C

Grey-Box Model ID: Hybrid Genetic Optimizer

C.1 User Guide

The Matlab code of the hybrid genetic algorithm optimizer used in Grey-Box model identification is given here. The complete collection of Matlab code is also available at Mathworks file exchange website http://www.mathworks.com/matlabcentral/fileexchange/, so that user feedback can be easily incorporated in future code improvements.

Section C.2 contains the Matlab code used in Chapter 4 to compare the performance of hybrid genetic optimizer versus the performance of the conventional genetic algorithm optimizer. The main script shows the typical way the optimizer functions are called. Then the optimizer functions are presented. For both optimizers, the optimization options, the parameters being optimized and the measurement data being fitted are passed to the optimizer functions as input variables. On the other hand, the error function is "hard coded" in the optimizer functions. The error function included here computes the model fit error for the power amplifier model discussed in Section 4.3.

Section C.3 contains the Matlab code used in model parameter identification of

the power amplifier model discussed in Section 4.3. The main script are broken into three sections. The first section sets up the modeling problem by setting the model order, the input excitation, the measured data to be fitted, and the model response to be computed. After setting up the modeling problem, the second section presents a proprietary model identification solution (not discussed in the dissertation) for completeness and comparison purposes. This proprietary solution also identifies the model parameters in several stages, with the initial estimate based on heuristics. What's different from the solution presented in Chapter 4 is that the nonlinear least square optimizer is used at each stage. This proprietary solution is the precursor to the iterative identification algorithm discussed in Chapter 4. The last section contains the same main script presented in Section C.2 to invoke either the conventional genetic algorithm optimizer or the hybrid genetic optimizer. All other functions called in the script can be found in the file archive at Mathworks file exchange.

C.2 Hybrid Genetic Optimizer

C.2.1 Main Script

```
%% ######################################################## GA
Opt.Optim_Meth = 'ga';% 'lsqnonlin';%
GA.TimeLimit        = 10*3600;% 10hr
GA.FitnessLimit     = 1;
GA.StallGenLimit = 500;
    %-----------------------------------------------
GA.Run = 1;
GA.gen = 300;       GA.TolFun = 1e-5;
    GA.pop = 250; Mark = 'ko';
    %-----------------------------------------------
    GA.FileNam = [Opt.DataType 'GA' '_N' num2str(Opt.N) '-PinIdx' ...
        num2str(Opt.MaxPin_IDX) '-' Opt.Optim_MtoneFit ...
        '_' Opt.Optim_Wt num2str(Opt.Optim_WtConst) ...
        '_' Opt.Optim_Err '-' Opt.Optim_Err2 ...
       '_' Opt.Nth_Atmpt{1} '-' Opt.Nth_Atmpt{2} '-' Opt.Nth_HD2band ...
       '_' num2str(GA.Run) '-Pop' num2str(GA.pop) '-Gen' num2str(GA.gen)];
    GA.FileNam
    GA.FileSav = fullfile(PATH.MatlabSave, [GA.FileNam '.mat']);
     whos('-file', GA.FileSav);
```

```
%%=====================================================
[GA.ITER1]      = Buss_3_GA( Opt, ...
    [], GA, ...%PATH,
    Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
    Qmix, Q_cell);
[b1  InputNumCEL  aNumCEL  zxNumQMIX  Opt.y]  = ...
    Buss_XR4_XReal2zxNumQMIX_2(GA.ITER1.xReal, Opt, InputSymCEL, 1);
    Opt.y
    GA.ITER1.Opt = Opt;
    save(GA.FileSav, 'GA', '-append');
    whos('-file', GA.FileSav);% load(GA.FileSav);

%% ######################################################  Hybrid-GA
    HYB.Opt              = Opt;
      HYB.Opt.Optim_Meth = 'lsqnonlin';
    HYB.Optim_OutputFcn = '';
    HYB.MaxIter         = 100;
    HYB.TolFun          = 0.001;
HYB.PrecondBandWidth = inf;% Inf-ChoskyFactor Better than 0=ConjGrad;
    GA.HYB = HYB;
%FileNam_HYB = [ GA.FileNam ...
%    '_LSQ_Iter' num2str(HYB_Opt.MaxIter)]% '_Soln' num2str(NumBestSoln)
    NumGenVEC       = [1 10];
    NumBestSolnVEC = [1 2];% [1 5] [6 10] [1 10]
%%=====================================================
[GA.ITER1.State_HYBRID] = Buss_3_HYBRID( GA, ...
    NumBestSolnVEC, NumGenVEC, ...
    Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
    Qmix, Q_cell)
        %iter.Opt     = Opt;
        %iter.GA       = GA;
```

C.2.2 Functions

Conventional Genetic Optimizer Function

```
function [iter] = Buss_3_GA( FileNam, FileSav, iter, ...
    LSQ, FMINsear, GA, ...
    Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
    Qmix, Q_cell)
Opt = iter.Opt;

NumParam = length(Opt.xReal_0);
switch Opt.Optim_Meth
    %================================
    case{'lsqnonlin','fminsearch'}
        [LSQ.MaxIter            NumParam]
        [FMINsear.MaxIter      NumParam]
        FileSav_LSQ = FileSav;
        save('FileNam_LSQ.mat', 'FileSav_LSQ')% filename var
    %================================
    case{'ga'}
        [GA.pop     GA.gen      NumParam]
        FileSav_GA = FileSav;
        save('FileNam_GA.mat', 'FileSav_GA')% filename var
    %================================
end

if( exist(FileSav, 'file') )
%--------------------------------
whos('-file', FileSav);
FileOpt = input([FileNam ' exist [0:del |1:keep ]: '], 's');
switch FileOpt
      %---------------
        case{'0'}
            delete(FileSav);
            FileCreat   = 1;
      %---------------
     %case{'2'}
        %  FileCreat = 1;
        %   FileNam     = input(['FileNam ' Opt.Optim_Meth ': '], 's');
        %   FileSav     = fullfile(PATH.MatlabSave, [FileNam '.mat'])
      %---------------
        otherwise% case{'1'}
            FileCreat   = 0;
```

```
      %---------------
end
else%---------------------------------
FileCreat = 1;
%---------------------------------
end
if FileCreat
    switch Opt.Optim_Meth
        case{'lsqnonlin'}
            save(FileSav_LSQ, 'LSQ');
        case{'fminsearch'}
            save(FileSav_LSQ, 'FMINsear');
        case{'ga'}
            save(FileSav_GA, 'GA');
    end
end

pause
%% ================================================== ITER1
switch Opt.Optim_Meth
    %=====================================================
    case{'lsqnonlin'}
        LSQ.Optim_Options = optimset(Opt.Optim_Meth);
        LSQ.MaxFunEvals     = 400 * NumParam;
        LSQ.Optim_Options  = optimset(LSQ.Optim_Options,...
            'PrecondBandWidth',LSQ.PrecondBandWidth, ...
          'MaxIter',LSQ.MaxIter, ...% 'MaxIter',400,
            'OutputFcn',{@Plot_LSQ_Func2}, ...
            'TolFun',LSQ.TolFun,'MaxFunEvals',LSQ.MaxFunEvals);
        % 'TolFun',0.1, 'MaxFunEvals',100*
        % {'Display','final', 'notify' 'iter' 'final'}
        % 'TolX',1e-4,
RunTime_LSQ = tic;
[x, resnorm, residual, exitflag, output] = lsqnonlin( @(x) ...
   Buss_XR4_Iter( x, Opt, ...
            Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
            Qmix, Q_cell), ...
  Opt.xReal_0, [],[],LSQ.Optim_Options);
RunTime_LSQ = toc(RunTime_LSQ)
        %----------------------
        iter.xReal_0 = Opt.xReal_0;
        iter.xReal     = x;
       iter.residual = residual;
```

```
       iter.output    = output;
        iter.RunTime    = RunTime_LSQ;
            iter.Pop_LSQ = load(FileSav_LSQ, 'Pop_LSQ');
            iter.Pop_LSQ = iter.Pop_LSQ.Pop_LSQ;
    %=====================================================
    case{'fminsearch'}%'OutputFcn' problem: dont like 'FunValCheck','on'
        %NOT PREFERRED to solve sum-of-square problem => lsqnonlin
        %only minimizes over the real, f(x) must return real
        %COMPLEX VARIABLES must be split into R+jI
     %robust for discontin problem, less efficient than fminunc ord >2
        FMINsear.Optim_Options = optimset(Opt.Optim_Meth);
        FMINsear.MaxFunEvals    = 400 * NumParam;
        FMINsear.MaxIter        = FMINsear.MaxIter * NumParam / 4;
        FMINsear.Optim_Options  = optimset(FMINsear.Optim_Options, ...
          'MaxIter',FMINsear.MaxIter, ...
            'OutputFcn',@Plot_LSQ_Func2, ...%
            'TolFun',FMINsear.TolFun,'MaxFunEvals',FMINsear.MaxFunEvals);
        % fminsearch=[1e-4 1e-4]
        % {'Display','final', 'notify' 'iter' 'final'}
        % 'FunValCheck','on',
        % 'TolX',1e-4,
RunTime_LSQ = tic;
[x, fval, exitflag, output] = fminsearch( @(x) ...
    Buss_XR4_Iter( x, Opt, ...
            Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
            Qmix, Q_cell), ...
  Opt.xReal_0, FMINsear.Optim_Options);
RunTime_LSQ = toc(RunTime_LSQ)
        %----------------------
        iter.xReal_0 = Opt.xReal_0;
        iter.xReal     = x;% x of next to Last iter
       iter.output    = output;
        iter.RunTime    = RunTime_LSQ;
            iter.Pop_LSQ = load(FileSav_LSQ, 'Pop_LSQ');
            iter.Pop_LSQ = iter.Pop_LSQ.Pop_LSQ;
    %=====================================================
    case{'fminunc'}
        %NOT PREFERRED to solve sum-of-square problem => lsqnonlin
        %LargeScale:  must provide the gradient
        %Medium-scale: Quasi-Newton line search (Algorithm selected)
      %Warning: Gradient must be provided for trust-region method
        %       =>  using line-search method instead.
        %Optimization terminated: relative infinity-norm of gradient
```

```
        %       less than options.TolFun.
RunTime_LSQ = tic;
[x, fval, exitflag, output] = fminunc( @(x) ...
   Buss_XR4_Iter( x, Opt, ...
            Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
            Qmix, Q_cell), ...
  Opt.xReal_0, FMINunc.Optim_Options);
        % fminunc=[1e-6 1e-6]
        % 'PrecondBandWidth',LSQ.PrecondBandWidth, ...
        % {'Display','final', 'notify' 'iter' 'final'}
        % 'FunValCheck','on', 'Diagnostics','on'
        % 'TolX',1e-4,
RunTime_LSQ = toc(RunTime_LSQ)
        %----------------------
    %=====================================================
    case{'ga'}
        GA.Optim_Options = gaoptimset;
        GA.Optim_Options  = gaoptimset(GA.Optim_Options, ...
           'PopulationSize',GA.pop, 'Generations',GA.gen,...
            ...     % 'InitialPopulation',final_pop,
            'PlotFcns',{@Plot_GA_Func}, 'PlotInterval',1,...
            ...     % @gaplotbestf @gaplotbestindiv @gaplotdistance
            'FitnessLimit',GA.FitnessLimit, 'TimeLimit',GA.TimeLimit,...
            ...     %500 default=-inf
            'TolFun',GA.TolFun, 'StallTimeLimit',GA.TimeLimit,...
            'StallGenLimit',GA.StallGenLimit)
            % 'FitnessLimit' = (RelErr 0.1 ~ -10dB) * Wt * #pts
            % 'Display','iter' , ...
            % 'PopInitRange', [x2INTERMED-(1+j); x2INTERMED+(1+j)],
            %       [Opt.xReal_0-(1+j); Opt.xReal_0+(1+j)]
            % 'HybridFcn',{@fminunc}, @fminsearch
            % MigrationDirection
%output.XXX
%randstate       rand  state before the algorithm started.
%randnstate      randn state (randstate + randnstate to reproduce)
%generations     # generations computed.
%funccount       # evaluations of fitness function
%message     same as output argument reason (why algorithm terminated)
%maxconstraint   Maximum constraint violation, if any
    %rand('state',  output.randstate);
    %randn('state', output.randnstate);
RunTime_GA = tic;
[x, F2, reason, output, final_pop, final_score] = ga(@(x) ...
```

```
   Buss_XR4_Iter( x, Opt, ...
            Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
            Qmix, Q_cell), ...
NumParam, [],[],[],[],[],[],[],GA.Optim_Options);
RunTime_GA = toc(RunTime_GA)
        %---------------------
        iter.xReal      = x.';% GA pass ROWvec
        iter.output    = output;
        iter.final_score = final_score;
        iter.RunTime    = RunTime_GA;
            iter.Pop_GA     = load(FileSav_GA, 'Pop_GA');
            iter.Pop_GA     = iter.Pop_GA.Pop_GA;
    %=====================================================
end
```

Hybrid Genetic Optimizer Function

```
function [State_HYBRID] = Buss_3_HYBRID( GA, ...
    NumBestSolnVEC, NumGenVEC, ...
    Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
    Qmix, Q_cell)

HYB = GA.HYB;
load(GA.FileSav, 'Pop_GA');% 'StateSave','ScoreSortIDX')
    whos('-file', GA.FileSav);
%=============================================
    temp = load(GA.FileSav);
if( isfield(temp, 'State_HYBRID') )
        %--------------------------------
        State_HYBRID          = temp.State_HYBRID;
        [NumGen  NumBestSoln] = size(State_HYBRID.Score)
      NumParam              = size(Pop_GA.StateSave(1).Population, 2)
    else
        %--------------------------------
        NumGen          = size(Pop_GA.StateSave, 2)
        NumParam        = size(Pop_GA.StateSave(1).Population, 2)
        State_HYBRID.Runtime    = zeros(NumGen,NumBestSolnVEC(2));
        State_HYBRID.Score      = zeros(NumGen,NumBestSolnVEC(2));
        State_HYBRID.Population=zeros(NumGen,NumBestSolnVEC(2),NumParam);
        %State_HYBRID(ngen, nsoln).Output = output;
        %--------------------------------
        switch HYB.Optim_OutputFcn
            case{'Plot_LSQ_Func'}
```

```
        State_HYBRID.Pop0  = zeros(NumGen, NumBestSolnVEC(2), NumParam);
        State_HYBRID.Optimval0  = zeros(NumGen, NumBestSolnVEC(2));
        end
        %--------------------------------
        save(GA.FileSav, 'State_HYBRID', '-append');
    end
  clear temp
%=============================================
FileSav_LSQ = GA.FileSav;
save('FileNam_LSQ.mat', 'FileSav_LSQ')% filename var
%=============================================

[NumGenVEC]
[NumBestSolnVEC]
NumGenOpt = input(['use [0:NumGen |1:NumGenVEC]: '], 's');
    switch NumGenOpt
        case{'0'}
            NumGenVEC = [1 NumGen];
    end
    [NumBestSolnVEC     NumGenVEC       HYB.MaxIter     NumParam]
pause
%% ######################################## HYBRID
HYB.Optim_Options = optimset(HYB.Opt.Optim_Meth);
HYB.MaxFunEvals     = 400 * NumParam;
switch HYB.Optim_OutputFcn
    %=================================================
    case{'Plot_LSQ_Func'}
        HYB.Optim_Options = optimset(HYB.Optim_Options, ...
        'PrecondBandWidth', HYB.PrecondBandWidth, ...
        'MaxIter',HYB.MaxIter, ...
        'OutputFcn',{@Plot_LSQ_Func},...
        ... %{'Display','final', 'notify' 'iter' 'final'}
        'TolFun',HYB.TolFun, 'MaxFunEvals',HYB.MaxIter);
        % Default=[100*,400], [400*length(x2_0), 600]
    %=================================================
    otherwise
        HYB.Optim_Options = optimset(HYB.Optim_Options, ...
        'PrecondBandWidth',HYB.PrecondBandWidth, ...
        'MaxIter',HYB.MaxIter, ...
        ...%'OutputFcn',{@Plot_LSQ_Func},...
        ...%{'Display','final', 'notify' 'iter' 'final'}
        'TolFun',HYB.TolFun, 'MaxFunEvals',HYB.MaxIter);
        % Default=[100*,400], [400*length(x2_0), 600]
```

```
    %=================================================
end

%% ######################################## HYBRID
for ngen = NumGenVEC(1) : NumGenVEC(2)
    %[ScoreSort, ScoreSortIDX] = sort( ...
    %       Pop_GA.StateSave(ngen).Score, 'ascend');
for nsoln = NumBestSolnVEC(1) : NumBestSolnVEC(2)
x0_HYBRID   = Pop_GA.StateSave(ngen).Population( ...
            Pop_GA.ScoreSortIDX(ngen,nsoln), :);
x0_HYBRID   = x0_HYBRID.';% % GA pass ROWvec
switch HYB.Opt.Optim_Meth
%=================================================
   case{'lsqnonlin'}
%output.XXX
%iterations     Number of iterations taken
%funcCount      # function evaluations
%algorithm      Algorithm used
%cgiterations   # PCG iterations (large-scale algorithm only)
%stepsize       final step size taken (medium-scale algorithm only)
%firstorderopt Meas 1st-ord optimality (lg-scale algorithm only)
%       lg-scale bound constrain problem: inf-norm of v.*g,
%       v:as in Box Constraint, g:gradient g=JTF (see NLlsq).
RunTime_HYBRID = tic;
[x_HYBRID, resnorm,residual,exitflag,output] = lsqnonlin( @(x) ...
      Buss_XR4_Iter( x, HYB.Opt, ...
            Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
            Qmix, Q_cell), ...
            x0_HYBRID, [],[],HYB.Optim_Options);
RunTime_HYBRID = toc(RunTime_HYBRID)
    %----------------------
    %save(GA.FileSav, 'RunTime_HYBRID','residual','x_HYBRID', '-append');
    State_HYBRID(ngen, nsoln).Runtime = RunTime_HYBRID;
    State_HYBRID(ngen, nsoln).Score = sum(abs(residual));%=sum(abs(F2))
    State_HYBRID(ngen, nsoln, :).Population = x_HYBRID;%
    State_HYBRID(ngen, nsoln).Output  = output;
    %----------------------
    switch HYB.Optim_OutputFcn
        case{'Plot_LSQ_Func'}
        load(FileSav_LSQ, 'Pop_LSQ');
        State_HYBRID(ngen, nsoln, :).Pop0   = Pop_LSQ.Pop_0;
        State_HYBRID(ngen, nsoln).Optimval0 = Pop_LSQ.Optimval_0;
        State_HYBRID(ngen, nsoln, :,:).PopIter  = Pop_LSQ.Pop_Iter;
```

```
        State_HYBRID(ngen, nsoln, :).OptimvalIter = Pop_LSQ.Optimval;
    end
    %----------------------
    save(GA.FileSav, 'State_HYBRID', 'ngen','nsoln','-append');
%=====================================================
case{'fminunc'}
        %NOT PREFER to solve sum-of-square problem => lsqnonlin
        %LargeScale:  must provide the gradient
        %Medium-scale: Quasi-Newton line search (Algorithm selected)
      %Warning: Gradient must be provided for trust-region method
        %       =>  using line-search method instead.
        %Optimization terminated:
        %       relative inf-norm of gradient less than options.TolFun.
    [x_HYBRID, HYBRID_F2, exitflag, HYBRID_output] = fminunc(@(x)...
        Buss_XR4_Iter( x, HYB.Opt, ...
            Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
            Qmix, Q_cell), ...
        x_GA, HYB.Optim_Options)
%=====================================================
   case{'fminsearch'}
        %NOT PREFER to solve sum-of-square problem => lsqnonlin
        %only minimiz over real, f(x) must return real
        %COMPLEX VARIABLES must be split into R+jI
     %robust for discontin problem, less efficient than fminunc ord >2
    [x_HYBRID, HYBRID_F2, exitflag, HYBRID_output] = fminsearch(@(x)...
        Buss_XR4_Iter( x, HYB.Opt, ...
            Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
            Qmix, Q_cell), ...
        x_GA, HYB.Optim_Options)
%=====================================================
end
end
end
```

C.3 Example Usage: PA Model ID

C.3.1 Main Script: Part 1

```
[PATH, PlotOpt] = Matlab_Settings('Athlon64X2')
%-----------------------------------------------
Opt.N = 5;
    Opt.Nth_HD2band     = 'ZHD2_Flat';% 'No' 'ZHD2_NO_REIO' 'ZHD2_REIO'
Opt.ZBre_ZHD2ie     = 'ZBreioZHD2roie';% 'ZBreZHD2ie'
    Opt.DataType        = 'LSNA_RFMD';% 'VSA'
    Opt.Optim_WtConst = 10;% 100 10 2
%-----------------------------------------------
FileNam_iNL = ['Buss_iNL_' num2str(Opt.N) '_Local.mat'];%_Globe
    temp = userpath;        temp = temp(1:end-1);
FileSav_iNL = fullfile(temp, FileNam_iNL)
%-----------------------------------------------
    Nord    = 9;
    Nfreq   = 3;
    Ntone   = 2;
    Buss_Sym% (Nord, Nfreq, Ntone, FileSav_iNL)
    Q_cell = {  1 [-f1 f1]        [f1]; ...
                2   [-f3 -f2 f2 f3] [f2 f3 2*f2-f3 2*f3-f2]};
    [cSIZE c cFCT Q_cell Qmix] = Buss_2_iNL( Opt.N, Q_cell, ...
                    s_SymVEC, FileSav_iNL);
%-----------------------------------------------
Opt.NumHD = size(Qmix,2)-1
HdBand      = ones(1, Opt.NumHD+1);
switch Opt.Nth_HD2band
    case{'No'}
        HdBand(1:2) = 0 %[ENV-band HD1-band]
    otherwise
        HdBand(1:3) = 0 %[ENV-band HD1-band HD2-band]
end

%% ######################################################## H=Zmat*(I=iNL)
syms Z% Zeq = sym('Z(s)');      % dummy abstr func
z_SymVEC = [Z];
syms V
Vvec = [V];
syms I
Ivec = [I];
%--------------------- Lin-Ckt-Source
Ymat = [ 1/Z ]
```

```
%--------------------- Lin-Txfr(s) for I{1}~=[1], control HiOrd iNL
ILin = [1];
Zmat        = factor( Ymat \ ILin );
%--------------------- Lin-Resp
VLin = subs(Zmat, s, s_SymVEC(1));% Zmat .* ILin


%% ########################################################## Meas Data
Opt.Atten_dB = 20;
switch Opt.DataType
    %========================= dataset VSA
    case{'VSA'}
        Opt.MaxPin_IDX  = 5;% 5~1.5dB 6~3.5dB
        [Q_RespMTONE VSA] = VSA_QRespMtoneLoad_Func( Opt, ...
    PATH, 'SD513301', ...
    'SD513301_2000Mhz_-28dBm12dBmAtt0_100hz_401pts10ms_LSNA.s2p', ...
    'SD513301_VSA_Arb2_Test.mat', ...
    PlotOpt.df_dBRef, Opt.Atten_dB);
    %========================= dataset LSNA
    case{'LSNA_RFMD'}
        Opt.MaxPin_IDX = 12;% 12~1dB 15~2.1dB
        [Q_RespMTONE LSNA] = VSA_QRespMtoneLoad_Func( Opt, ...
    PATH, 'SD513301', ...
    'SD513301_2000Mhz_-28dBm12dBmAtt0_100hz_401pts10ms_LSNA.s2p', ...
    'SD513301_LSNA_SWP_Corr.mat', ...
    PlotOpt.df_dBRef, Opt.Atten_dB);
    %=========================
end

%% ########################################################## MTONE
[   Qmix zSymQMIX VvecQMIX    zSymQMIX_ENV zSymCEL ...
fQMIX fxQMIX   fxVEC       fCEL] = Buss_zvQMIX_Func( ...
    z_SymVEC, Vvec, ...
    HdBand, Q_RespMTONE, ...
   Qmix, Q_cell, ...
    0);% (, Opt_Disp)
%============================================
aSymCEL         = SYMvec2SYMcel(a_SymVEC(1:Opt.N));% [ aSymCEL{:} ]
%---------------------
InputCharVEC    = 'xy';
InputSymVEC     = CHARvec2SYMvec(InputCharVEC);
InputSymCEL     = {InputSymVEC};% SYMvec2SYMcel(InputSymVEC);
```

```
Opt_Disp = 0;
for nQ = 1 : size(Q_cell,1)%nQ = 2;
Q    = Q_cell{nQ,1};
  fVEC = Q_cell{nQ,2};
        f_SymVEC = fVEC(length(fVEC)/2+1 : end);
    if(Opt_Disp)
        disp(['%################################### nQ=' int2str(nQ)])
    end
for nRESP = 1 : length(Q_cell{nQ,3})%nRESP = 3
   fRESP   = Q_cell{nQ,3}(nRESP);
    Fmix    = Q_cell{nQ,4}{1,nRESP};
    Fidx    = Q_cell{nQ,4}{2,nRESP};
    cFidx   = Q_cell{nQ,4}{3,nRESP};
    if(Opt_Disp)
        disp([ '%########################### nRESP=' num2str([nRESP]) ])
        %fRESP   = Q_cell{nQ,3}(nRESP)
        %Fmix
    end
%=============================================
tic
switch Opt.DataType
    %-------------------------
    case{'LSNA_RFMD', 'VSA'}% Core1: 67sec MOST TIME CONSUME
    [Hv     Hv_SymCHAR Hv_SymCEL Hv_SymCEL_IDX] = Buss_HvCell4_Func( ...
        Qmix, zSymQMIX, Zmat, ...
        z_SymVEC, a_SymVEC, InputSymVEC, f_SymVEC, ...% VvecQMIX,
     zSymCEL, aSymCEL, InputSymCEL, ...
        Opt.N, cSIZE, c, cFCT, ...
        Fmix, Fidx, cFidx, ...
        0);
    Q_cell{nQ,4}{4,nRESP}=Hv;           %{n}[nV,1]
    Q_cell{nQ,4}{5,nRESP}=Hv_SymCHAR;   %{nV,n}[Char]
    Q_cell{nQ,4}{6,nRESP}=Hv_SymCEL;    %{nV,n}{nSYM}
    Q_cell{nQ,4}{7,nRESP}=Hv_SymCEL_IDX;%{nV,n}{nCEL [nSYM;IDX nSYM;IDX]}
    Q_cell{nQ,4}{8,nRESP}={};          % {nV,n} NumFreqPts
    %-------------------------
    case{'CDMA'}
    %-------------------------
end
toc
%=============================================
end
end
```

C.3.2 Main Script: Part 2

```
%% ######################################################## InitGuess
Opt.Third_Atmpt = 21;
Opt.Third_nPwr = 1;
    figVEC = [  600 601     0   603 ... %yModel (EVEN,ODD,RI,dB)
                610 611         ...    %ZB    (RI,MA)
                0   621        ...     %Z_Im3 (RI,dB)
                630 631     632 633];   %Z_HD2 (EVEN,ODD,RI,dB)
switch Opt.DataType
    %------------------------ dataset VSA
    case{'VSA'}
        [Opt.x0     Opt.x] = Buss_XR4_N3Estim_NLSQ_3b( Opt, ...
            VSA, fQMIX, fxQMIX, ...% Q_RespMTONE,
            Qmix, Q_cell, ...
            figVEC, PlotOpt, 1);
    %------------------------ dataset LSNA
    case{'LSNA_RFMD'}
        [Opt.x0     Opt.x] = Buss_XR4_N3Estim_NLSQ_3b( Opt, ...
            LSNA, fQMIX, fxQMIX, ...% Q_RespMTONE,
            Qmix, Q_cell, ...
            figVEC, PlotOpt, 1);
%-------------------------
end
    % Opt = rmfield(Opt, 'x_');
    Opt.x0
    Opt.x% close all
%=============================================
n_start = 5;      n_step = 2;     numpts = 4;
    Opt.PS.ordVEC = [n_start : n_step : (n_start+(numpts-1)*n_step)];
    Opt.PS.ordVEC
[Opt.PS.a3 Opt.PS.b3] = PolyFit_AMPM2( Q_RespMTONE{1,1}{1}, ...
    Vp02dBm_dB( Q_RespMTONE{1,2}{2} ), ...
    Opt.PS.ordVEC, [1:7], 'AMPM_dBm', 'Odd', 1, ...
    '.', PlotOpt);
Opt.PS.a3( 1:2, 2:8 )%DEBUG

%% ######################################################
Core  = '1';% 1 2 8 Grendel
switch Opt.Nth_HD2band
    case{'No'}
        FileNam_Optim = ['matlab_core' Core '_N' num2str(Opt.N) ...
            '_HD2' Opt.Nth_HD2band]
    otherwise
```

```
        FileNam_Optim = ['matlab_core' Core '_N' num2str(Opt.N) ...
            '_HD2band']
end
    FileSav_Optim = fullfile(PATH.MatlabSave, [FileNam_Optim '.mat'])
    whos('-file', FileSav_Optim);
    % save( FileSav_Optim )
    % load( FileSav_Optim )

%------------------------------------------------- clear LSQ
    LSQ.MaxIter         = 100;
LSQ.PrecondBandWidth = inf;% Inf-ChoskyFactor Better than 0=ConjGrad;
    FMINsear.MaxIter   = 100;
%-------------------------------------------------
    Opt.Qstart     = 2;
    Opt.Optim_Wt    = 'WtConst';% 'WtConst' 'WtPin' 'WtPinDf'
        % 'WtConst':    [10 10 1 1] Better Match S21
        % 'WtPin':      Reduce Accur at HI-PWR
    Opt.Optim_Err = 'DiffRel_RI';% 'DiffRel' 'Diff' 'RatLog_RI'
    Opt.Optim_Err2 = 'RI';% 'RI'='RIAbs' 'RIAbsSqr'=worst 'Abs'='AbsSqr'
        %Opt.Optim_Err2 = 'Abs';        46.7606 +89.8401i
        %Opt.Optim_Err2 = 'AbsSqr';     46.7606 +89.8401i
        %Opt.Optim_Err2 = 'RIAbsSqr';   117.13  +222.48i
    Opt.Optim_MtoneFit  = 'Pout';% 'S21H3';%
    %-------------------------
    Opt.Nth_Atmpt = {'a1Magb1'    'g2Mag'   'Zf'    'Real'};

%%######################################################## ITER1
    LSQ.TolFun        = 0.001;% 1e-6;%
    FMINsear.TolFun    = 1;
    %-------------------------
Opt.Nth_HD2band     = 'ZHD2_Flat';% whats updated
Opt.ZBre_ZHD2ie     = 'ZBreZHD2ie';%
    Opt.Optim_Meth      = 'lsqnonlin';% 'ga';% 'fminsearch';%
    %-------------------------
    FileNam_LocSear     = [Opt.DataType 'LSQ' '_N' num2str(Opt.N) ...
        '-PinIdx' num2str(Opt.MaxPin_IDX) '-' Opt.Optim_MtoneFit ...
        '_' Opt.Optim_Wt num2str(Opt.Optim_WtConst)  ...
        '_' Opt.Optim_Err '-' Opt.Optim_Err2 ...
        '_' cell2mat(Opt.Nth_Atmpt) '-' Opt.Nth_HD2band]
    FileSav_LocSear         = fullfile(PATH.MatlabSave, ...
        [FileNam_LocSear '.mat']);
        whos('-file', FileSav_LocSear);
    %-------------------------
```

```
Opt.x   = rmfield( Opt.x, 'gN');
Opt.x   = rmfield( Opt.x, 'ZHDN');
%------------------------
    ITER1.Opt       = Opt;
[ITER1.Opt]     = Buss_XR4_Param2XReal_3b( ITER1.Opt, ...
        ITER1.Opt.x, ITER1.Opt);
[b1     InputNumCEL     aNumCEL     zxNumQMIX   ITER1.Opt.y0] = ...
    Buss_XR4_XReal2zxNumQMIX_3b( ITER1.Opt.xReal_0, ...
            ITER1.Opt, InputSymCEL, 0);
        %ITER1.Opt.x
        %ITER1.Opt.y0
    [Q_RespMTONE Q_cell]    = Buss_XR4_Init( ITER1.Opt.xReal_0, ...
        ITER1.Opt, Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
        Qmix, Q_cell, 1);
    [F2_0           Q_RespMTONE] = Buss_XR4_Iter( ITER1.Opt.xReal_0, ...
        ITER1.Opt, Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
        Qmix, Q_cell, 0);
%==========================================
[ITER1] = Buss_3_GA( FileNam_LocSear, FileSav_LocSear, ITER1, ...
    LSQ, FMINsear, [], ...
    Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
    Qmix, Q_cell);
[b1     InputNumCEL     aNumCEL     zxNumQMIX   ITER1.Opt.y]  = ...
    Buss_XR4_XReal2zxNumQMIX_3b( ITER1.xReal, ITER1.Opt, InputSymCEL,0);
        ITER1.Opt.y
save(FileSav_LocSear, 'ITER1', 'FileNam_LocSear', ...
    'FileSav_LocSear', '-append');
    %f1f2 dBErr:    40  / 10 /(10*12)   = 0.0333
    %im3 dBErr:     100 / 1  /(10*12)   = 0.8333
    %VSA    PinIdx=5    Wt=100 Err = [130 126] RunTime_LSQ = [305 518]
    %LSNA   PinIdx=15   Wt=100 Err = [366 ]    RunTime_LSQ = [232 ]
    %LSNA   PinIdx=12   Wt=10  Err = [23]      RunTime_LSQ = [550 ]

%%######################################################## ITER2
    LSQ.TolFun        = 0.001;
    FMINsear.TolFun    = 0.001;
    %------------------------
Opt2 = ITER1.Opt;
    Opt2.Nth_HD2band = 'ZHD2_REIO';% whats updated
Opt2.ZBre_ZHD2ie    = 'ZBreioZHD2roie';
    Opt2.Optim_Meth   = 'lsqnonlin';% 'ga';% 'fminsearch';%
%------------------------
    ITER2.Opt           = Opt2;
```

```
[ITER2.Opt]         = Buss_XR4_Param2XReal_3b( ITER2.Opt, ...
                            ITER1.Opt.y, ITER1.Opt);
[b1     InputNumCEL     aNumCEL     zxNumQMIX   ITER2.Opt.y0] = ...
    Buss_XR4_XReal2zxNumQMIX_3b( ITER2.Opt.xReal_0,...
        ITER2.Opt, InputSymCEL, 0);
        %ITER2.Opt.x
        %ITER2.Opt.y0
    [Q_RespMTONE Q_cell]         = Buss_XR4_Init( ITER2.Opt.xReal_0,...
        ITER2.Opt, ...
        Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
        Qmix, Q_cell, 1);
    [F2_0           Q_RespMTONE] = Buss_XR4_Iter( ITER2.Opt.xReal_0,...
        ITER2.Opt, ...
        Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
        Qmix, Q_cell, 0);
%===========================================
[ITER2] = Buss_3_GA( FileNam_LocSear, FileSav_LocSear, ITER2, ...
    LSQ, FMINsear, [], ...
    Q_RespMTONE, fQMIX, fxQMIX, fxVEC, InputSymCEL, ...
    Qmix, Q_cell);
[b1     InputNumCEL     aNumCEL     zxNumQMIX   ITER2.Opt.y]  = ...
    Buss_XR4_XReal2zxNumQMIX_3b( ITER2.xReal, ITER2.Opt, InputSymCEL, 0);
        ITER2.Opt.y
save(FileSav_LocSear, 'ITER2', '-append');
```