■2011/04/04

これは、暫定版です。
随時バージョンアップしますので、ときどきダウンロードしてチェックしてみて下さい。

佐藤史明

■2011/04/18

多少 ver. up しました。前 ver.（上記の４月４日の版）には、sweep_check.sce が zip ファイルに含まれていなかったかもしれません。失礼いたしました。
fft_check.sce の補足については、次回とさせて下さい。

佐藤史明

■2011/04/19

これで final ver.となります。
お問い合わせが多い事項があった場合には、更に ver. up する可能性もあります。
ひとまずこれにて失礼いたします。

佐藤史明

［1］au3 について

・図 1-1 と図 1-2 は au3 の画面です。原稿の中で表現しづらかった箇所等を吹き出しで示します。

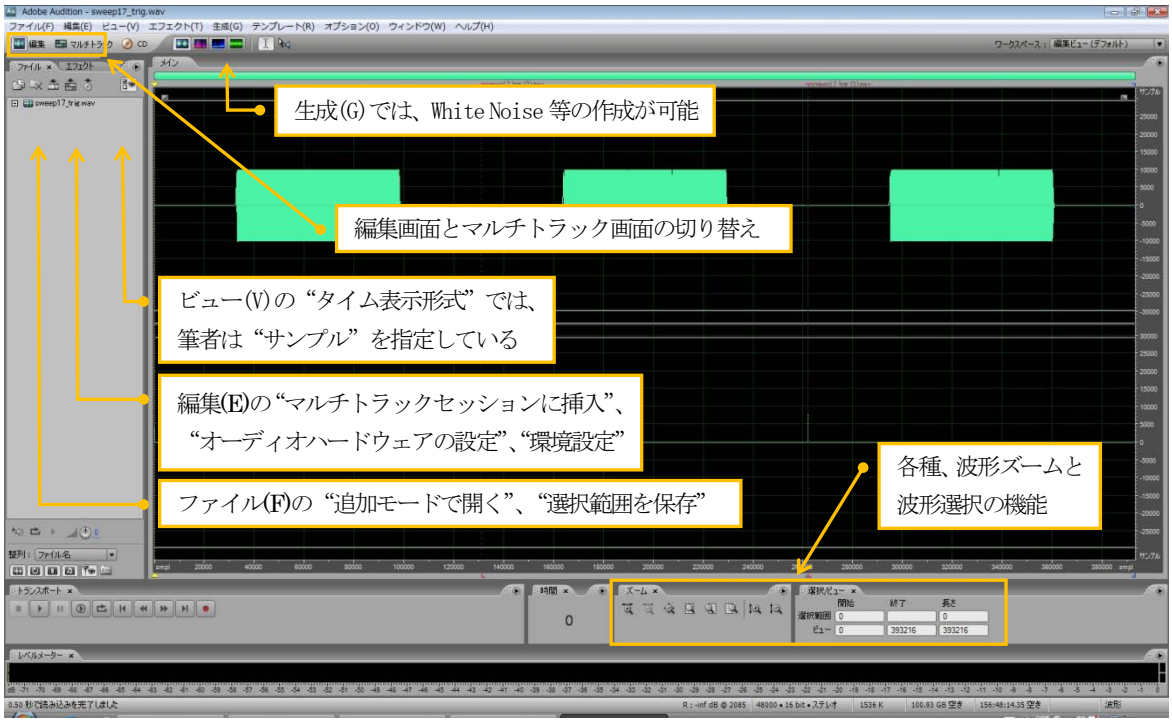

図 1-1　au3 の波形編集画面

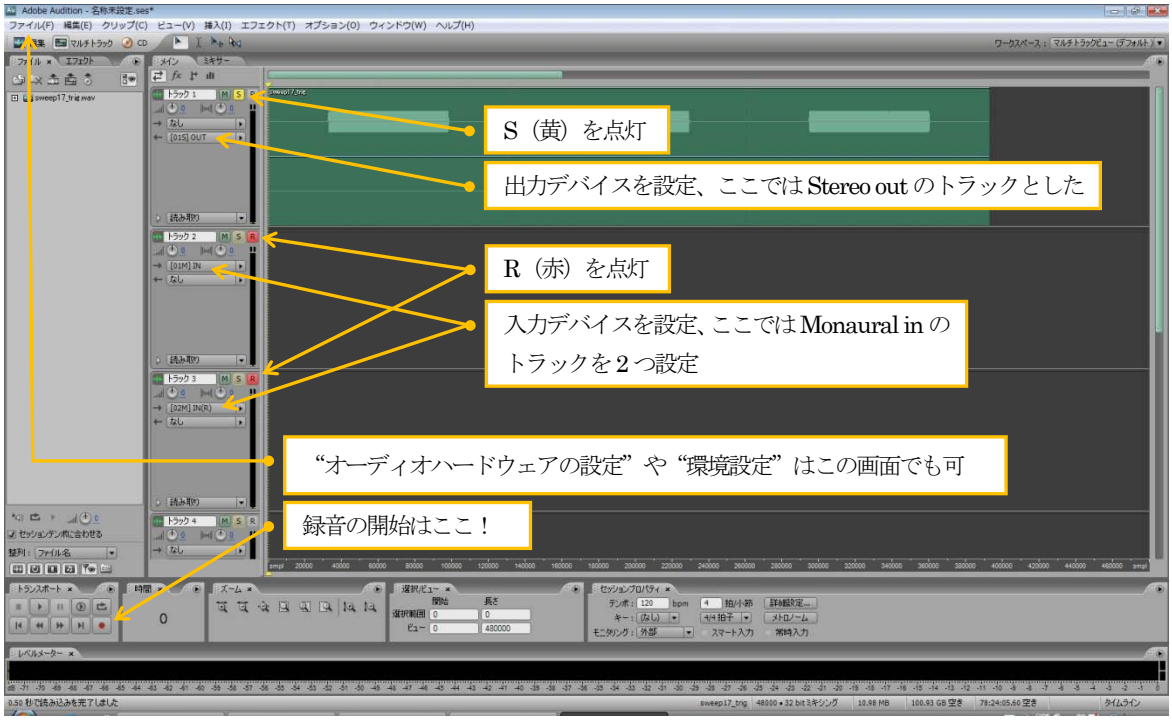

図 1-2　au3 のマルチトラック（同時録音再生）画面

・図 1-3～図 1-5 は、筆者が使用した au3 の環境設定やオーディオハードウェアの設定状態の主な箇所です。本文でも記しましたが、“環境設定”の“マルチトラック”の設定画面で、ステレオパンニングモードのデフォルト設定を“対数による L/R カット”にして下さい。そうしないと（“-3dB センター”にすると）、出力が-3dB となり、デジタル入出力の確認（本文 4.3(2)の確認）ができません。

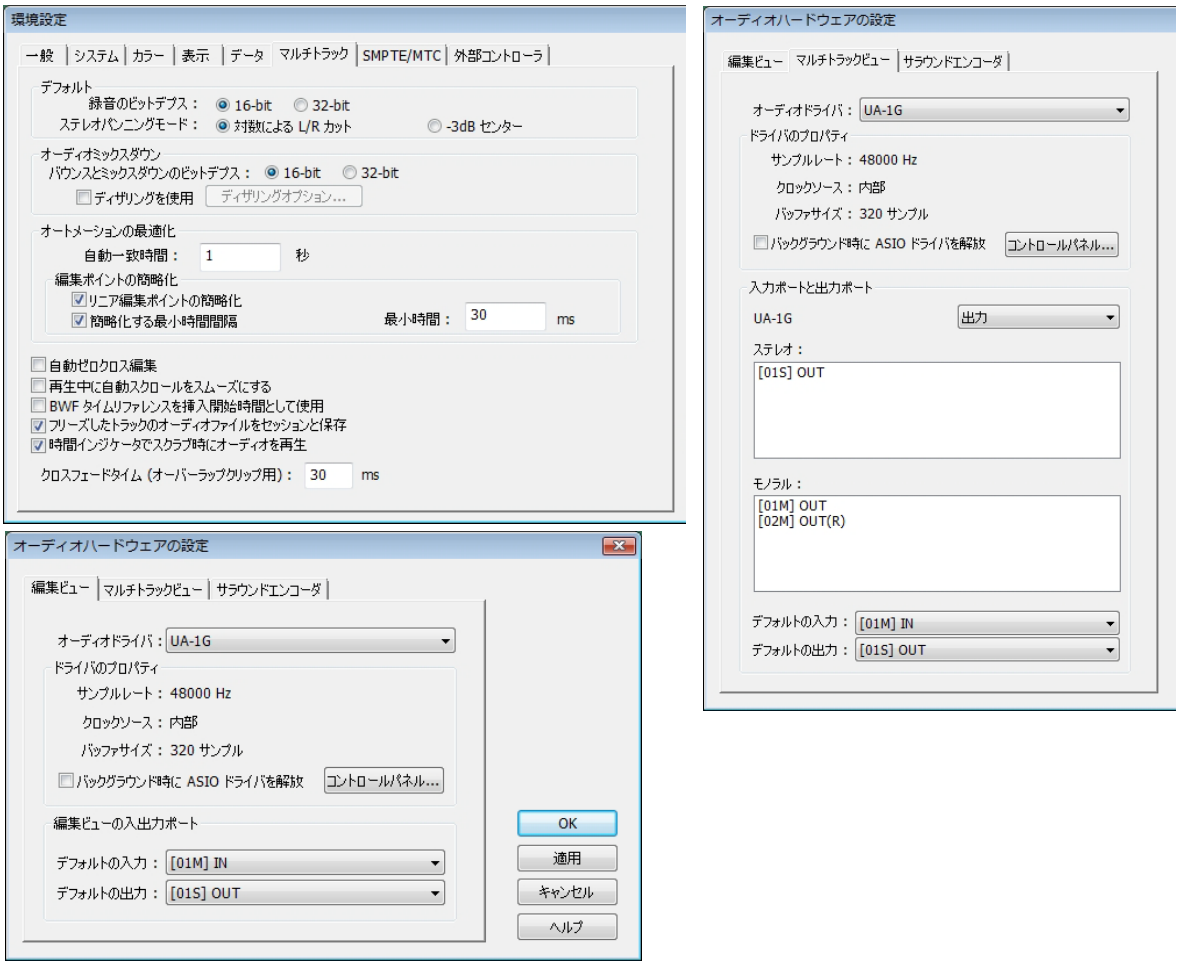

図 1-3～図 1-5　筆者の au3 の環境設定やオーディオハードウェアの設定状態

［2］Scilab について

・原稿の表 1 でも記載したとおり、Scilab 5.2.2 を http://www.scilab.org/ からダウンロードし、日本語でフルインストールしました。
・ダウンロードした Scilab のプログラムファイル（以下 sce ファイル）は、1 つのディレクトリにまとめた方がいいと思います。
・どれかの sce ファイルをクリックすると、図 2-1 のような Scilab のコンソール画面と図 2-2 のような Scilab のテキストエディタが開きます。
・テキストエディタが開かない場合は、コンソールの“ファイル(F)”の“ファイルを開く(O)”から所望のファイルを開いて下さい。“アプリケーション(A)”で“エディタ(E)”を起動してからでも開けます。
・開いた sce ファイルの実行は、Scilab テキストエディタの“実行”の“Scilab へロード”をクリックします。もちろん“Scilab へファイル実行”をクリックしても構いません。

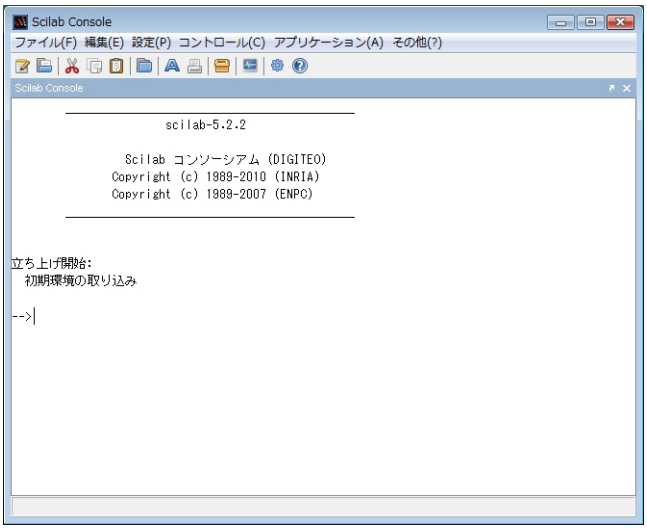

図 2-1　Scilab のコンソール画面

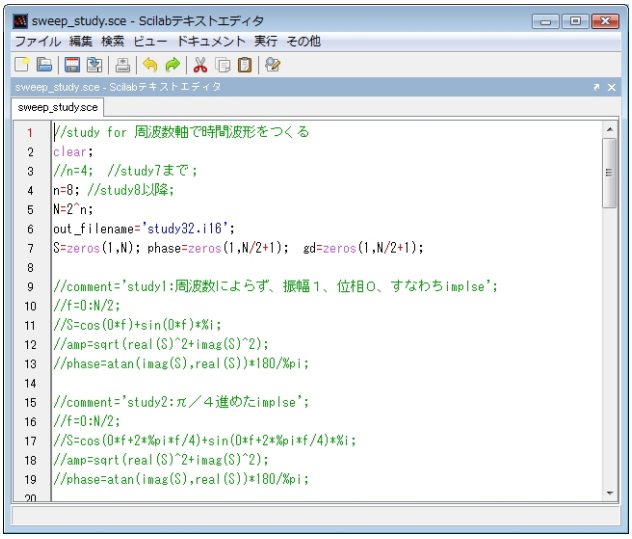

図 2-2　Scilab のテキストエディタ画面

［3］各プログラム解説

今後のプログラミングの参考になればと思い、できるだけいろいろな表現で作ってみました。
コメント行も参考にして下さい

■ mk_impulse_wav.sce

・1 秒ごとに、左チャンネルはプラスのパルス、右チャンネルはマイナスのパルスとしています。
・11 行目の様に、要所要所で配列（マトリックス）の大きさを確かめることをお勧めします。
この確認は、コンソール画面でもできます。実行後に、コンソール画面で length(y)とタイプしてみて下さい。また、y とタイプしたり、y(1,:)とタイプしたり、y(2,:)とタイプしてみて下さい。
Scilab（インターラプト型言語）の特徴の一つですね。
・ただし、インターラプト型言語は、それまでに使用した変数等の値が残るので、この点が注意事項の一つとなります。ですので今回は、全てのプログラムの 1 行目に clear;と記述しています。

■ chk_waveform.sce

・簡単な wav ファイルの波形確認プログラムです。どんどん改良してみて下さい。
・プログラムを実行したら、コンソールで問われる st_point = に 1 や 48000 や 48001 等、入力してみて下さい。

■ mk_impulse_step_wav.sce

・左チャンネルは 1 秒ごとにプラスのパルスが立ちます。右チャンネルは、冒頭の 100 サンプルについて、100,200,300,・・・,10000 と、100 ずつサンプル値が増えていきます。

■ check_mono_wav.sce

・原稿の図 2 を作成したプログラムです。
・配列 y（波形：時間軸）をフーリエ変換した結果が、配列 Y（周波数軸）に格納されます。
・モノラルの wav ファイルを読み込み、波形ならびに周波数特性を表示しています。
波形表示については、冒頭の 512 サンプルを表示しています。
周波数特性については、読み込んだ全長さ(8192 サンプル)についてフーリエ変換しています。
・5 行目や 6 行目の、y=zeros(1,N)や Y=zeros(1,N)は、記述しなくてもプログラムは走りますが、N が大きくなると実行速度が低下します。明示する習慣をつけることを強く推奨します。
・プログラムを実行したら、コンソール画面で、y とか real(y)とか imag(y)とか、さらには Y とか real(Y)とか imag(Y)とかタイプしてみて下さい。
さらにさらに、7 行目のコメント行のマークをはずして同様なことをしてみて下さい。
・11 行目の plot2d の中の y(1,t)は、そもそも 1 次元配列なので、y(t)でも構いません。

■ sweep_study.sce

・原稿中の study1～study10 の表現に関わるプログラムが、sweep_study1.sce～sweep_study10.sce にそれぞれ相当します。また、sweep_study1.sce～sweep_study10.sce をまとめたのが、sweep_study.sce となります。
・ここでは、Group Dealy 擬きが描画されます。
二つのベクトルのなす角を計算しているだけなので、遅れたのか？、それとも進んだのか？、については Phase の図と合わせて考える必要があるということです。
・その他、図 3-1 と図 3-2 で補足解説します。

sweep_study.sce - Scilabテキストエディタ

ファイル 編集 検索 ビュー ドキュメント 実行 その他

*sweep_study.sce

```
1  //study for 周波数軸で時間波形をつくる
2  clear;
3  n=4;  //study7まで;
4  //n=8; //study8以降;
5  N=2^n;
6  out_filename='study32.i16';
7  S=zeros(1,N); phase=zeros(1,N/2+1);  gd=zeros(1,N/2+1);
8
9  //comment='study1:周波数によらず、振幅１、位相０、すなわちimplse';
10 //f=0:N/2;
11 //S=cos(0*f)+sin(0*f)*%i;
12 //amp=sqrt(real(S)^2+imag(S)^2);
13 //phase=atan(imag(S),real(S))*180/%pi;
14
15 comment='study2:π／４進めたimplse';
16 f=0:N/2;
17 S=cos(0*f+2*%pi*f/4)+sin(0*f+2*%pi*f/4)*%i;
18 amp=sqrt(real(S)^2+imag(S)^2);
19 phase=atan(imag(S),real(S))*180/%pi;
20
21 //comment='study3:π３／４遅らせたimplse';
22 //f=0:N/2;
23 //S=cos(0*f+2*%pi*f*3/4)-sin(0*f+2*%pi*f*3/4)*%i;
24 //amp=sqrt(real(S)^2+imag(S)^2);
25 //phase=atan(imag(S),real(S))*180/%pi;
```

図 3-1　sweep_study.sce の冒頭の部分

- 5 行で 1 グループ（一つの study）です。
- グループ単位でコメントマーク（//）をつけたりはずしたりして下さい。
- 左図では、study2 が実行されます。
- study1～study7 を実行する時は、n=4 が適当です。(4 行目をコメント行にする)
- study8～study10 を実行する時は、n=8 が適当です。(3 行目をコメント行にする)

- 11 行目の sin や cos の括弧内は妙な記述ですが、こうしないと f で Loop しないようです。

sweep_study.sce - Scilabテキストエディタ

ファイル 編集 検索 ビュー ドキュメント 実行 その他

*sweep_study.sce

```
71 scf();
72 subplot(3,2,1); plot(f,real(S)); title('Real part');
73 subplot(3,2,2); plot(f,imag(S)); title('Imag part');
74 subplot(3,2,3); plot(f,amp); title('Amp');
75 subplot(3,2,4); plot(f,phase); title('Phase');
76
77 for k=N/2+1:N-1;
78     kk=k+1;
79     S(kk)=conj(S(N-kk+2));
80 end
81 s=ifft(S);
82
83 gd(1)=0;
84 for x=2:N/2+1;
85   A=[real(S(x-1)),imag(S(x-1))];
86   B=[real(S(x)),imag(S(x))];
87   gd(x-1)=acos(A*B')*180/%pi;
88 end
89
90 x=0:N-1;
91 subplot(3,2,5); plot(x,s); title(comment);
92 x=0:N/2;
93 subplot(3,2,6); plot(x,gd); title('Group Delay MODOKI');
94 //Fid=mopen(out_filename,'wb');  mput(s,'f',Fid);  mclose(Fid);
```

図 3-2　sweep_study.sce の終わりの部分

83 行目～88 行目

- 前述のとおり、Group Delay 擬きです。
- 内積（A*B' は成分同士のかけ算）使ってベクトル A とベクトル B のなす角を求めています。
  ここでは振幅が 1 なので、
  ベクトル A も B も単位ベクトルです。

■ sweep_check.sce, hanten.sce, c_conv.sce, l_conv.sce

・この 4 つは、原稿の 6.2 の学習用です。n をそろえて下さい。デフォルトは 13 としています。
・プログラム中の諸々のコメントもよく読み込んで下さい。
・sweep_check.sce の 4 行目と 5 行目は排他使用で、4 行目の使用で up 信号が、5 行目の使用で down 信号が作成できます。
・hannten.sce は、sweep_chkeck.sec で作った信号を時間反転させます。
・c_conv.sce は円状たたみ込み、l_conv.sce は直線たたみ込みのプログラムです。
・ここでは、入出力のファイル形式をバイナリー形式、具体的には 2 byte (16 bit)の integer (整数型) としています。au3 で波形を確認する際、図 3-3 を参照して読み込んで下さい。

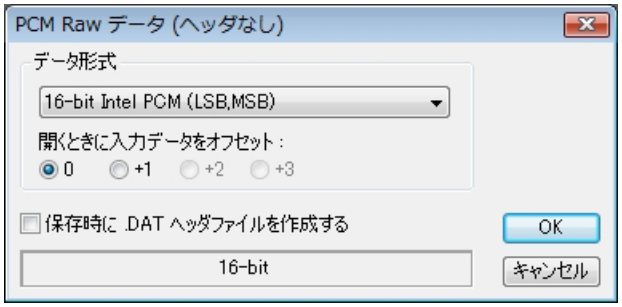

図 3-3　au3 での PCM ファイルの読み込み

■mk_sweep.sce, bin2wav_trig.sce

・原稿の 6.3 と 6.4 の学習用です。
・mk_sweep.sce は、sweep 信号と逆 sweep 信号が作成できます。(バイナリー形式)
・bin2wav_trig.sce は、上記で作成したバイナリー形式の sweep 信号を左チャンネルに、右チャンネルにはトリガー信号をつけたステレオ信号が wav 形式で作成できます。
・デフォルトで n は 15 となっています。原稿の 6.4(a)と 6.4(c)では n=15 としました。
原稿の 6.4(b)の際には n=17 としました。

■get_ir_by_c_conv.sce

・原稿の図 2 の S/N がよくなったもの（周波数特性がよりシャープに表現される）が作画されます。

■check_sp_cord.sce

・原稿の図 4 を作成したプログラムです。

■get_ir_by_l_conv.sce

・原稿の図 6 を作成したプログラムです。

■fft_check1.sce

・インパルスを色々な FFT 長さで FFT してみます。
・FFT 長さを 2 倍にすると、トータルのエネルギーが ＋3dB になってしまいます。
・この方式では各ラインスペクトルの値は変わらない。（筆者の好みではない）
・single event の波形なので、使用する FFT の長さによらず同じ値になってほしい。

以下、上記と同じ内容になりますが、

■fft_check2.sce

・check1 と同様、single event の波形として、mk_sweep.sce で作成した sweep15_i16.pcm を使ってみます。
・2 の 15 乗の FFT に対し、末尾に 0 を付け足して 2 の 16 乗で FFT すると、同じ方式なので当たり前ですが、信号のエネルギーが +3 dB になってしまいますね。

■fft_check3.sce

・au3 で作成可能ですが、2 の 15 乗（32768 words）や 2 の 16 乗（65536 words）のデータをつくってチェックしてもよいでしょう。
・条件 1 と条件 2 の比較は上述と同じです。
・条件 3 は条件 1 に対して +6 dB になってしまいます。
信号の長さが倍になっただけなので +3 dB になってほしいですね。
（信号が完全に重なって 2 倍になったわけではない。）

■fft_check4.sce

・同様に純音でも確認してみましょう。
・ここでは、fs を sampling 周波数として、長さ N1 と N2 の 1 Hz (f)の波形を作っています。
・そしてそれぞれ、N1 の長さの FFT と N2 の長さの FFT で FFT しています。
・グラフだと dB の値が読みにくいですが、コンソール画面には、約 42 dB の値と約 48 dB の値が表示されます（すなわち 6 dB の差）。
・配列 X1 や X2 の最初（ X1(1)と X2(1) ）は DC 成分を表しますので、1 Hz のレベルを表示させるのに、X1(2)と X2(3)を使っているところにも注意して下さい。
もちろん、X1(N1/fs+1)や X2(N2/fs+1)と記述した方がベターかもしれません。

■fft_check5.sce

・以上の確認から Scilab の FFT の場合、使用する FFT の長さによらずその信号のエネルギー（時間信号の場合は 2 乗積分値）と整合性のとれた値を得るためには（相対値ではあるが）、FFT 後に FFT 長さの√で除すとよいでしょう。

以　上