ADDENDUM A

INTRODUCTION TO ADDENDUM A

This document contains the input data used in the Universal SCF determination in Methodology II, (Chapter 4), and the input data and MATLAB simulation program used in Methodology III, (Chapter 6). The document also defines the various parameters used in these practical sessions.

UNIVERSAL SCF Determination Data

The input data used in Methodology I, is available in Chapter 3 of the dissertation document.

Calculation of Universal SCF Data for Methodology II

This consisted of Probability of Error, P_b, and K the number of users. Equation (4-26) in Chapter 4, was employed in these calculations. The data in Tables 1-1 to 1-5, was used to plot the Probability of Error versus K graphs of Figures 4-1 to 4-11, in Chapter 4 of the dissertation document.

K users	Pb at SCF=0.47	Pb at SCF=0.48	Pb at SCF=0.49	Pb at SCF=0.50	Pb at SCF=0.51	Pb at SCF=0.52	Pb at SCF=0.53	Pb at SCF=0.54
5	4.94E-04	4.71E-04	4.50E-04	4.30E-04	4.12E-04	3.95E-04	3.80E-04	3.66E-04
10	3.86E-03	3.72E-03	3.60E-03	3.50E-03	3.40E-03	3.33E-03	3.26E-03	3.21E-03
15	1.43E-02	1.43E-02	1.43E-02	1.44E-02	1.45E-02	1.47E-02	1.51E-02	1.55E-02
20	3.83E-02	3.94E-02	4.60E-02	4.25E-02	4.44E-02	4.67E-02	4.92E-02	5.21E-02
25	8.38E-02	8.84E-02	9.35E-02	9.93E-02	1.06E-01	1.12E-01	1.20E-01	1.28E-01
30	1.61E-01	1.65E-01	1.75E-01	1.86E-01	1.98E-01	2.10E-01	2.32E-01	2.35E-01

Table A-1: Calculated Data for K and P_b, (range SCF =0.47 to 0.54)

K users	Pb at SCF=0.55	Pb at SCF=0.56	Pb at SCF=0.57	Pb at SCF=0.58	Pb at SCF=0.59	Pb at SCF=0.60	Pb at SCF=0.61	Pb at SCF=0.62
5	3.53E-04	3.40E-04	3.29E-04	3.19E-04	3.10E-04	3.02E-04	2.95E-04	2.88E-04
10	3.17E-03	3.15E-03	3.14E-03	3.14E-03	3.16E-03	3.19E-03	3.24E-03	3.30E-03
15	1.59E-02	1.65E-02	1.72E-02	1.80E-02	1.89E-02	2.00E-02	2.12E-02	2.25E-02
20	5.54E-02	5.90E-02	6.30E-02	6.74E-02	7.23E-02	7.76E-02	8.33E-02	8.94E-02
25	1.37E-01	1.46E-01	1.56E-01	1.66E-01	1.76E-01	1.87E-01	1.99E-01	2.10E-01
30	2.48E-01	2.61E-01	2.74E-01	2.87E-01	2.99E-01	3.12E-01	3.24E-01	3.35E-01

Table A-2: Calculated Data for K and P_b, (range SCF =0.55 to 0.62)

K users	Pb at SCF=0.63	Pb at SCF=0.64	Pb at SCF=0.65	Pb at SCF=0.66	Pb at SCF=0.67	Pb at SCF=0.68	Pb at SCF=0.69	Pb at SCF=0.70
5	2.83E-04	2.78E-04	2.74E-04	2.70E-04	2.68E-04	2.66E-04	2.64E-04	2.64E-04
10	3.37E-03	3.47E-03	3.58E-03	3.72E-03	3.87E-03	4.05E-03	4.25E-03	4.49E-03
15	2.41E-02	2.57E-02	2.76E-02	2.97E-02	3.20E-02	3.60E-02	3.73E-02	4.03E-02
20	9.60E-02	1.03E-01	1.10E-01	1.18E-01	1.26E-01	1.35E-01	1.44E-01	1.53E-01
25	2.22E-01	2.33E-01	2.45E-01	2.56E-01	2.68E-01	2.79E-01	2.91E-01	3.01E-01
30	3.47E-01	3.58E-01	3.68E-01	3.78E-01	3.87E-01	3.96E-01	4.04E-01	4.12E-01

Table A-3: Calculated Data for K and P_b, (range SCF =0.63 to 0.70)

K users	Pb at SCF=0.71	Pb at SCF=0.72	Pb at SCF=0.73	Pb at SCF=0.74	Pb at SCF=0.75	Pb at SCF=0.76	Pb at SCF=0.77	Pb at SCF=0.78
5	2.64E-04	2.65E-04	2.66E-04	2.69E-04	2.72E-04	2.76E-04	2.80E-04	2.86E-04
10	4.75E-03	5.04E-03	5.37E-03	5.73E-03	6.14E-03	6.60E-03	7.10E-03	7.66E-03
15	4.35E-02	4.70E-02	5.08E-02	5.49E-02	5.92E-02	6.39E-02	6.88E-02	7.40E-02
20	1.62E-01	1.72E-01	1.81E-01	1.91E-01	2.01E-01	2.11E-01	2.21E-01	2.31E-01
25	3.12E-01	3.23E-01	3.33E-01	3.42E-01	3.52E-01	3.61E-01	3.69E-01	3.77E-01
30	4.20E-01	4.27E-01	4.33E-01	4.39E-01	4.44E-01	4.50E-01	4.54E-01	4.59E-01

Table A-4: Calculated Data for K and P_b, (range SCF =0.71 to 0.78)

K users	Pb at SCF=0.79	Pb at SCF=0.80	Pb at SCF=0.81	Pb at SCF=0.82	Pb at SCF=0.83	Pb at SCF=0.84	Pb at SCF=0.85	Pb at SCF=0.86
5	2.92E-04	3.00E-04	3.08E-04	3.18E-04	3.28E-04	3.40E-04	3.53E-04	3.68E-04
10	8.28E-03	8.96E-03	9.70E-03	1.05E-02	1.14E-02	1.24E-02	1.35E-02	1.46E-02
15	7.94E-02	8.51E-02	9.11E-02	9.74E-02	1.04E-01	1.10E-01	1.17E-01	1.24E-01
20	2.41E-01	2.55E-01	2.60E-01	2.70E-01	2.79E-01	2.89E-01	2.98E-01	3.06E-01
25	3.85E-01	3.93E-01	4.00E-01	4.07E-01	4.13E-01	4.19E-01	4.25E-01	4.30E-01
30	4.63E-01	4.67E-01	4.70E-01	4.73E-01	4.76E-01	4.79E-01	4.81E-01	4.84E-01

Table A-5: Calculated Data for K and P_b, (range SCF =0.79 to 0.86)

The values of spreading gain $N = 31$, and $E_B / N_O = 10dB$, were used in the calculations. The graphs were plotted in Microsoft Excel.

Centre For Radio & Digital Communication (CRDC)
Department of Electrical, Electronic & Computer Engineering
University of Pretoria

P^3MUD (PPPMUD) MATLAB Program

Overview of P^3MUD MATLAB Program Structure

The program to simulate the P^3MUD outputs was originally obtained from the shareware MATLAB software employed by Sridhar Rajagopal *et al*.[7], for determining the CDMA/WCDMA performances at Rice University, Houston, Texas. Thus, the P^3MUD program uses the structures of the shareware program as a base to build on, and the program modules that have been developed are similar in some instances to the original modules, but different in others as seen in figure A-1, and in the comparison of Table A-6.

ITEM	PPPMUD	Conventional PIC MUD	Comment
1	tester.m		For PPPMUD, tests validity of input data functions in Information4.m
2	Information4.m		For PPPMUD, main excecutible program, controlling input and output data and GUI activation. It also contains the Conventional Matched Filter Receiver program code and the Single-user optimum detector code for the respective error bound graphs.
3	Optimum_detect.m	detect_pipe.m	For Conventional MUD, main excecutible program, but for PPPMUD is a function where code syntax has been optimised. Program generates channel estimate parameters for the randomly selected users' data which is spread before transmission over channel. Program performs Maximum Likelihood channel estimation process and outputs used in Matched Filter & Multistage PIC Detector stages. Resultant Output is detected bits and error count from detection process, as well as graphical BER vs SNR, with additional BER vs K (simultaneous users) for PPPMUD
4	gendata.m	gendata.m	Function for generating randomly selected users and data and spreading codes, channel estimate parameters to be used in detection process
5	genseq.m	genseq.m	function to generate spreading code sequences
6	splitAmatrix.m	SplitAmatrix.m	Function to assist in manipulating matrices efficiently in detection process calculations
7	SCF.m		For PPPMUD, function activates the soft cancellation factor compensator mechanism

Table A-6: Comparison of P^3MUD and Conventional PIC MUD program structures

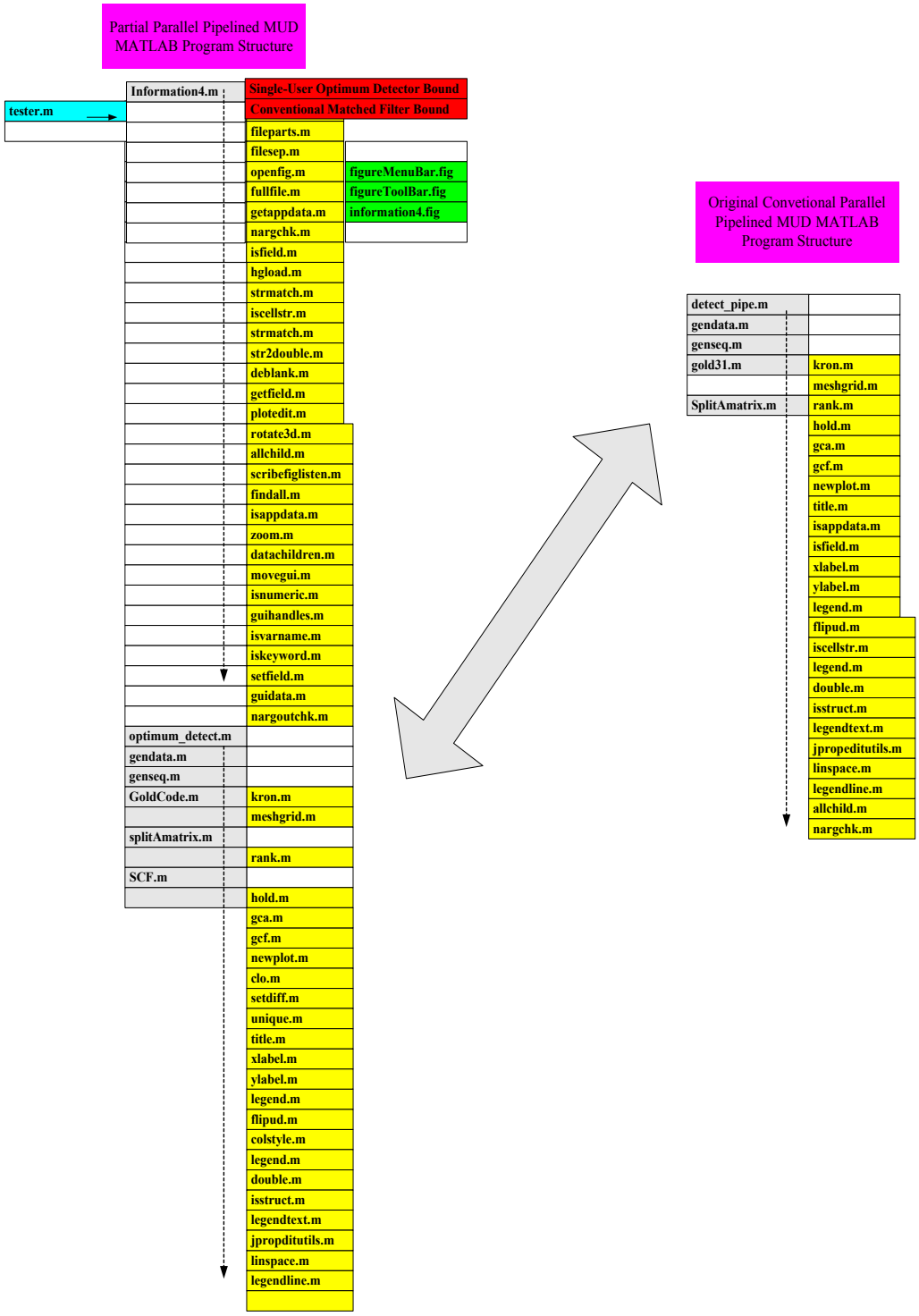

Figure A-1: Overview of P^3MUD MATLAB Program Structure

P^3MUD Program Description

The program basically simulates a CDMA system in MATLAB, with the process consisting of the transmission of randomly selected user data bits over a channel, and detection of those bits at the receiver side by comparing the preamble and transmitted bits. The number of errors is counted and forms the input to the BER output graphs, which is a measure of the efficiency of the detection scheme. The standard optimum detection graphs are also included as a comparison measure.

MATLAB Program Assumptions

The MATLAB code version assumes a simplistic AWGN channel model, which supports Walsh and Gold codes for asynchronous detection. Also no pulse shaping or fading is assumed, with the receiver consisting of a single antenna case. Multi-path propagation is supported. There is no source/channel coding for the algorithms used.

Multiuser Channel Estimation and Detection

If we assume Binary Phase Shift Keying (BPSK) modulation for the transmitted signal, the scheme uses spread spectrum signalling, where each active user uses a unique signature sequence or spreading code to modulate the data bits. The receiver system receives a summation of the signals of all the active users after they travel through different paths in the channel. These paths will induce different delays, attenuations and phase-shifts to their signals. Also, the signals from different users interfere with each other in addition to the Additive White Gaussian noise (AWGN) present in the channel. Thus, the multiuser channel estimation consists of the joint estimation of these unknown parameters for all users in order to mitigate these undesirable effects and accurately detect the received bits of the different users. Also, the multiuser detection refers to the detection of received bits for all users jointly by cancelling the interference between the different users.

In the MATLAB program, the channel estimation scheme is enhanced for an iterative scheme that avoids matrix inversions and has a simple fixed point architecture. The multiuser detection is enhanced for a highly pipelined and bit-oriented scheme.

Another advantage of scheme is that there is no division except for multiplication by convergence parameter μ which can be implemented as a right-shift, by making it a power of two. This can be done as the algorithm converges for a wide range of μ. For simulation purposes, a Gold code of length 31 was assumed. This code was chosen for WCDMA [19] as it has desirable auto-correlation and cross-correlation properties for asynchronous transmission scheme.

Pipelined Multiuser Detection Process

The stages in the multistage detector are efficiently pipelined to avoid edge computations and to work on a bit streaming basis. This is equivalent to the normal detection of a block of infinite length, detected is a simple pipelined fashion. Also the computations can be reduced to work on smaller matrix sets. The computations performed on the intermediate bits reduce to

$$L = \Re\left[\hat{\mathrm{A}}_1^H \hat{\mathrm{A}}_0\right]. \tag{A.1}$$

$$C = \Re\left[\hat{\mathrm{A}}_0^H \hat{\mathrm{A}}_0 + \hat{\mathrm{A}}_1^H \hat{\mathrm{A}}_1 - diag\left(\hat{\mathrm{A}}_0^H \hat{\mathrm{A}}_0 + \hat{\mathrm{A}}_1^H \hat{\mathrm{A}}_1\right)\right]. \tag{A.2}$$

$$y_i^{(l)} = y_i^{(0)} - L\hat{d}_{i-1}^{(l-1)} - C\hat{d}_i^{(l-1)} - L^T \hat{d}_{i+1}^{(l-1)}. \tag{A.3}$$

$$y_i^{(l)} = y_i^{(0)} - (SCF)L\hat{d}_{i-1}^{(l-1)} - (SCF)C\hat{d}_i^{(l-1)} - (SCF)L^T \hat{d}_{i+1}^{(l-1)}. \tag{A.4}$$

$$\hat{d}_i^{(l)} = sign = (y_i^{(l)}). \tag{A.5}$$

where $y_i^{(l)}$ and $\hat{d}_i^{(l)}$ are the soft and hard decisions respectively, after each stage of the multistage detector. *Refer to Chapter 5 and 6 for other symbol definitions*. These computations are iterated for $l = 1,2,\cdots,M$ where M is the maximum number of iterations chosen for convergence (typically, 3). Equation (A.3) represents subtracting the interference from the past bit of users, who have more delay, and the future bits of the users, who have less delay than the desired user. Equation (A.4) is the Soft Cancellation implementation of equation (A.3).

Centre For Radio & Digital Communication (CRDC)
Department of Electrical, Electronic & Computer Engineering
University of Pretoria

Running the P^3MUD Program and Parameter Settings

The Program was run on MATLAB Version 6.1.0, Release 12.1. Typing in the command 'information4', starts the execution of the program. Depending on the number of bits being transmitted, if less than 100000, than the program can be executed on a Pentium III, 500MHz processor system, with the execution time dependent on the bits transmitted. A virtual memory setting of 900 Mb is sufficient. This is done by varying the 'Ld' variable in the 'Optimum_detect' program. For transmitted bits above 100000, precisely 2000000 bits, this was done on a Dell with Intel Xeon Processor 2.20 GHz. The virtual memory was set to 4 Gb. The program execution took 40 minutes for different settings of the SCF value. The parameters are input in the GUI after it is called up as shown in the diagram below. The simulations used a μ value of 1/300, and the number of paths P=4. For the Single-user Optimum Detector Bound, $\rho = 0$, was used, and for the Conventional Matched Filter Bound, $\rho = 0.16$, was used.

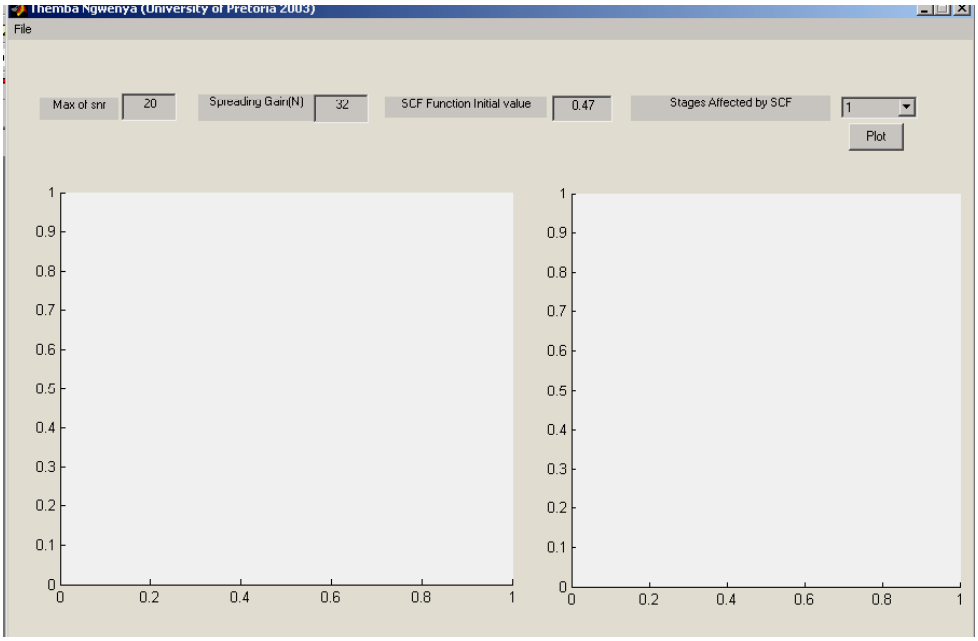

Figure A-2: Default Program GUI to enter variable data

Centre For Radio & Digital Communication (CRDC)
Department of Electrical, Electronic & Computer Engineering
University of Pretoria

That is for example, for constant number of users *k*, the BER is varied with the SNR, and also, for constant SNR, the BER is varied with *k*. Also input is the SCF value. If the Compensator mechanism is to be used, when using the $SCF_{UNV} = 0.57$ value, then the SCF.m file is loaded with the appropriate fractional values, i.e 0.15, 0.25..etc, otherwise these fractional values can be set to zero when varying other SCF values not SCF_{UNV}.

A sample of the output graphs is shown in figure A-3.

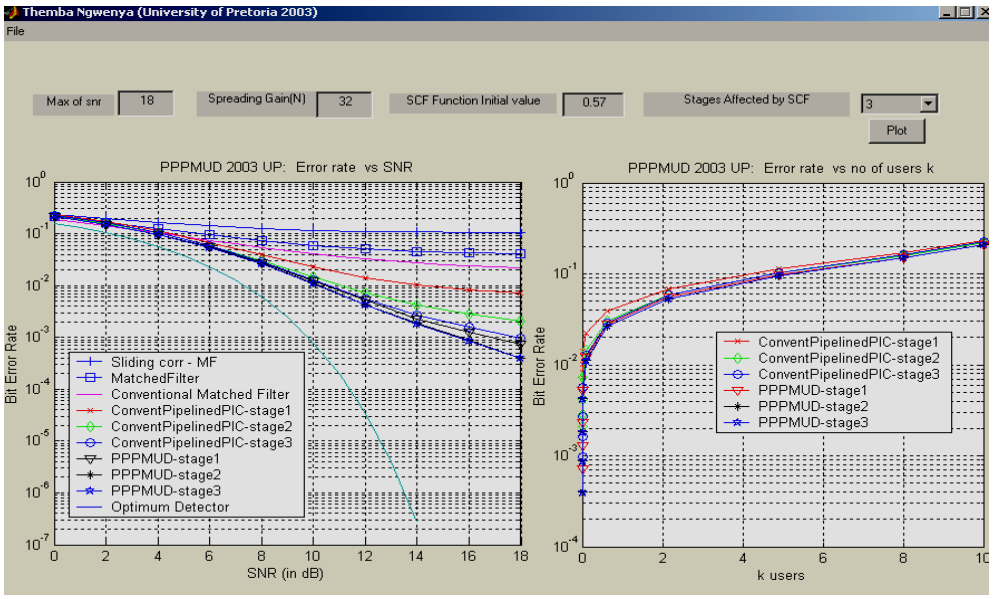

Figure A-3: BER vs SNR at constant k (k=10), and BER vs k at constant SNR (SNR=18).

It is seen from figure A-3, that the SCF_{UNV} value of 0.57 is being used, indicating that the compensator mechanism is also in use. The output graphs indicate the comparisons between the various detection schemes, i.e.,

For the BER versus SNR case:

- Sliding correlator Matched Filter
- The Matched Filter (first stage of Multiuser Detectors)
- The Verdu Matched Filter Approximate Detector
- Three stages of Conventional Bit-Streaming Pipe-lined Multiuser detection
- Three stages of Partial Parallel Pipelined Multiuser Detection
- The Verdu Single-user Optimum Detector

For the BER versus k case:

- Three stages of Conventional Bit-Streaming Pipelined Multiuser detection
- Three stages of Partial Parallel Pipelined Multiuser Detection

MATLAB Program

The MATLAB Program Modules that have been attached are:

- Information4.m
- Tester.m
- Optimum_detect.m
- Gendata.m
- Genseq.m
- SCF.m
- SplitAmatrix.m

The software version is also included and will be available from my supervisor, Professor Louis Linde, at the Centre For Radio & Digital Communication (CRDC) at the Department of Electrical, Electronic & Computer Engineering, University of Pretoria.

Information4.m

```
function varargout = Tester(varargin)
% INFORMATION4 Application M-file for information4.fig
%    FIG = INFORMATION4 launch information4 GUI.
%    INFORMATION4('callback_name', ...) invoke the named callback.
% Last Modified by Themba v2.0 27-April-2004 22:43:32
if nargin == 0  % LAUNCH GUI
                fig = openfig(mfilename,'reuse');
                % Generate a structure of handles to pass to callbacks, and store it.
                handles = guihandles(fig);
                guidata(fig, handles);
                if nargout > 0
                                varargout{1} = fig;
                end
elseif ischar(varargin{1}) % INVOKE NAMED SUBFUNCTION OR CALLBACK
                try
                                if (nargout)
                                                [varargout{1:nargout}] = feval(varargin{:}); %
FEVAL switchyard
                                else
                                                feval(varargin{:}); % FEVAL switchyard
                                end
                catch
                                disp(lasterr);
                end
end

%| ABOUT CALLBACKS:
%| GUIDE automatically appends subfunction prototypes to this file, and
%| sets objects' callback properties to call them through the FEVAL
%| switchyard above. This comment describes that mechanism.
%%| Each callback subfunction declaration has the following form:
%| <SUBFUNCTION_NAME>(H, EVENTDATA, HANDLES, VARARGIN)
%%| The subfunction name is composed using the object's Tag and the
%| callback type separated by '_', e.g. 'slider2_Callback',
%| 'figure1_CloseRequestFcn', 'axis1_ButtondownFcn'.
%%| H is the callback object's handle (obtained using GCBO).
%%| EVENTDATA is empty, but reserved for future use.
%%| HANDLES is a structure containing handles of components in GUI using
%| tags as fieldnames, e.g. handles.figure1, handles.slider2. This
%| structure is created at GUI startup using GUIHANDLES and stored in
%| the figure's application data using GUIDATA. A copy of the structure
%| is passed to each callback.  You can store additional information in
%| this structure at GUI startup, and you can change the structure
%| during callbacks.  Call guidata(h, handles) after changing your
%| copy to replace the stored original so that subsequent callbacks see
%| the updates. Type "help guihandles" and "help guidata" for more
%| information.
%|%| VARARGIN contains any extra arguments you have passed to the
%| callback. Specify the extra arguments by editing the callback
%| property in the inspector. By default, GUIDE sets the property to:
%| <MFILENAME>('<SUBFUNCTION_NAME>', gcbo, [], guidata(gcbo))
%| Add any extra arguments after the last argument, before the final
%| closing parenthesis.
```

```
% --------------------------------------------------------------------
function varargout = pushbutton1_Callback(h, eventdata, handles, varargin)
% this function is called when the plot button is pressed
%i1=get(handles.txtNoUsers,'String');
% first do some validation tests
   % test  for a valid value of the no of users
   if isempty(str2num(get(handles.txtNoUsers,'String')))
      set(handles.errmsg,'Visible','on');
      set(handles.errmsg,'String','no of users not numeric'); % on error display error message
      return ;
   elseif floor(str2num(get(handles.txtNoUsers,'String'))) > 32 |
floor(str2num(get(handles.txtNoUsers,'String')))<2
      set(handles.errmsg,'Visible','on');
      set(handles.errmsg,'String','no of users greater than 32  or less than 2 not supported'); % on
error display error message
      return;
   end
   %test for a valid value for the spread gain
   if isempty(str2num(get(handles.txtSpreadGain,'String')))
      set(handles.errmsg,'Visible','on');
      set(handles.errmsg,'String','Spread Gain not numeric'); % on error display error message
      return ;
   elseif floor(str2num(get(handles.txtSpreadGain,'String'))) > 32 |
floor(str2num(get(handles.txtSpreadGain,'String'))) <=0
      set(handles.errmsg,'Visible','on');
      set(handles.errmsg,'String','invalid sread gain cannot be less than or equal to zero or greater
than 32');
      return;
   end
   %test for a valid value for the SCF starting value
   if isempty(str2num(get(handles.txtSCF,'String')))
      set(handles.errmsg,'Visible','on');
      set(handles.errmsg,'String','SCF start not numeric'); % on error display error message
      return ;
   elseif str2num(get(handles.txtSCF,'String')) > 1 | str2num(get(handles.txtSCF,'String')) <= 0
      set(handles.errmsg,'Visible','on');
      set(handles.errmsg,'String','invalid value for SCF cannot be less than or equal to zero or
greater than one');
      return;
   end
      set(handles.errmsg,'Visible','off'); % if no error make the error textbox invisible
      maxsnr=0;
      maxsnr=floor(str2num(get(handles.txtNoUsers,'String'))); %get the no of user
      spreadgain=0;
      spreadgain=floor(str2num(get(handles.txtSpreadGain,'String'))); %get the spread gain
      SCFstart=0.47;
      SCFstart=str2num(get(handles.txtSCF,'String')); % get the correlator initial value
      stages=get(handles.cmbStages,'Value');
       maxnousers=0;  % initialise max no of users
               maxnousers=floor(maxsnr/2) + 1; % get the maximum no of users
         i=1;
                  for i=1:maxnousers % fill matrix with the no of users
                    kval(i)=(maxnousers) * (((maxnousers - i)/maxnousers)^(i-1));
```

```
                end

  [error_slvec,error_vec,err1_vec,err2_vec,err3_vec,epart1_vec,epart2_vec,epart3_vec,snr] = ...
        optimum_detect(maxnousers,spreadgain,SCFstart,stages); % function call to do the required
calculations to return the parameters

    %MATLAB code for Single User Optimum Detector:
ro=0.00;
range = [0 14];
k=[0 1 2 6];
for na1=range(1):range(2)
nna1(na1-range(1)+1) = 10^(na1/10);
end;
[k1 k2]=size(k);
plt = zeros(range(2)-range(1),k2);
for i=1:k2
for j=1:range(2)-range(1)+1
a1=nna1(j);
a2=k(i)*nna1(j);
if (a1>a2*ro)
plt(j,i)=0.25*erfc(sqrt(a1-a2*ro)/sqrt(2))+0.25*erfc(sqrt(a1+a2*ro)/sqrt(2));
end;
if(a1<= a2*ro)
plt(j,i)=0.5*(1-erfc(sqrt(a2*ro-a1)/sqrt(2))/2)+0.25*erfc(sqrt(a1+a2*ro)/sqrt(2));
end;
end;
end;
n=range(1):range(2);

   %MATLAB for Conventional Matched  Filter:
ro=0.16;
range=[0 18];
nou = 10;

for i=range(1):range(2);

nna1(i-range(1)+1)=sqrt(10^(i/10));
end;
for k=1:(range(2)-range(1)+1)
end;

xr =range(1):range(2);
for i=range(1):range(2)
temp(i-range(1)+1)=0.5*erfc(nna1(i-range(1)+1)/(sqrt(2)*sqrt(1+(nou-1)*ro^2*nna1(i-
range(1)+1)^2)));
end;



      axes(handles.axes1);   % set axes 1 as the current drawing view
 hold off; % erase any previous drawings

 semilogy(snr,error_slvec,'-+');   % plot graph with y axis in logarithmic plot of base 10 (sliding
correlator),x snr
```


Centre For Radio & Digital Communication (CRDC)
Department of Electrical, Electronic & Computer Engineering
University of Pretoria

```
 hold on; % dont erase the drawing view draw on top
  semilogy(snr,error_vec,'-s'); % plot graph with y axis in logarithmic plot of base 10 (error
vector),x snr
 hold on; % dont erase the drawing view draw on top
 semilogy(xr, temp,'-m'); %plot Conventional Matched  Filter analytical results
 hold on;
  semilogy(snr,err1_vec,'-xr');  % plot graph with y axis in logarithmic plot of base 10 (PIC 1),x snr
 hold on; % dont erase the drawing view draw on top
 semilogy(snr,err2_vec,'-dg');  % plot graph with y axis in logarithmic plot of base 10 (PIC 2),x snr
 hold on;  % dont erase the drawing view draw on top
 semilogy(snr,err3_vec,'-o');  % plot graph with y axis in logarithmic plot of base 10 (PIC 3),x snr
 hold on;  % dont erase the drawing view draw on top
 semilogy(snr,epart1_vec,'-vk');  % plot graph with y axis in logarithmic plot of base 10 (epart1),x
snr
 hold on;  % dont erase the drawing view draw on top
 semilogy(snr,epart2_vec,'-*k');  % plot graph with y axis in logarithmic plot of base 10 (epart2),x
snr
 hold on;  % dont erase the drawing view draw on top
 semilogy(snr,epart3_vec,'-p');  % plot graph with y axis in logarithmic plot of base 10 (epart3),x
snr
 hold on;  % dont erase the drawing view draw on top
 grid
 semilogy(n,plt,'-');%Plot single user optimum detector
hold on;

title('PPPMUD 2003 UP:  Error rate  vs SNR');
 xlabel('SNR (in dB)');
 ylabel('Bit Error Rate');
 legend('Sliding corr - MF','MatchedFilter','Conventional Matched Filter','ConventPipelinedPIC-
stage1','ConventPipelinedPIC-stage2','ConventPipelinedPIC-stage3','PPPMUD-stage1','PPPMUD-
stage2','PPPMUD-stage3','Optimum Detector');



 axes(handles.axes2); % make axes 2 the current drawing view
 hold off; % erase any previous drawings on axis

 semilogy(kval,err1_vec,'-xr');  % plot graph with y axis in logarithmic plot of base 10 (PIC 1),x
users k
 hold on;  % dont erase the drawing view draw on top
 semilogy(kval,err2_vec,'-dg'); % plot graph with y axis in logarithmic plot of base 10 (PIC 2),x
users k
 hold on;  % dont erase the drawing view draw on top
 semilogy(kval,err3_vec,'-o'); % plot graph with y axis in logarithmic plot of base 10 (PIC 3),x
users k
 hold on;  % dont erase the drawing view draw on top
 semilogy(kval,epart1_vec,'-vr'); % plot graph with y axis in logarithmic plot of base 10 (epart1),x
users k
 hold on;  % dont erase the drawing view draw on top
 semilogy(kval,epart2_vec,'-*k'); % plot graph with y axis in logarithmic plot of base 10 (epart2),x
users k
 hold on;  % dont erase the drawing view draw on top
 semilogy(kval,epart3_vec,'-p');% plot graph with y axis in logarithmic plot of base 10 (epart3),x
users k
```

```
 hold on;  % dont erase the drawing view draw on top
 grid
 hold on;
 title('PPPMUD 2003 UP:  Error rate  vs no of users k');
 xlabel('k users');
 ylabel('Bit Error Rate');
 legend('ConventPipelinedPIC-stage1','ConventPipelinedPIC-stage2','ConventPipelinedPIC-
stage3','PPPMUD-stage1','PPPMUD-stage2','PPPMUD-stage3');


% --------------------------------------------------------------------
function varargout = txtNoUsers_ButtonDownFcn(h, eventdata, handles, varargin)
% --------------------------------------------------------------------
function varargout = txtNoUsers_Callback(h, eventdata, handles, varargin)
% --------------------------------------------------------------------
function varargout = pushbutton1_ButtonDownFcn(h, eventdata, handles, varargin)
% --------------------------------------------------------------------
function varargout = txtSpreadGain_Callback(h, eventdata, handles, varargin)
% --------------------------------------------------------------------
function varargout = TxtSCF_Callback(h, eventdata, handles, varargin)
% --------------------------------------------------------------------
function varargout = cmbStages_Callback(h, eventdata, handles, varargin)
```

Tester.m

```
function tester

F= 20* ones(1,11);

i=1;

%F(1)=12

for i=1:F

disp(num2str(i));

end

i=11

w(i)=2.0/(F(i)*30.0)
```

Optimum_detect.m

```
function [error_slvec,error_vec,err1_vec,err2_vec,err3_vec,epart1_vec,epart2_vec,epart3_vec,snr]
= optimum_detect(LN,N,SCFstart,stages)

% Program for maximum likelihood estimation and multiuser detection
```

```
% Originally written by Sridhar Rajagopal sridhar@rice.edu

%  http://www.ece.rice.edu/~sridhar

%% Modified by Themba Ngwenya, themba.ngwenya@ericsson.com

% University of Pretoria, Pretoria, South Africa 2003

% LN is the no of user

% N is the Spread gain

% SCFstart is the SCF start value

% stages is the no of PIC stages affected by the Function

% error_slvec is sliding correlator vector

% error_vec is matched filter

% err1_vec PIC stage 1

% err2_vac PIC stage 2

% err3_vec PIC stage 3

% epart1_vec epart1

% epart2_vec epart2

% epart3_vec epart3

% should warn if no of users exceed 32 because of K it should always be <= N

ij=1;

for ij=1:LN

   snr(ij)=(ij-1)*2;

end

K = 20*ones(1,LN);   % 1rix of 20 by 11 matrix of onesfor k = 1:length(snr)

for k = 1:LN

  % N =32; this is now a program parameter

   L = 200;sinr = 0;P =4;

rand('state',0);

randn('state',0);

% Estimation part
```


Centre For Radio & Digital Communication (CRDC)
Department of Electrical, Electronic & Computer Engineering
University of Pretoria

```
[r,codes,bits,atau,aamp] = gendata(N,K(k),sinr,snr(k),L,P);

bt=kron(bits(:,1:L),[1;0]) + kron(bits(:,2:L+1),[0;1]);

Yest = zeros(2*K(k),N);

Rbb = zeros(2*K(k),2*K(k));

Rbr = zeros(2*K(k),N);

Windowsize = L+1;

mu = 1/5000;

for (i = 1:L)

                if (i < Windowsize)

 Rbb = Rbb + (bt(:,i)*bt(:,i)');

 Rbr = Rbr + (bt(:,i)*r(:,i)');

 else

 Rbb = Rbb +  bt(i)*bt(i)' - bt(i-Windowsize)*bt(i-Windowsize)';

 Rbr = Rbr +  (bt(i)*r(i)' - bt(i-Windowsize)*r(i-Windowsize)');

 end;

                Yest = Yest - mu*(Rbb*Yest  - Rbr);

   end;

Y= Yest';

%Y = Rbr';

%size(Y);

A0 = [];A1 = [];

                for i= 1:2*K(k)

                a = mod(i,2);

                                if(a ~= 0)

                                A0 = [A0 Y(:,i)];

                                else

                                A1 = [A1 Y(:,i)];

                                 end;
```


Centre For Radio & Digital Communication (CRDC)
Department of Electrical, Electronic & Computer Engineering
University of Pretoria

```
                end;
    Rbr = Rbr'; % Comparison with sliding correlator
  A0sl = [];A1sl = [];
                for i= 1:2*K(k)
                a = mod(i,2);
                                if(a ~= 0)
                                A0sl = [A0sl (Rbr(:,i))/L];
                                else
                                A1sl = [A1sl (Rbr(:,i))/L];
                                 end;
                end;
Lmat = real(A1'*A0);
C = real(A0'*A0 + A1'*A1 -diag(diag(A0'*A0 + A1'*A1 )));
error = 0;
err1 = 0;
err2 = 0;
err3 = 0;
epart1 = 0;
epart2 = 0;
epart3 = 0;
errorsl = 0;
error1= zeros(1,3);
picerror=zeros(1,3);
Ld = round(100000);% loading preamble bits
y = zeros(K(k),Ld);
d = y;
dsl = y;
ytemp = y;
```


Centre For Radio & Digital Communication (CRDC)
Department of Electrical, Electronic & Computer Engineering
University of Pretoria

```
ysl = y;

% Detection part

rand('state',0);

randn('state',0)

[r,codes,bits,atau,aamp] = gendata(N,K(k),sinr,snr(k),Ld,P);

r = reshape(r,N,Ld);

% Matched Filter - MF, makes initial detection of transmitted bits,and the soft and hard outputs fed
to Multiuser Detection stage

for(i=1:Ld)

                    if (i==1)

      y(:,i) = real(A0'*r(:,i));

      ysl(:,i) = real(A0sl'*r(:,i));

    else

    y(:,i) = real(A1'*r(:,i-1) + A0'*r(:,i));

                    ysl(:,i) = real(A1sl'*r(:,i-1) + A0sl'*r(:,i));

  end;

    y(:,i);

    d(:,i) = sign(y(:,i));

    for(j =1:K(k))

                if( d(j,i) ~= bits(j,i))

                 error = error +1;

                 end;

    end;

     dsl(:,i) = sign(ysl(:,i));

    for(j =1:K(k))

                if( dsl(j,i) ~= bits(j,i))

                 errorsl = errorsl +1;

                 end;
```


Centre For Radio & Digital Communication (CRDC)
Department of Electrical, Electronic & Computer Engineering
University of Pretoria

```
      end;
end;
%Detection Stages for Conventional and PPPMUD
% Conventional Parallel Pipelined Multiuser Detector
% PIC - Stage 1 ,2 and 3
ij=1;
for(ij=1:3)
   for(i=1:Ld)
       if(i==1)
        ytemp(:,i) = y(:,i)  - C*d(:,i) - Lmat'*d(:,i+1);  % first row
      elseif (i ==Ld)
        ytemp(:,i) = y(:,i) - Lmat*d(:,i-1) - C*d(:,i);   % other rows
      else
      ytemp(:,i) = y(:,i) - Lmat*d(:,i-1) - C*d(:,i) - Lmat'*d(:,i+1); % last row
      end;
           d(:,i) = sign(ytemp(:,i));   % get the sign
      for(j =1:K(k))
                  if( d(j,i) ~= bits(j,i))
                   picerror(ij) = picerror(ij) +1;   % increment the bit error count
                   end;
      end;
   end;
end
 % Partial Parallel Pipelined Multiuser Detector - PPPMUD
 % The SCF multiplys the channel gain estimates before subtraction from received signal
   ij=1;
for(ij=1:3)
   for(i=1:Ld)
```


Centre For Radio & Digital Communication (CRDC)
Department of Electrical, Electronic & Computer Engineering
University of Pretoria

```
        if(i==1)

        ytemp(:,i) = y(:,i)  - SCF(snr(k),ij,SCFstart,stages)*C*d(:,i) -
SCF(snr(k),ij,SCFstart,stages)*Lmat'*d(:,i+1);  % first row

     elseif (i ==Ld)

        ytemp(:,i) = y(:,i) - SCF(snr(k),ij,SCFstart,stages)*Lmat*d(:,i-1) -
SCF(snr(k),ij,SCFstart,stages)*C*d(:,i); %second row

      else

      ytemp(:,i) = y(:,i) - SCF(snr(k),ij,SCFstart,stages)*Lmat*d(:,i-1) -
SCF(snr(k),ij,SCFstart,stages)*C*d(:,i) - ...

         SCF(snr(k),ij,SCFstart,stages)*Lmat'*d(:,i+1); % last row

      end;

           d(:,i) = sign(ytemp(:,i));

      for(j =1:K(k))

                  if( d(j,i) ~= bits(j,i))

                   error1(ij) = error1(ij) +1;

                   end;

      end;

   end;

end


       ind=K(k)*Ld;

   error_slvec(k) = errorsl/ind; % sliding correlator

   error_vec(k) = error/ind;       % error

  err1_vec(k) = picerror(1)/ind;    % error in PIC stage 1

  err2_vec(k) = picerror(2)/ind;    % error in PIC stage 2

  err3_vec(k) = picerror(3)/ind;    % error in PIC stage 3

  epart1_vec(k) = error1(1)/ind;     % error in PPPMUD Stage 1

  epart2_vec(k) = error1(2)/ind;     % error in PPPMUD stage 2

  epart3_vec(k) = error1(3)/ind;    % error in PPPMUD stage 3
```

```
end;
```

Gendata.m

```
function [y,codes,bits,tau,amplitude,dim] = gendata(n,k,sinr,snr,L,P,a)
%GENDATAGenerate Direct-sequence CDMA antenna data
%%          function [y,codes,bits,tau,amplitude,dim] = gendata(n,k,sinr,snr,L,P,a)
%           PARAMETER:
%           n = Spreading gain (i.e. # of chips/symbol)
%           k = Number of users
%           sinr = Signal-to-interference noise ratio (in dB)
%           snr = Signal-to-noise ratio (in dB)
%           L = Number of snapshots to be obtained. Each column of y
%              corresponds to 1 snapshot (i.e. "n" chips) of data.
%     P = Number of Paths
%           a = Optional parameter which defines the spreading waveforms.
%              If the parameter is undefined, it will be generated
%              internally.
%           OUTPUT:
%           y = Observed data corrupted by noise ( N*L)
%           codes = The codes of the various users, obtained from Gold code 31(N*K)
%           bits = The sequence of bits which are transmitted( K*(L+1)
%             observation length +1)
%     tau = The delays of the various bits.
%((K*P) * 1) K user's P paths
%     amplitude = The amplitude of the various bits
%( 1 * (K*P))
%     dim : The total number of orthogonal dimensions required.
%           Genseq(n,i) is used to generate the
```


Centre For Radio & Digital Communication (CRDC)
Department of Electrical, Electronic & Computer Engineering
University of Pretoria

```
%                spreading waveform for user i
%
%       Initially Written by : Kishore Kota,Chaitali Sengupta
%        Modified by : Sridhar Rajagopal
% Generate the spreading sequences
% Fill the Codes matrix with the Gold Code corresponding to the
% randomly selected user.
if (nargin == 6)
codes = zeros(n,k);
maxusers = genseq(n);
[tmp index] = sort(rand(maxusers,1));
for i=1:min([k maxusers])
  codes(:,i) = genseq(n,index(i))';
end
if (k > maxusers)
  for i=(maxusers+1):k
    codes(:,i) = sign(randn(n,1));
  end
end
end
a = kron(ones(1,P),codes);
% Make them asynchronous; Generate random delays
tau = ((n-2)/P)*rand(k,1);
for p=2:P
 tau = [tau; ((n-1)/P)*rand(k,1)+tau((p-2)*k+1:(p-2)*k+k)];
end
tau = tau+1;
% Generate the amplitudes and  MAI
```

```
mai = 10.^(-sinr*rand(1,k)/20);

mai(1) = 1.0;

amp = exp(sqrt(-1)*2*pi*rand(1,k));

amplitude=amp.*mai;

%Amplitude of Path P assumed as Amplitude/P

for p=2:P

 for i=1:k

   amplitude((p-1)*k+i) = amplitude(i)/p;

end

end

% estimate noise variance

var = sqrt(n)*10^(-snr/20);

% Generate random data(currently it is uncorrelated)

bits = round(round(rand(k,L+1)) - .5*ones(k,L+1));

x = kron(ones(P,1),bits);

% Generate random noise

noise = (randn(n,L) + sqrt(-1)*randn(n,L))*var*sqrt(0.5);

% Generate the data

 [aleft, aright] = splitAmatrix(a,tau);

%splits a into aleft and aright matrices

xx = diag([amplitude'; amplitude']) * [x(:,1:L); x(:,2:L+1)];

y =  (aleft+aright) * xx;

dim=rank(y);

y = y + noise;
```

Genseq.m

```
function [code] = genseq(n, k)

%GENSEQ    Generate a m-sequence
```

```
%               function [code] = genseq(n, k)
%               n = length of the m-sequence
%               k = index into the sequence table
%               code = a vector containing the m-sequence
%%              If the parameter "k" is missing, then the function returns with
%               the number of codes of length "n" currently implemented.
%%              NOTE: Currently the sequence table is implemented only for n=31
%%      Written by : Kishore Kota
gold=0;
if (n == 31)
  [GC]=GoldCode(31);
  gold = 1;
  seqtable = [0 1 1 1 0 0; ...
                    0 0 1 0 1 0; ...
                    1 0 1 0 0 1; ...
                    1 0 1 1 1 1; ...
                    1 1 1 0 1 1; ...
                    ]';
elseif (n == 3)
  seqtable = [1 1 1]';
elseif (n == 7)
  gold=1;
  [GC]=GoldCode(7);
  seqtable = [1 1 0 1]';
elseif (n == 15)
  seqtable = [1 1 0 0 1; ...
                    1 1 1 1 1]';
elseif (n == 63)
```


Centre For Radio & Digital Communication (CRDC)
Department of Electrical, Electronic & Computer Engineering
University of Pretoria

```
  seqtable = [1 1 0 0 0 0 1; ...
                    1 1 1 0 1 0 1; ...
                    1 1 1 0 0 1 1; ...
                    1 0 0 1 0 0 1; ...
                    1 0 1 1 0 0 0; ...
                    ]';
elseif (n == 127)
  seqtable = [1 0 0 1 0 0 0 1; ...
                    1 1 1 1 0 0 0 1; ...
                    1 0 1 1 1 0 0 1; ...
                    1 1 1 0 1 1 1 1; ...
                    1 1 1 1 1 1 0 1; ...
                    1 0 1 0 1 0 1 1; ...
                    1 1 0 0 0 0 0 1; ...
                    1 1 0 1 0 0 1 1; ...
                    1 0 1 0 0 1 1 1; ...
                    ]';
elseif (n == 255)
  seqtable = [1 0 1 1 1 0 0 0 1; ...
                        1 1 1 0 1 1 1 0 1; ...
                        1 1 0 0 1 1 1 1 1; ...
                        1 0 0 1 0 1 1 0 1; ...
                        1 0 1 1 1 1 0 1 1; ...
                        1 1 1 0 0 1 1 1 1; ...
                        1 1 0 1 0 1 0 0 1; ...
                        1 1 1 0 1 0 1 1 1; ...
                        1 1 0 0 1 0 0 0 0; ...
                        1 0 1 0 0 1 1 0 1; ...
```


Centre For Radio & Digital Communication (CRDC)
Department of Electrical, Electronic & Computer Engineering
University of Pretoria

```
                                1 1 0 1 0 0 0 1 1; ...

                                1 1 0 0 0 1 1 0 1; ...

                                1 1 0 1 1 0 0 0 1; ...

                                1 1 1 1 1 1 0 0 1; ...

                                1 1 1 1 1 0 1 0 1; ...

                                1 1 0 0 0 0 1 1 1; ...

                                1 0 0 1 1 1 0 0 1; ...

                                1 1 1 1 1 0 0 0 0; ...

                                1 1 1 0 0 0 0 0 0; ...

                                ]';

elseif (n == 511)

  seqtable = [1 0 0 0 1 0 0 0 0 1; ...

                                1 0 0 1 1 0 1 0 0 1; ...

                                1 0 0 0 1 1 1 0 0 1; ...

                                1 0 0 1 1 0 0 1 0 1; ...

                                1 1 0 0 1 0 0 0 1 1; ...

                                1 0 1 1 0 1 0 0 0 1; ...

                                1 1 1 0 1 1 1 0 0 1; ...

                                1 0 0 0 0 1 1 0 1 1; ...

                                1 1 0 1 1 0 1 1 0 1; ...

                                1 0 1 0 0 0 0 1 1 1; ...

                                1 1 1 0 1 0 0 0 0 1; ...

                                1 0 0 1 0 1 1 1 1 1; ...

                                1 1 0 0 0 1 1 1 1 1; ...

                                1 1 1 1 0 0 0 1 1 1; ...

                                1 1 0 1 0 1 1 0 1 1; ...

                                1 0 0 0 0 0 0 0 1 1; ...

                                1 1 1 1 0 1 1 0 0 1; ...
```


Centre For Radio & Digital Communication (CRDC)
Department of Electrical, Electronic & Computer Engineering
University of Pretoria

```
                                    1 0 1 1 0 0 1 1 1 1; ...

                                    1 1 1 0 1 1 1 0 1 1; ...

                                    1 1 0 1 0 0 1 1 1 1; ...

                                    1 0 1 1 1 1 1 0 0 1; ...

                                    1 0 1 0 1 0 1 1 1 1; ...

                                    1 0 1 0 1 0 0 1 0 1; ...

                                    1 0 1 1 1 1 0 1 0 1; ...

                                    1 1 0 1 0 0 0 0 0 0; ...

                                    1 1 0 1 1 1 1 1 1 1; ...

                                    1 0 0 1 0 0 1 0 1 1; ...

                                    1 0 1 0 1 0 0 0 1 1; ...

                                    1 1 1 0 1 1 0 1 0 1; ...

                                    ]';

elseif (n == 1023)

  seqtable = [ 1 0 0 1 0 0 0 0 0 0 1; ...

                                    %1 1 1 1 0 0 0 0 0 0 1; ...

                                    1 0 1 1 0 0 0 0 1 0 1; ...

                                    1 0 0 1 1 1 1 1 1 1 1; ...

                                    %1 1 1 1 0 1 0 1 0 0 1; ...

                                    %1 0 1 0 1 1 0 0 0 0 1; ...

                                    1 1 1 1 0 1 1 0 0 0 1; ...

                                    1 0 1 1 0 0 1 0 1 1 1; ...

                                    1 1 0 1 1 1 1 1 1 0 1; ...

                                    1 1 0 1 1 0 0 0 0 0 1; ...

                                    1 1 0 0 0 1 0 0 1 0 1; ...

                                    1 0 0 0 1 1 0 0 1 0 1; ...

                                    1 1 0 0 1 0 0 0 0 1 1; ...

                                    1 1 0 0 0 1 1 0 1 1 1; ...
```


Centre For Radio & Digital Communication (CRDC)
Department of Electrical, Electronic & Computer Engineering
University of Pretoria

```
1 0 1 0 0 1 1 1 1 0 1; ...

1 0 0 1 1 0 0 0 1 0 1; ...

1 1 1 1 1 1 1 0 0 1 1; ...

1 0 1 0 1 0 1 0 1 1 1; ...

1 1 1 1 0 0 0 1 1 0 1; ...

1 0 0 1 1 1 0 0 1 1 1; ...

1 1 0 0 1 0 1 1 0 1 1; ...

1 1 1 0 0 0 1 0 1 1 1; ...

1 1 0 0 1 0 0 1 1 1 1; ...

1 1 1 0 0 0 1 1 1 0 1; ...

1 1 1 0 1 0 1 1 0 0 1; ...

1 0 1 0 1 1 0 1 0 1 1; ...

1 0 1 1 0 1 0 0 0 0 1; ...

1 0 1 1 1 1 1 0 1 1 1; ...

1 0 0 1 1 1 1 0 0 1 1; ...

1 1 1 0 0 1 0 0 0 0 1; ...

1 0 1 0 1 0 0 0 0 1 1; ...

1 1 1 1 1 0 1 1 0 1 1; ...

1 0 0 1 0 0 0 1 0 1 1; ...

]';


  elseif ((log2(n)-floor(log2(n))) == 0)

    % Code is a power of 2 then use Walsh codes.....

    H = 1;gold = 1;

for i=1:ceil(log2(n))

  H = [ H H;H -H];

end;

  GC = H;
```


Centre For Radio & Digital Communication (CRDC)
Department of Electrical, Electronic & Computer Engineering
University of Pretoria

```
  %code = seqtable(1:k,:);
  %return;
else
  n
  error('Table not yet implemented the above value of N');
end
if (nargin == 1)
 if (gold == 0)
    code = length(seqtable(1,:));
    return
  else
     code = length(GC(:,1));
     return
  end
end
% If it is a gold code simply retrive it from the table and return
if (gold==1)
  code = GC(k,:);
  return
end
if ((log2(n)-floor(log2(n))) == 0)
code = GC(1:k,:);
return;
return;
end;
stages = floor(1.442695*log(n) + 1.0);
maxnum = 2^stages;
init = zeros(stages + 1,1); init(1) = 1;
```


Centre For Radio & Digital Communication (CRDC)
Department of Electrical, Electronic & Computer Engineering
University of Pretoria

```
perm = zeros(stages+1,stages+1);
for i=2:stages+1
  perm(i,i-1) = 1;
end
register = init;
sequence = seqtable(:,k);
for i=1:n
   register = perm*register;
   code(i) = register(stages+1);
   if (code(i) == 0)
      code(i) = -1;
   end
   mask = register & sequence;
   for j=1:size(mask)
      register(1) = xor(register(1), mask(j));
   end
end
```

SCF.m

```
function r=SCF(k,ki,start,stagesaffected)
% this is the Compensator mechanism function
% k is the snr based on the no of users
% ki is an integral value representing the particular PIC stage
% stagesaffected is the no of stages of the PIC algorithm affected by this function
% start is the starting value for this function
if stagesaffected <=0 or stagesaffected >3
    stagesaffected=1;
end
if ki > stagesaffected
```

```
        r=1;
        return;
end
r=start;
if  k > 2 & k <= 8
    r = r + 0.15;
elseif  k>2 & k<=12
        r = r + 0.25;
elseif   k > 2 & k <=35
        r = r + 0.35;
else
        r=r;
end
```

SplitAmatrix

```
function [aleft, aright] = splitAmatrix(a,tau)
%  function [aleft, aright] = shiftsteeringmatrix(a,tau)
% Splits the a matrix ( n* (p*k)) into aleft and aright matrices (n* 2*(p*k))
% Note  : k used below in this code is NOT the number of users; it is K*P
 [n,k] = size(a);
k = min([k max(size(tau))]);
aleft = zeros(n,2*k);
aright = zeros(n,2*k);
tau = rem(tau + n*ones(size(tau)), n);
frac = tau - floor(tau);
for i=1:k
   s = floor(tau(i));
   if (s ~= 0)
```


Centre For Radio & Digital Communication (CRDC)
Department of Electrical, Electronic & Computer Engineering
University of Pretoria

```
    aleft(1:s,i) = (1-frac(i)) * a(n-s+1:n,i);
  end
  aleft(s+1:n,k+i) = (1-frac(i)) * a(1:n-s,i);
  aright(1:s+1,i) = frac(i) * a(n-s:n,i);
  if (s ~= n-1)
    aright(s+2:n, k+i) = frac(i) * a(1:n-s-1,i);
  end
end
```


Centre For Radio & Digital Communication (CRDC)
Department of Electrical, Electronic & Computer Engineering
University of Pretoria

REFERENCES

[1] Mats Nilsson, "Third Generation Radio Access Standards," *Ericsson Review, No.3 1999,* pp. 110-121.

[2] Fakhrul Alam, "Simulation of Third Generation CDMA Systems," Masters thesis, Virgina Polytechnic Institute & State University, Blacksburg, Virgina, 1999.

[3] Sorour Falahati, Pal Frenger, Ann-Louise Johansson, Pal Orten, Tony Ottosson, Arne Svensson, "Future Improvements to Wideband CDMA", *RadioVetenskap och Kommunikation 99*, 1999. Available: http://citeseer.nj.nec.com/173524.html *or* http://www.hk-r/rvk99/Prel_program.htm

[4] Hwan-Min Kang and Sung Ho Cho, "Multiuser Detection Algorithms for CDMA," Tech Report, Communications and Signal Processing Laboratory, Hanyang University,Seoul, South Korea, 2001. Available: http://dir.pe.kr/research.html

[5] Malik, R. Dubey, V.K. McGuffin, "A Hybrid Interference Canceller For CDMA Systems in Rayleigh Fading Channels," *IEEE Vehicular Technology Conference,*2001. *VTC 2001 Spring. IEEE VTS 53rd, Volume 2, 2001*,pp(s) 1523 – 1527 vol.2

[6] Pascal G. Renucci, "Optimization of Soft Interference Cancellation in DS-CDMA Receivers," Masters thesis, Virgina Polytechnic Institute & State University, Blacksburg, Virgina,1998.

[7] S. Rajagopal, J.R. Cavallaro, "A Bit-Streaming, Pipelined Multiuser Detector For Wireless Communication Receivers," *IEEE International Symposium on Circuits and Systems (ISCAS), Sydney, Australia, May* 2001, *Volume 4,* pp. 128-131.

[8] M. K. Varanasi and B. Aazhang, "Multistage detection in asynchronous code-division multiple-access communications," *IEEE Transactions on Communications*, April 1990, vol. 38, no.4, pp. 509–519.

[9] R.M. Buehrer, B.D. Woerner, "The Application of Multiuser Detection to Cellular CDMA," PhD thesis, Virgina Polytechnic Institute & State University, Blacksburg, Virgina, June 1996.

[10] Theodore S. Rappaport, "Wireless Communications Principles and Practice", Prentice Hall PTR, 1996.

[11] R.A. Scholtz, "The Origins of Spread Spectrum Communications", *IEEE Commun Magazine, COM-30, May 1982, pp. 822 – 85.*

[12] 4G "All IP Wireless", Available: http://research.sun.com/features/4g_wireless/

[13] IMT-2000 ITU-R M[1].1457-2-200304, "Radio Interfaces Standards" ITU-R Recommendation, April 2003. Available: http://www.itu.int/rec/

[14] 3GPP TS 25.211, "Physical Channels & Mapping of Transport Channels onto Physical Channels (FDD)," 3GPP Technical Specification, Mar 2001, Ver 3.2.0.

[15] ETSI TS 125.221: Universal Mobile Telecommunications Systems (UMTS), Requirements for the UMTS Terrestrial Radio Access System (UTRA), ETSI Technical Report, UMTS 21.01, version 3.0.1, November 1997.

[16] H. Holma, A. Toskala, WCDMA for UMTS, Radio Access for Third Generation and Mobile Communications, J. Wiley & Sons, Chichester, 2000.

[17] G. Alsenmyr, J. Bergstrom, M. Hagberg, A. Milen, W. Muller, H. Palm, H. Van der Velde, P. Wallentin, F. Wallgren, "Handover between WCDMA and GSM", *Ericsson Review, 2003,* No.1, pp. 6 – 11.

[18] J. Baltersee, G. Fock, P. Schulz-Rittich, H. Meyr, "Performance Analysis of Phasor Estimation Algorithms for a FDD-UMTS RAKE Receiver", *IEEE 6th International Symposium on Spread-Spectrum Tech. & Appli., NJIT,New Jersey,USA,* September 2000

[19] J. Baltersee, G. Fock, P. Schulz-Rittich, A. Senst, "Advanced Synchronization Algorithms for RAKE Receivers for WCDMA", Joint Project Paper with Agere Systems, Aachen University of Technology, Integrated Signal Processing Systems, Aachen, Germany.

[20] K. Knoche, J. Rinas, K. Kammeyer, "MAI-Suppression with Fractional T-equalizer for CDMA", Available: www.ant.uni-bremen.de/pub/papers/krk03_paper.pdf

[21] S. Moshavi, "Multi-user detection for DS-CDMA communications,"*IEEE Communications Magazine*, Oct 1996, pp. 124–136.

[22] 3GPP2 C.S0002-C, "Physical Layer Standard for CDMA2000 Spread Spectrum Systems", 3GPP2 Technical Specification, 28 May 2002, version 1.0, Release C.: Available: www.3GPP2.org

[23] 3GPP 3G TS 25.213, "Spreading and Modulation (FDD)", 3GPP Technical Specification Group Radio Access Network, March 2000, version 3.2 (2000-03), Release 1999: Available : www.3GPP.org

[24] S. Rajagopal, "Baseband Architecture Design for Future Wireless Base-Station Receivers", Masters Thesis, Electrical and Computer Engineering, Rice University, Houston, Texas, May 2000.

[25]WCDMAChapter:Available http://www.privateline.com/3G/WCDMA

[26] Z. Xu, "Low-Complexity Multiuser Channel Estimation With Aperiodic Spreading Codes," IEEE Transactions on Signal Processing, Nov 2001, vol 49, No. 11.

[27] A. Kaul, B.D. Woerner, "An Adaptive Multistage Interference Cancellation Receiver for CDMA", Masters thesis, Virgina Polytechnic Institute & State University, Blacksburg, Virgina, Mar 1995.

[28] N.S. Correal, R.M. Buehrer, B.D. Woerner, "Improved CDMA Performance Through Bias Reduction for Parallel Interference Cancellation", *In Personal Indoor Mobile Radio Communications (PIMRC), Helsinki, Finland, Sept 1997, pp. 565- 569.*

[29] N.S. Correal, "Efficient Interference Cancellation Techniques for Advanced DS-CDMA Receivers", PhD thesis, Virgina Polytechnic Institute & State University, Blacksburg, Virgina, Jul 1999.

[30] R.M. Buehrer, A. Kaul, S. Striglis, B.D. Woerner, "Analysis of DS-CDMA Parallel Interference Cancellation with Phase & Timing Errors", *IEEE J Selected Areas Commun, Oct 1996, pp. 1522 –1535.*

[31] D. Divsalar, M. Simon,"Improved CDMA Performance Using Parallel Interference Cancellation", *Technical Report, JPL, Oct 1995.*

[32] S. Verd´u, *Multiuser Detection*, Cambridge University Press,1998.

[33] G. Xu and J. R. Cavallaro, "Real-time implementation of multistage algorithm for next generation wideband CDMA systems," in *Advanced Signal Processing Algorithms, Architectures,and Implementations IX, SPIE*, Denver, CO, July 1999, vol. 3807, pp. 62–73.

[34] N.S. Correal, R.M. Buehrer, B.D. Woerner, "A DSP-Based DS-CDMA Multiuser Receiver Employing Partial Parallel Interference Cancellation", *IEEE Journal on Selected Areas in Communications, vol 17, no. 4, April 1999.*

[35] N.B. Mandayam, "Wireless communication technologies", WINLAB, Electrical and Computer Engineering, Rutgers University, 2003,Available: . http://winwww.rutgers.edu/~narayan/Course/Wless/Lectures02/lect21.pdf

[36] S. Rajagopal, G. Xu, J.R. Cavallaro, "Implementation of Channel Estimation and Multiuser Detection Algorithms for WCDMA on Digital Signal Processors", Electrical and Computer Engineering, Rice University, Houston, Texas, 1999.