FnattLabME 2.Release

Helmut Dersch

March 22, 2006

Abstract

This revised version of FnattlabME, the Matlab/Octave/Scilab compatible calculator for mobile devices, has been extended and the user interface has been adapted to run on pointerless phones. Many math functions have been added and memory card based filesystems are now supported (JSR-75). The underlying math engine has been changed from Jampack to Jama with some additions from the Slatec package.

Contents

1 Introduction **2**

2 Installation **2**
2.1 Example 1 3
2.2 Example 2 3

3 Usage **4**
3.1 User Interface 4
3.2 Operators and commands 4
3.2.1 Operators 5
3.2.2 Variables and Constants 5
3.2.3 Language 6
3.2.4 Matrices 6
3.2.5 Functions 7
3.2.6 Polynomials 7
3.2.7 Special functions 8

3.2.8 Plotting 8
3.2.9 Numbers 8
3.2.10 String Functions 8
3.2.11 General Commands 9
3.3 Desktop and Windowmanager 9

4 License **10**

1 Introduction

FnattLabME is a port and extension of August Berings program FnattLab [1] to the J2ME [3]-API. For a basic description see the article describing the first version [2]. This documentation provides a user manual for the new version. Besides many additions, there are also some items which have been removed:

- The builtin ftpclient has been removed since the recent availability of file access (Java API jsr-75) makes file transfer much simpler. A filebrowser and text editor is included in FnattlabME.
- The math engine is now Jama [4] (was Jampack [5]). I made this change because Jampack never seems to leave pre-alpha status, and a few serious bugs could not be corrected. Jama is smaller, but limited to real matrices.
- There is no prepacked Palm version available. For Palms please use the Midlet-installer that comes with the Java Runtime.

2 Installation

To install FnattLabME on a mobile device, first check that a suitable Java runtime is installed and the following requirements are satisfied:

- Java API specs CLDC 1.1, MIDP 2.0, JSR-75.
- Maximum JAR-size $>$ 300kByte.
- Heapsize $>=$ 2MByte

. For Nokia phones, this information can be obtained from Nokia's website [12]. I am running FnattLabME on a Nokia 6230i. Most Palm handhelds are suitable, e.g. the Tungsten E running the IBM-Java runtime. FnattLabME also runs in the WTK2.2 J2ME emulator from Sun [3], and should therefor be useable on many more devices.

Installation depends on the particular device. I suggest to check the manual of your device for Midlet-installation procedures. The following two examples are for Linux-users, who can not run the Windows programs supplied with the mobile devices.

2.1 Example 1

Installation onto Nokia 6230i from Linux desktop PC:

- The Nokia phone has all Java software preinstalled. Enable bluetooth connections for software installation.
- On the Linux PC, install gammu [13] and bluetooth support.
- Enter the directory, which contains the files FnattLab.jar and FnattLab.jad, and execute the command
  `gammu --nokiaaddfile Application FnattLab` You have to acknowledge the installation request on the phone.

2.2 Example 2

Installation onto Palm Tungsten E from Linux desktop PC:

- Download and install the free Java J2ME runtime for Palm PDAs [14].
- Java-Midlets are installed either by packaging them into PALM executables (.prc) and using the standard sync-operation. Alternatively, you can install Midlets using the onboard Midlet installer MidletHQ, which comes with the Java Runtime. I will explain this route in the following.
- On your Linux-desktop, start any HTTP-server and put FnattLabME.jar into the server directory. I use MonkeyWeb on a DSL-installation.
- Establish a TCP-connection between PC and Palm. I use a pppd-over-USB connection. Irda, Bluetooth or WLAN should equally work.

- Start MidletHQ on the Palm, choose “Install MidLet” and type the URL of the file to install, e.g.
  `http://192.168.1.2/FnattLab.jar`.

- The file will be installed after you agree to install from an insecure source (which I certainly am), and you start the program by tapping the desktop icon.

Upon startup FnattLab tries to open and read a file named *path* if present. This should be a plain textfile containing directory paths of locations where FnattLab finds m-files and other data (one full path per line). This file is supposed to be in the folder *vfs*. The file can be created and edited using the build-in text-editor and filebrowser (Use the menu-option *File*). The *vfs*-directory is located in the persistant storage area of the device independent of memory cards. It can only be accessed by the FnattLab-application, but you can copy it to any other location using FnattLab’s filebrowser. By default, FnattLab searches files only in the */res/m* directory. The distribution contains a few files in this location, which can be read but not changed.

3 Usage

3.1 User Interface

User interaction proceeds via the worksheet concept. Commands are edited and typed using a text input frame. On a cellphone this is similar to typing SMS-messages. The command string is supplied to FnattLabME by hitting the Enter button or Enter menu option. FnattLabME’s answer is supplied in a separate window together with a copy of the input. The worksheet is scrollable (options ^ and v), and previous inputs can be recalled in the text input window using the history buttons or menu options labeled < and >. Answers to commands are suppressed if the command is terminated with a semicolon.

3.2 Operators and commands

A large subset of Matlab/Octave/Scilab commands and operators is supported, which are implemented to closely resemble the original. It is suggested to look up their usage in one of the many documentations for Matlab [6], Octave [7] or SciLab [8].

3.2.1 Operators

```
% Comment
x = 3                % assignment/initialisation
[1,2,3]              % row vector
z = [1+i;2;3]        % col vector
z'                   % transpose / complex conjugate
M = [1,2;3,4]        % matrix
M(1,1)               % element
M(1,:)               % row 1
M(:,2)               % col 2
0:0.1:1              % vector from 0 to 1 (inclusive)
                     % in 0.1 steps
-(1:4)               % vector from -1 to -4 in 1.0 steps
*                    % scalar product
+                    % element-wise addition
-                    % element-wise substraction
/                    % right division
\                    % left division
^                    % exponent
.*                   % element-wise multiplication
./                   % element-wise division
.^                   % element-wise exponent
==                   % equals
<>                   % not equal
< > <= >=            % smaller, greater, smaller or
                     % equal, greater or equal
~                    % not
&                    % and
|                    % or
```

3.2.2 Variables and Constants

```
ans             % last output
pi              % 3.14...
i,j             % imaginary unit
realmax, realmin % limits for real numbers
Inf, NaN        % infinity, not a number
```

```
eps            % accuracy of double arithmetics
```

3.2.3 Language

```
if x==0 xp=1; end
if x==0 xp=1; else xp=2; end
if x==0 xp=1; elseif x==1 xp=2; end
eval('x=24')       % evaluate command string
while x<17 x=x+1; end
for i=0:100 x=x+i; end
function y=ttwo(x) % function definition
    y=2*x;         % must be saved in file ttwo.m
nargchk(a,b,c)     % check if a<=c<=b
return, continue, break
error(message)
```

3.2.4 Matrices

```
eye(8,6)           % diag=1, all others 0
zeros(3,2)         % all 0
ones(7)            % all 1
rand(3)            % 3x3 random matrix
randn(3)           % 3x3 normaly distributed random matrix
linspace(x1,x2,n)  % generates n points between x1 and x2
numel(m)           % number of elements
ndims(m)           % dimension
chol(m)            % Cholesky decomposition
det(m)             % determinant
diag(m)            % diagonal elements as vector
[a,b]=qr(m)        % QR-decomposition
[a,b]=lu(m)        % LU-decomposition
eig(m)             % Eigenvalues
svd(m)             % singular value decomposition
sum(m)             % sum columns
cumsum(m)          % cummulative sum of columns
prod(m)            % multiply columns
diff(m)            % subtract columnwise
inv(m)             % invert
```

```
any(m)              % are there non-zero elements
all(m)              % are there zero elements
mpower(a,b)         % a^b
[a,b]=size(m)       % matrix dimensions, a - nrows, b - ncols
length(x)           % length of vector
reshape(A,m,n)      % m-by-n matrix whose elements are taken
                    % columnwise from A.
   % A must have m*n elements.
norm(x)             % vector and matrix norm
y = filter(b,a,x)   % filters vector x with the filter
                    % described by numerator coefficient vector
                    % b and denominator coefficient vector a
[y,zf] = filter(b,a,x) % final state values in vector zf.
ismatrix(x)
```

3.2.5 Functions

```
imag(x),real(x),abs(x),conj(x),angle(x)
sign(x), floor(x), ceil(x)
sqrt(x), sin(x), asin(x), cos(x), tan(x)
atan(x), acos(x), exp(x), log(x), sinh(x)
cosh(x), tanh(x), asinh(x), acosh(x), atanh(x)
gamma(x), gammaln(x)
gcd(a,b),lcm(a,b),choose(n,k),factorial(n)
acsc(x),acsch(x),csc(x),sch(x),sec(x),sech(x),asec(x),
csch(x),asech(x), min(x,y), max(x,y)
```

3.2.6 Polynomials

```
poly(A)      % If A is a square n-by-n matrix, poly (A)
             % is the row vector of the coefficients of
             % det (z * eye(n) - A). If x is a vector,
             % poly (x) is a vector of coefficients of
             % the polynomial whose roots are the
             % elements of x.
polyval(c,x) % Value of polynomial with coefficients c
             % at point x
roots(c)     % roots of polynomial with coefficients c
```

3.2.7 Special functions

```
quad('sin(x)',0,2*pi) % numeric integration of sin(x) using
              % Simpson's rule (n=200)
fzero('x*x-1',0) % function zero with initial estimate,
              % uses Slatec DFZERO
fzero('x*x-1',[-2,2])
fzero('x*x-1',[-2,2],struct('TolX',1e-10))
fft(x)        % Discrete Fourier transform
fft(x,n)
sort(x)       % sort vector, or columns of matrix (bubblesort)
```

3.2.8 Plotting

```
plot(x,y)     % Plot y vs. x with options. Options may
              % be used to specify
plot(x,y,'*g')% markers (o,x,+,*) and/or colors(r,g,b,c,m,y,w).
ezplot('2*x',[0,5]) % Easy plotting of y=2*x
ezplot('x.*x',[0,5,2,3])
```

3.2.9 Numbers

```
primes(x)        % factors of x
rat(x)           % rational approximation
rat(x,tol)
sieve(100)       % all primes <= 100
fix(x,y)         % integer divide
rem(x,y)         % remainder of integer divide
```

3.2.10 String Functions

```
lower('ABC')     % to lower case
lower('abc')     % to upper case
deblank('  a  ') % remove blanks
blanks(7)        % 7 spaces
findstr(a,b)     % find pos of a in b or vice versa
strcmp(a,b)      % compare a and b
setstr(a)        % create Stringmat
str2mat(a)       % convert string to Stringmat
```

```
strrep(a,b,c)    % replace c for b in a
strtok(a)
strtok(a,del)    % first token in a, default del=" "
int2str(6)       % integer to string
num2str(3.24)    % number to string
sprintf('x is %f',x)
x=sscanf(s,'%f')
x=str2num('1.3') % string to number
dec2hex('10.3')  % decimal to hex string
hex2dec('1fa')   % hex to decimal string
x=hex2num('1fa') % hex string to number
isletter(x)
toascii('a')     % ascii code of letter 'a'
```

3.2.11 General Commands

```
help('plot')    % works for only few commands
who             % list all variables
load('file')    % load variable definitions
save('file')    % save all variables
s=struct('field',val) % define data structure
setfield(x,'f',val) % set field f in struct x to val
getfield(x,'f')  % get field f in struct x
exist(a), isempty(x), isfinite(y)
isinf(z), isstr(s), isequal(a,b)
```

3.3 Desktop and Windowmanager

Plotting is accomplished using the *plot* command or the derived command*ezplot*. The syntax for *plot* is `plot(x,y,options)`. Options may be used to specify markers (`o,x,+,*`) and/or colors(`r,g,b,c,m,y,w`). Example:

```
x = 0:.2:2*pi;  % Create array of x-values
y = sin(x);     % Create array of y-values
plot(x,y,'*m'); % plot using magenta stars
plot(x,y,'b');  % plot using blue line
```

The command `ezplot` is defined as m-file.
Usage: `ezplot('sin(x)',[0,2*pi])`. The function to plot must be supplied as string, and the argument must be `x`.

After executing *plot* a desktop with windows is displayed, and the graph is plotted in the top window. You can return to the shell by opening the menu in the lower right. The desktop is not destroyed and can be reentered by using the `Desktop`-item in the worksheet menu. Subsequent plot commands plot into the topmost graph-window.

On a device with pointer (PDA) the desktop is managed similar to using a mouse on a PC. Windows may be scaled, moved, minimized, or closed as usual. Coordinates in the window can be made visible by clicking into the window area. By pressing the blue button, the pointer may be used to create and move a straight line, whose polynomial coefficients are displayed. This feature may be used to visually analyse graphs. By pressing the button again, pointer coordinates are redisplayed. On a pointerless device (cellphone), the coodinates of a moveable mark can be displayed. For this and for window managment the following key actions are available:

Key Enter Toggle mark. If menu is shown: select menu item.

Navigation Keys Move mark. If menu is shown, change selected item.

Key 0 Cycle through Windows.

Key 1 Toggle menu. Choices are `Close` to close the front window, and `exit` to return to the worksheet.

Key 3 Toggle window size.

Key 7 Move window to the right. It wraps and reappears on the left side.

Key 9 Move window down. It wraps and reappears on the top.

Key * Increase window size.

Key # Decrease window size.

4 License

FnattLabME is distributed under GNU-license, see the included license text. Some source files are taken from other open projects which have their own license:

- FnattLabME uses the Jama [4] numerical package. From the copyright note: This software is a cooperative product of The MathWorks and the National Institute of Standards and Technology (NIST) which has been released to the public domain.
- The files Random.java, and MPN.java are slightly modified copies from the GNU-Classpath [16] project. This project is also distributed under GNU-license.
- The file JMath.java is a pure Java implementation of the java.lang.Math class. It is derived from open sources and copyright Visual Numerics Inc [15]. See the source file for details of its license.
- poly.m, roots.m are ports of the corresponding Octave [10] files.
- Blasf77.java and Fzero.java is a port of Slatec dfzero to Java by Steve Verrill [17].

References

[1] http://reactos.ccp14.ac.uk/projects/fnattlab/

[2] http://www.hs-furtwangen.de/ dersch/FnattLabME/FnattLabME.pdf

[3] http://java.sun.com/j2me/

[4] http://math.nist.gov/javanumerics/jama/

[5] ftp://math.nist.gov/pub/Jampack/Jampack/AboutJampack.html

[6] http://www.utexas.edu/math/Matlab/Manual/ReferenceTOC.html

[7] http://www.octave.org/doc/index.html

[8] http://www-pool.math.tu-berlin.de/doc/scilab/manual/

[9] www.mathworks.com

[10] www.octave.org/

[11] www.hs-furtwangen.de/ dersch

[12] www.forum.nokia.com/main/0,,150,00.html?matrixType=midp2

[13] www.mwiacek.com/gsm/soft/gammu.html

[14] www.palm.com/us/support/jvm/

[15] www.vni.com/

[16] www.gnu.org/software/classpath/

[17] www.netlib.org