流体計測特論（ウェーブレット解析）

機械工学科 飯田 明由

要　約

スペクトル解析はランダムデータを解析する上で有効なツールであるが，時間に関する情報（信号がいつ発生したか）については情報が得られない．乱流のような非定常，間欠的な現象では信号の時間的な変化も重要な情報である．ウェーブレット解析はフーリエ変換から発展した新しい手法で，局在する信号の検出に威力を発揮する．この章では連続ウェーブレットを用いた非定常信号解析手法について解説する．

1. スペクトル解析の問題点

フーリエ変換によって得られるパワースペクトルは，ランダム信号が持っている各周波数毎のエネルギー強度を表すことができるため，ランダム信号解析，特に乱流解析において重要な解析手法となっている．しかし，フーリエ変換はその定義上，無限に続く周期信号（三角関数）を用いて信号を分解するため，各周波数の正弦波が時間軸上のどの時点で発生したかは問題にしない．

信号Aは振幅の小さな正弦波 $a_1\sin(\omega_1 t)$が無限に続く信号である．信号Bは局所的に大きな振幅の制限波 $a_2\sin(\omega_1 t)$が存在するような信号とする．$a_1<a_2$ のとき，二つの信号のパワースペクトルには同じ周波数 $f=2\pi\omega_1$ にピークを持つ信号が得られる．もちろん，信号Bは局所的に信号が変化するのでパワースペクトルにはその影響による高調波成分が含まれるが，周波数 f に関しての情報はほぼ同等な信号がえらる．仮に信号Aに雑音が含まれている場合は，信号Aのパワースペクトルにも高周波数成分が現れるので，両者を区別するのは困難になる．

一方，乱流のような非定常信号には，間欠的に現れる信号成分が含まれることが多い．図 1 は円柱後流の速度信号を測定した例である．カルマン渦の基本周波数成分が定常的に発生しているが，その他に局所的に強い信号が含まれている．このような局所的な信号は，振幅が大きいにもかかわらず，スペクトル解析では見逃してしまう可能性もある．

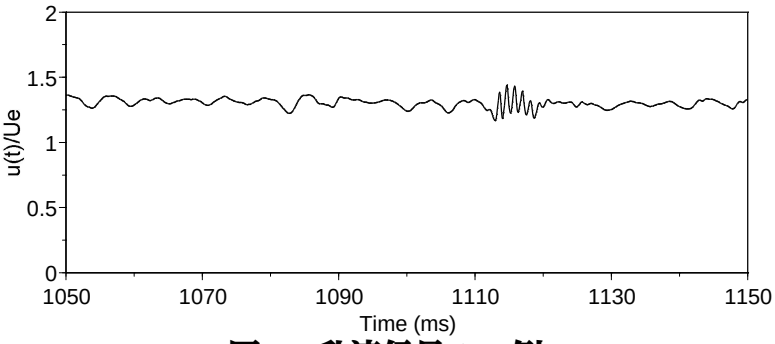


図 1　乱流信号の一例

つまり，信号 $x(t)$ をフーリエ変換することによって得られるスペクトル $X(f)$は周波数のみの関数であり，時間の情報は失われています．

スペクトルの時間的変化を求めるためには, 信号の一部を窓関数（Window Function）を 用いて切り出し, この窓をずらしながら, その区間の信号のスペクトルを次々に解析する必要がある.

信号の周波数の時間的変化を解析する手法として, 短時間フーリエ変換（Short-time Fourier Transform）と連続ウェーブレット変換 (Continuous Wavelet Transform) がある.

2. 短時間フーリエ変換（Short-time Fourier Transform）

短時間フーリエ変換では, 一定の大きさの窓関数を用いて信号を切り出し, その結果をフーリエ変換してスペクトルを計算する. すなわち, 設定した1つの窓に対して 1 組のスペクトルが得られるので, これらを集めることにより, スペクトルの時間的変化を求める手法である.

入力信号 $x(t)$ を, 時間 b を中心としたガウス窓（Gaussian Window）を用いて切り出した信号のスペクトル $X(b,?)$ は

$$X_s(b,\omega)=\frac{1}{2\sqrt{\pi}\sigma}\int_{-\infty}^{\infty}\exp\left(-\frac{(t-b)^2}{\sigma^2}\right)x(t)\exp(-i\omega t)dt$$

と表すことができる.

なお, s はガウス分布の広り, すなわち切り出す信号のサイズを決める定数であり, スペクトルは時間 b と周波数 ? の関数である.

このようなガウス窓を用いた短時間フーリエ変換を Gabor 変換 と呼ぶこともある.

ここで, 信号にガウス窓を乗じる操作とフーリエ変換を 1 つの変換操作とみなしてまとめると, 次のような関数 Gs(t,?) が得られます.

$$G_s(t,\omega)=\frac{1}{2\sqrt{\pi}\sigma}\exp\left(-\frac{t^2}{\sigma^2}\right)\exp(-i\omega t)$$

ここで, s と ? は独立であり, 例えば s を固定したまま ? の値を変化させると, 関数 Gs(t,?) の形状は複雑に変化する. （極大点と極小点の数は, ほぼ ? に比例して増加する. ）
一般に, 周波数の分解能は窓関数のサイズに反比例するので, この窓関数のサイズを大きく設定すると, 周波数分解能は向上するが, 時間軸に関する情報, すなわち時間分解能が低下する.
通常のスペクトル解析はこの時間窓を無限に大きくとった場合である.

一方, 窓関数のサイズを小さく設定すると, 時間分解能は向上するが, （本来, 無限に続く周期関数を仮定したフーリエ解析の性質から）周波数分解能は逆に低下する. このように, 周波数分解能と時間分解能は, 一方が高くなると他方は低下する. これを「不確定性の原理」と言う.

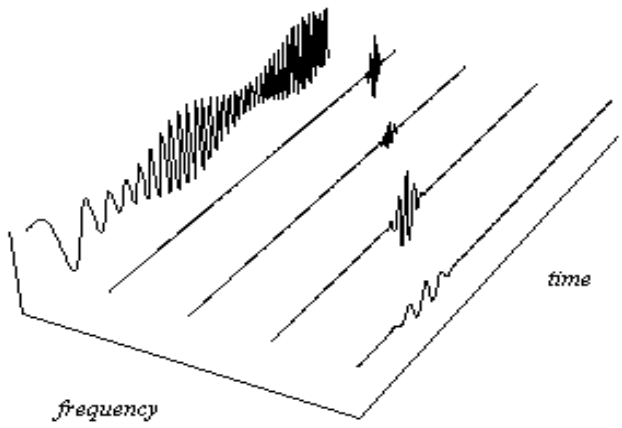
すなわち, 短時間フーリエ変換では, 周波数分解能と時間分解能を両立させることができない.
このような問題を解決するため, フランス人の石油探査技師であった Morlet は, 1980 年初頭に次に示す連続ウェーブレット変換を考案した.

3. 連続ウェーブレット変換（Continuous Wavelet Transform）

短時間フーリエ変換では, 窓のサイズと周波数を独立に変化させることができるので, たとえば, 上に示したGabor変換では, s を固定したまま? の値を変化させると, 関数 Gs(t,?) の 形状が, 複雑に変化する.

ウェーブレットとは「さざ波」のことですが, ウェーブレット変換では, マザーウェーブレットと呼ばれる基本的な関数を拡大・縮小させることにより, 信号の周波数－時間軸の解析を行う.
　上記 Gabor 変換に対応するマザーウェーブレットを, 次式で表します.

$$\psi(t) = \frac{1}{2\sqrt{\pi}\sigma}\exp\left(-\frac{t^2}{\sigma^2}\right)\exp(-it)$$



この関数 ? (t)は, 複素正弦波にガウス関数を乗じたものとなっていますが, 周波数 ? には依存しない点が, 短時間フーリエ変換の場合と異なる.

ここで, 上のマザーウェーブレットを, 時間方向に拡大・縮小・シフトするに ? (t)をパラメータ a と b を用いて

$$\psi\left(\frac{t-b}{a}\right)$$

と表す.
パラメータの a は, マザーウェーブレット? (t) を時間方向に拡大・縮小する比率を決定するものであり, 周波数に相当する. また, パラメータの b は, 短時間フーリエ変換と同様時間のシフト量に対応し, ガウス窓の位置（解析する時間）を決定する.

この関数を用いて連続ウェーブレット変換を以下のように定義する.

$$G_w(b,a) = \int_{-\infty}^{\infty} \frac{1}{\sqrt{a}}\overline{\psi\left(-\frac{t-b}{a}\right)}x(t)dt$$

ここで示した連続ウェーブレット変換を, Gabor のウェーブレット変換（時に, 単に Gabor 変換）と言

う. なおこの変換は，フーリエ変換で成立したような直交規底の条件を満たしていない点に注意する必要がある.

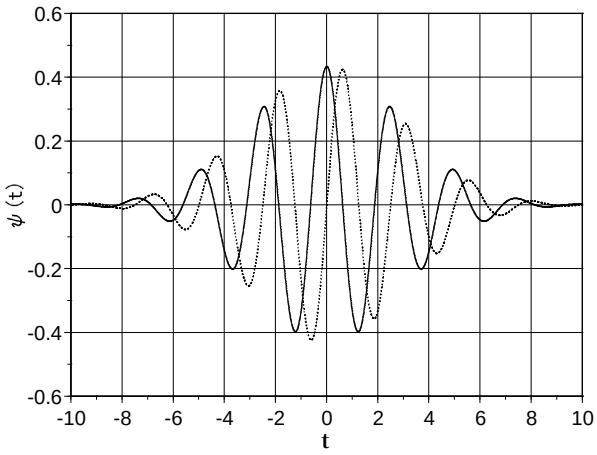


図 2　マザーウェーブレット

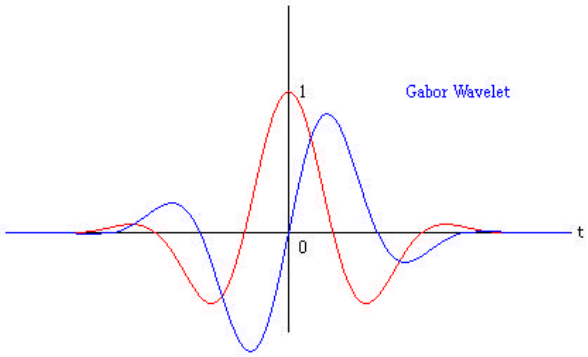
4.　代表的な連続ウェーブレット関数

連続ウェーブレット変換のマザーウェーブレットとして，以下のような関数が用いられます.

(1) Gabor のウェーブレット

先に示したように，Gabor のウェーブレット関数は複素数となり，以下のように表されます.
時間 t の絶対値が大きくなっても，この関数値は完全に 0 にはならず，いわゆる「コンパクト・サポート」ではないが，マザーウェーブレットが周波数と一対一に対応しているので，信号の局所的な周波数成分を検出するのに適している.

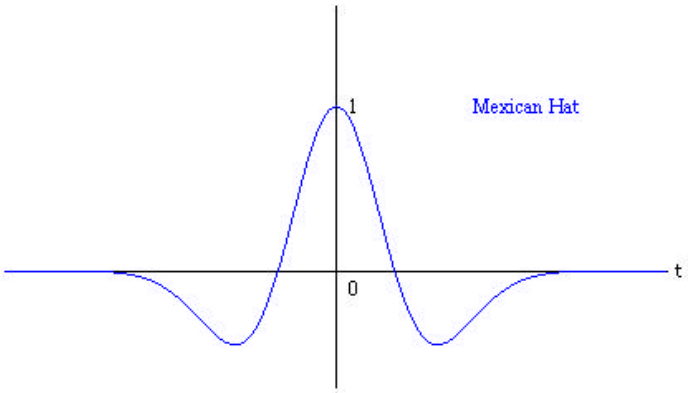
$$\psi(t) = \frac{1}{2\sqrt{\pi}\sigma}\exp\left(-\frac{t^2}{\sigma^2}\right)\exp(-it)$$



(2) Mexican hat

ガウス関数の 2 次の導関数はメキシカンハットのような形状となり, これをマザーウェーブレットとする. 式で表すと次のような実関数となり, 計算が簡単なのでウェーブレットを学習する際に良く利用される.

$$\psi(t) = (1 - 2t^2)\exp(-t^2)$$



(3) French hat

この関数も次式で示されるような実数値であり, (2)の Mexican hat を矩形を用いて単純化したような 形状をしている.

両端で完全に 0 となるので, コンパクト･サポートと呼ばれますが, 直交ウェーブレットではない.

$$\psi(t) = \begin{cases} 1 & -1 \le t \le 1 \\ -0.5 & -3 \le t \le -1, 1 \le t \le 3 \\ 0 & \text{上記以外} \end{cases}$$

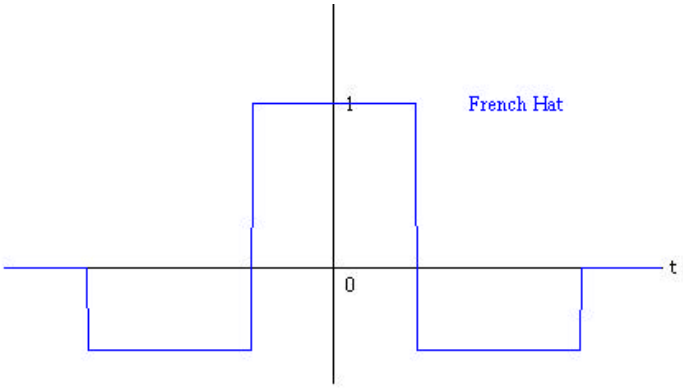


図 3 はGabor変換を用いて信号波形をWevelet解析した結果である. 局所的に現れた乱雑信号が捉えられていることがわかる.

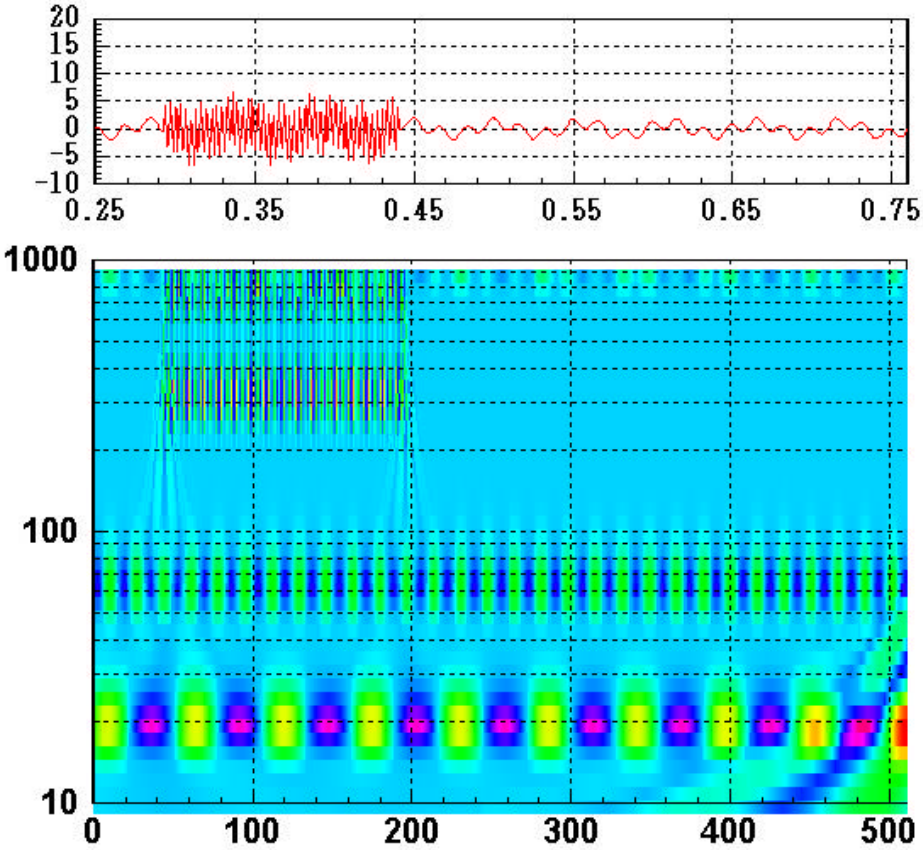


図 3　ウェーブレット変換による過渡応答信号の解析例

```
/*********************************************/
/* Wavelet Transform Program (Gabor)        */
/*                 1996.6.21 Akiyoshi Iida  */
/*********************************************/
# include <stdio.h>
# include <math.h>
# define PI       3.14159265358979
# define DIV      6.0
# define A0       (pow(2.0,1.0/(float)DIV))
# define M             42
# define SIGMA    0.8909
# define OMEGA    0.4
# define GN       200
# define GNC      100
# define KEI      0.01162
# define N_DATA   3050

char *progname;
void main(argc ,argv)
        int argc ;
        char *argv[];
{
        int      i,k;
        float  x[N_DATA];
        float sr[N_DATA][M],si[N_DATA][M],u[N_DATA];
        float w,dt,dummy;

        FILE     *fp,*fd, *fopen() ;
        fp=fopen(argv[1],"r");

        for(i=0; i<N_DATA; i++)
                {

        fscanf(fp,"%f %f %f\n",&u[i],&dummy,&dummy) ;
                        x[i]=(float)i*0.0001;
                }
        dt=x[1]-x[0];
        fclose(fp);

        fp=fopen(argv[2],"w");

        for(i=0; i<N_DATA-1024; i++)
        {
                fprintf(fp,"%d %f\n",i,u[i+512]) ;
        }
        fclose(fp);

        for(i=0; i<512; i++)
        {
                w=0.02*(float)(i-128);
                u[i] *= (tanh(w)+1.0)/2.0;
                u[N_DATA-i]*=(tanh(w)+1.0)/2.0;
        }

                wt12(u,sr,si,N_DATA);
                fp=fopen(argv[3],"w");
                fd=fopen(argv[4],"w");
                fprintf(fp,"Real ");
                fprintf(fd,"Abs ");
                for (k=0;k<M;k++)
                {
                        fprintf(fp,"%f
",1.414216562/(PI*A0*pow(2.0,((float)k/DIV))*dt)) ;
                        fprintf(fd,"%f
",1.414216562/(2.0*A0*pow(PI,((float)k/DIV))*dt));
                }
                fprintf(fp,"\n");
                fprintf(fd,"\n");
                for (i=0; i<N_DATA-1024; i++)
                {
                        fprintf(fp,"%d ",i);
                        fprintf(fd,"%d ",i);
                        for (k=0;k<M;k++)
                        {
                                fprintf(fp,"%f ",sr[i+512][k]) ;
                                fprintf(fd,"%f ",si[i+512][k]) ;
                        }
                        fprintf(fp,"\n");
                        fprintf(fd,"\n") ;
                }
                fclose(fp);
                fclose(fd);

        iwt12(u,sr,si,N_DATA);
        for(i=0;i<N_DATA;i++)
        {
                printf("%d %e\n",i,u[i]);
        }
}

wt12(u,sr,si,num)
        float sr[][M],si[][M],*u;
        int num;
```

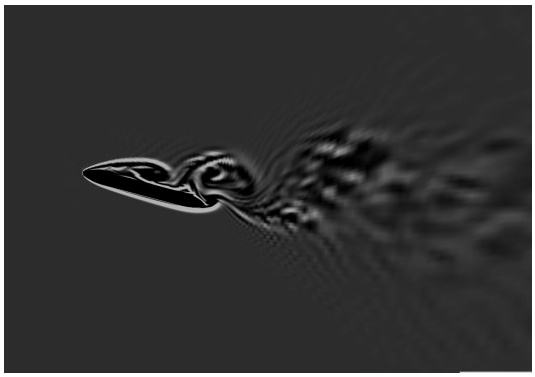
```
{
        int i,n,m;
        float qr[GN],qi[GN];
        float sumr,sumi,a0,norm;
        float pp,qq,rr;

        a0=A0;
        norm=sqrt(sqrt(PI))/sqrt(SIGMA);
        pp=norm*pow(a0,(-(float)m/2.0))*
                                exp(-
1.0/SIGMA/SIGMA/2.0*OMEGA*OMEGA
                                *((float)i-GNC)
                                *((float)i-GNC)/rr/rr);
                        qr[i]=pp*cos(     2.0*PI*OMEGA*((float)i-
GNC)/rr);
                        qi[i]=pp*sin(-2.0*PI*OMEGA*((float)i-
GNC)/rr);
                }
        for(n=0;n<num;n++)
        {
                sumr=0;
                sumi=0;
                for(i=0;i<GN;i++)
                {
                        sumr+=(float)u[(num+n+i-GNC)%num]*qr[i];
                        sumi+=(float)u[(num+n+i-GNC)%num]*qi[i];
                }
                sr[n][m]=sumr;
                si[n][m]=sumi;
        }
    }
}

iwt12(u,sr,si,num)
        float sr[][M],si[][M],*u;
        int num;
{
        int i,k,n,m;
        float qr[GN],qi[GN];
        float sumr,sumi,a0,norm;
        float pp,qq,rr;

        for(i=0;i<num;i++)
        {
                u[i]=0.0;
        }
        for (m=0;m<M;m++)
        {
                rr=pow(a0,(float)m);

                for (i=0;i<GN;i++)
                {


        a0=A0;
        norm=sqrt(sqrt(PI))/sqrt(SIGMA);

        for (m=0;m<M;m++)
        {
                rr=pow(a0,(float)m);


                for (i=0;i<GN;i++)
                {
                        pp=norm*pow(a0,(-(float)m/2.0))*
                                exp(-
1.0/SIGMA/SIGMA/2.0*OMEGA*OMEGA
                                *((float)i-GNC)
                                *((float)i-GNC)/rr/rr);
                        pp=pp/rr;
                        qr[i]=pp*cos(     2.0*PI*OMEGA*((float)i-
GNC)/rr);
                        qi[i]=pp*sin(-2.0*PI*OMEGA*((float)i-
GNC)/rr);
                }
        for(n=0;n<num;n++)
        {
                sumr=0;
                sumi=0;
                for(i=1;i<GN;i++)
                {
                        sumr+=(float)sr[(num+n+i-
GNC)%num][m]*qr[GN-i];
                        sumi+=(float)si[(num+n+i-
GNC)%num][m]*qi[GN-i];
                }
                u[n]+=sumr-sumi;
        }
    }
        for(k=0;k<num;k++)
        {
```

```
                u[k]=u[k]*KEI;
        }
}
```

4. 離散ウェーブレット

連続ウェーブレット変換は信号ののパターンや相似性の解析に用いられるのに対し，離散ウェーブレット変換は正規直交系となるため，データ圧縮やエネルギ解析等に用いられることが多い．本講義では離散ウェーブレットについては扱わないが，画像圧縮等の技術を乱流解析に摘要した一例を示す．

図 3 は二次元平板翼周りの空力音源分布を離散ウェーブレット変換を用いて画像処理（多重解像度解析）した結果の一例である．翼の負圧面ではく離した渦が音源となり，空力音が放射されていく様子を渦スケール別に可視化することが可能である．各スケールごとにアニメーションを作成することで，空力音を発生させている渦の時空間的な変化を調べることが可能である．



元画像（数値計算結果）

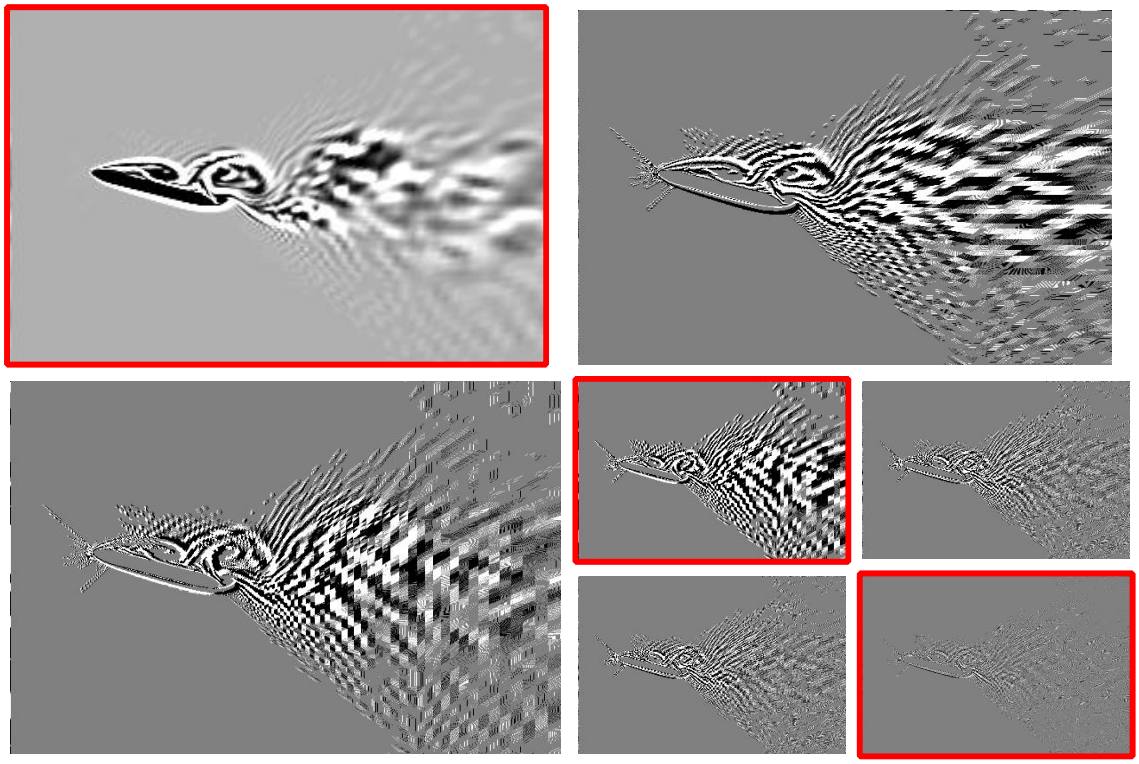


図 3　離散ウェーブレット解析の応用例