papers

- 1. Design of Unapodized Surface-Wave Transducers with Spectral Weighting - Graham R. Nudd, Michael Waldner, R.L. Zimmerman (IEEE Transactions on Microwave Theory and Techniques, January 1974)

Signal Processing Texts

- 2. Contemporary Linear Systems - Strum, Kirk (PWS Publishing Company) 1996
- 3. Radar Systems - Lynn (Macmillan Education) 1987

Software Engineering Texts

- 4. Object-Oriented Analysis and Design - Booch (The Benjamin/Cummings Publishing Company, Inc.) 1994
- 5. Data Structures - Stubbs, Webre (Brooks/Cole Publishing Company) 2nd Ed

C++ Reference Texts

- 6. C++ The Complete Reference - Schildt (McGraw-Hill) 1995 2nd edition
- 7. The C++ Programming Language - Bjarne Stroustrup (Addison-Wesley) 2nd edition

Macintosh Application Programming Reference

- 8. Inside Macintosh Series (CD Rom) - Apple Computer, Inc. (Addison-Wesley)