短時間フーリエ変換および離散ガボール変換の MATLAB 実装について *

○ 矢田部 浩平 (早稲田)

1 まえがき

音響信号の解析および処理において，時間周波数変換は欠かせないツールとなっている．特に，窓関数とフーリエ変換を用いた「短時間フーリエ変換 (STFT)」と，STFT を間引いた変換である「離散ガボール変換 (DGT)」がよく用いられており，音響工学のあらゆる分野で見ることができる．

STFT 自体は 1940 年代から論じられており，古くからある基礎的なトピックとして扱われることも多い．しかし，STFT や DGT の数理的な基盤ができたのは 1980 年代に入ってからであり，1990 年代に確立したことを考えると，そこまで古典的な話という感じでもない．実際，行列分解に基づく高速アルゴリズムが 2012 年に提案されるなど，近年においても発展が見られる分野である．ただし，それらは応用数学としての発展であり，音響工学などの応用分野にはあまり浸透していないように見受けられる．

STFT および DGT の基礎から発展まで様々なトピックを紹介するために，音響学会誌で半年に渡って「短時間フーリエ変換入門」と題した解説記事を連載した [1–6]．その第五回で実装に関する話題を扱い，付録として MATLAB 実装の配布も行った [7]．この実装は，ユーザー目線で簡単かつ高速に動作することを目標とし，実用的なツールに仕上がったと思っているが，英語の説明しか提供されていないのがハードルを高くしていると考えられる．そこで本稿では，公開したツールの説明書を日本語で残すことを目的とし，具体的なコード例で使用方法を説明する．

2 DGTtool オブジェクト

STFT および DGT を簡単に計算できるツールとして DGTtool オブジェクトを公開した [7]．MATLAB Central の File Exchange とも連携されているので，MATLAB のアドオン追加機能を用いてダウンロードすることもできる．とはいえ，「DGTtool」で Google 検索し，そこから飛ぶのが最も簡単かもしれない．

2.1 クイックスタート

DGTtool は，オブジェクトとして定義することでパラメータを内部で保持し，ユーザーが覚える必要のある手続きを可能な限り削減することを目指した．

DGTtool を使う際は，前準備として DGTtool オブジェクト F を生成する．例えば，コマンド

```
F = DGTtool
```

を実行すると，ワークスペースに F が現れる．以後，変換に必要なパラメータは全て F の内部に保存されるので，ユーザーが管理する必要はない．もちろん，F 以外の自由な変数名でオブジェクトを定義できるが，本稿では便宜上 F を用いる．なお，DGTtool を呼ぶ際にオプションを指定しない場合は，パラメータとしてデフォルト値が設定される (後述)．

一度 DGTtool オブジェクト F を生成した後は，

```
X = F(x);
```

によって信号 x をスペクトログラム X に変換できる．窓などの全てのパラメータを F が保持しているので，変換の際に信号 x 以外の引数は取らない．同様に，スペクトログラム X を時間領域に逆変換するには

```
x = F.pinv(X);
```

とすれば良い．信号を完全再構成するために必要な諸々の計算 [4, 5] は，F.pinv を最初に実行した際に行われ，二度目以降は使い回すことで，何度も変換する場合における無駄な計算を省いている．なお，再構成された信号の長さは，必要なゼロ埋めによって元の信号長よりも長くなる場合があるが，ゼロ埋めの量は必要最小限に留めてある．

窓長やシフト幅などの各パラメータは，DGTtool オブジェクトを生成する際に指定する．例えば，

```
DGTtool('windowLength',K,'windowShift',a)
```

のようにオプションとして入力することで，窓長が K でシフト幅が a の変換となる．同様に，窓の種類は 'windowName'，周波数ビン数は 'FFTnum' で指定する (後述)．これらのオプションを入力する順番は自由で，全てのパラメータを指定する必要はない．

2.2 DGTtool の設計

実用的なツールとなることを目指して，DGTtool は次の三つの方針に基づいて設計されている．

- 扱いが容易で使い勝手が良い
- 内部的な実装では実行速度を優先
- アルゴリズム中での反復利用に適する

具体的な内容は以下で述べるが，これらの方針に興味がない場合は，この節は飛ばして差し支えない．

---

*On MATLAB implementation of STFT and DGT. By Kohei Yatabe (Waseda University).

まず第一に使い勝手を重視して実装した．個人的には，変換に関するパラメータを管理するのが面倒だと感じることが多い．また，引数の入力順を覚えたり，毎回パラメータを入力するのも面倒である．そこで，オブジェクト内にパラメータを保持し，変換自体は可能な限りシンプルに扱えるようにした．

実用面では実行速度も重要なので，各種計算は速度を優先して実装している．処理の上で不要な計算を省くことで，MATLAB のみで書かれた変換としてはかなり高速に動作する．代わりに，位相の取り扱いや数学的な利便性がいくらか犠牲になっている．

さらに，反復アルゴリズム内で何度も変換することを想定し，複数回の実行が高速になる工夫も行った．毎回共通で計算する情報を保持しておくことで，二回目以降の計算を省いて高速化した．多チャンネル信号にも対応しており，時間領域と時間周波数領域を行き来する最適化アルゴリズムを実装しやすくした．

2.3 メソッドの呼び出しについて

MATLAB におけるメソッドの扱いに馴染みのない場合に配慮して，ここで簡単にフォローしておく．

DGTtool は，窓などのパラメータをオブジェクト F の内部に保持している．メソッドと呼ばれる関数は，それらの内部情報を計算に利用できるので，引数をシンプルにすることができる．例えば，pinv メソッドの引数はスペクトログラム X のみで良く，その他の必要な情報は F の内部を参照する．

MATLAB では，メソッドは二種類の方法で呼び出すことができる．一つ目は上で用いたドット表記で，F.pinv(X) のようにオブジェクトとメソッド名の間にドットを挟んで呼び出す方法．もう一つは，オブジェクトを第一引数として pinv(F,X) のように代入する方法である．どちらを用いても良いが，本稿では引数がわかりやすいドット表記を用いる[1]．

3 使い方

DGTtool は単一の関数ファイルに書かれているので[2]，「現在のフォルダー」に DGTtool.m を配置するだけで利用することができる．もちろん，通常通り，パスが通っている他のフォルダに配置しても良い．

DGTtool のヘルプやドキュメンテーションは力を入れて書いた．コマンドラインに help DGTtool や doc DGTtool と打てば読むことができるので，適宜参照願いたい．なお，メソッドのヘルプを表示するには，help DGTtool/methodName のように，スラッシュで区切って指定する[3]．例えば，help DGTtool なら DGTtool 全体のヘルプが表示され，メソッドとしての DGTtool (コンストラクタ) のヘルプを表示するには help DGTtool/DGTtool と打つ必要がある．

表–1 DGTtool メソッドに入力できる Name,Value ペア

パラメータ名	値	デフォルト値
'windowLength'	正の整数	2048
'windowShift'	正の整数	256
'FFTnum'	正の整数	窓長と同じ
'windowName'	文字列	'4termC5Nuttall'
'windowVector'	縦ベクトル (double)	—

3.1 DGTtool オブジェクトの生成

DGTtool を使用するには，まず DGTtool オブジェクト F を生成する必要がある．具体的には，パラメータ名 Name とその値 Value をカンマで列挙して

```
F = DGTtool(Name,Value,...)
```

のように DGTtool メソッドに入力する．

入力可能なパラメータを表–1 に示す．パラメータはどの順番で入力しても良く，窓長と周波数ビン数は独立に決めることができる (周波数ビン数が窓長より小さくても良い)．また，入力しなかったパラメータは，表–1 のデフォルト値になる．窓は，種類を名称で指定するか，直接ベクトルで入力する．ただし，ベクトルで入力した場合は，名称と窓長は無視される．入力できる名称は help DGTtool/DGTtool で確認でき，前方一致していれば認識される[4]．

設定したパラメータは，上記のようにセミコロンを付けなければ，まとめて表示される．また，ワークスペースに生成された DGTtool オブジェクトをダブルクリックして確認することもできる．

3.2 順変換

信号 x をスペクトログラム X に変換するには，

```
X = F(x);
```

とすれば良い．ただし，信号 x は「縦ベクトル」か「時間 × チャンネルの二次元配列」であり，信号長はチャンネル数よりも大きくなければいけない．X は，周波数 × 時間 × チャンネルの配列となる．また，

```
[X,f,t] = F(x);
```

のように戻り値を複数個指定することで，サンプリング周波数で正規化された周波数 f と，窓の中心時刻 (サンプル) t を得ることができる．

[1] 順変換を計算する F(x) だけは特殊で，括弧により subsref メソッドを呼び出し，その中で DGT メソッドを呼んでいる．これは，順変換は F.DGT(x) よりも F(x) の方がわかりやすいと思うのと，作用素っぽく書きたいのでこのようにした．

[2] 全ての機能を単一のファイルに実装したのは，他のコードと一緒に DGTtool を再配布しやすいようにという意図がある．

[3] スラッシュの代わりにドットで区切っても良い．

[4] 例えば，'b' のみを入力するだけで，'Blackman' として認識される．Nuttall 窓以外は，頭文字のみを入力すれば良い．

DGTtool では，信号は周期列として扱われ，最初のサンプルと最後のサンプルが隣り合っていると思って変換する [3]．また，与えられたパラメータと信号長が不整合を起こす場合は，信号が自動的にゼロ埋めされる．これらの仕様は，スペクトログラムから信号を再構成できるようにするために設定されている [4]．

3.3 逆変換

DGTtool によって計算されたスペクトログラム X は，順変換と同じ DGTtool オブジェクト F を用いて

```
x = F.pinv(X);
```

とすれば信号 x に逆変換できる．ただし前述のように，再構成された信号 x はゼロ埋めされていることがある．また，計算過程で丸め誤差が発生するので，計算機イプシロンの定数倍の誤差は存在する．

メソッド名 pinv は MATLAB 標準関数の pinv からとっており，Moore–Penrose の疑似逆行列を表す．これは，F を行列として考え，F(x) を行列ベクトル積だと思ったときに，F.pinv(X) が F の疑似逆行列をかける操作に対応するからである [4]．すなわち，F.pinv(F(x)) は単位行列をかけることに相当し，計算結果は (丸め誤差を除いて) 元の信号となる．

疑似逆の計算は，窓によって実装されている [4]．F.pinv が呼ばれた際に，dualWin が存在しなければ計算する．これは，完全再構成を実現するための合成窓で，双対窓と呼ばれている [4]．ある窓の双対窓は無数に存在し，その中で疑似逆行列に対応するものは標準双対窓と呼ばれる [4]．dualWin を管理するために，双対窓かどうかを表す isDual と，標準双対窓かどうかを表す isCanonical というフラグが用意されている．F.pinv は，dualWin が双対窓でないときは標準双対窓を計算し直す．また，dualWin が標準双対窓でない一般の双対窓の場合は警告を発する．

3.4 完全再構成ではない逆変換

F.pinv は窓として標準双対窓のみを想定しているので，それ以外の窓を使用するために F.H も実装されている．これは，F を行列として考えたときの複素共役転置に対応し，スペクトログラム X から

```
y = F.H(X);
```

のように時間領域の信号 y を計算できる．正しく複素共役転置になるように，デフォルトでは分析に用いた窓 win を合成にも用いる．そのため，一般に信号は再構成されず，F.H(F(x)) は x と異なる．

自由に窓を指定したいときのために，第二引数に

```
y = F.H(X,synthesisWindow);
```

のように窓を入力することもできる．この場合，入力された窓を用いて信号 y を計算する．ただし，そのような使用方法は基本的には想定していない．

信号を再構成できないのに F.H を実装したのは，最適化アルゴリズムで行列の転置が要求されることが多いからである．実際，最適化変数を時間領域で定義し，時間周波数領域でのコスト関数を考えると，自然に転置の計算が出てくる．それをシンプルに記述し，高速に計算するために実装したのが F.H である．

3.5 タイト窓の計算

分析と合成の両方に用いて完全再構成できる窓をタイト窓と呼ぶ [4]．DGTtool には，与えられたパラメータからタイト窓を計算するメソッドがあり，

```
F.makeWindowTight
```

とすれば，win の標準タイト窓が計算されて，win と dualWin に代入される．その結果，転置と疑似逆が一致し，F.H でも信号を再構成できるようになる [4]．ただし，周波数ビン数が窓長より短い場合は，

```
F.makeWindowTight(signalLength)
```

のように信号長を入力する必要がある．

3.6 スペクトログラムの描画

スペクトログラムの表示のみを行いたい場合や，窓などのパラメータを決めるのに有効なので，描画用の plot メソッドを実装している．信号 x のみを

```
F.plot(x)
```

と入力すれば，スペクトログラムが内部で計算されて描画される．また，周辺化に対応する時間波形と周波数スペクトルも一緒に描画される[5]．サンプリング周波数 fs がわかっている場合は，それも一緒に

```
F.plot(x,fs)
```

と入力すれば，軸が物理量として表示される．

位相スペクトログラムを観察するには，対数振幅を明度，位相を色相で表現した HSV 表示が有効である [8]．DGTtool では，plot と全く同じように

```
F.plotPhase(x)
```

とすれば HSV 表示でき，サンプリング周波数を第二引数に入力することもできる．なお，表示される位相は，窓の線形位相成分を除去してある [5]．

これら描画系のメソッドには，共通して'range', 'trunc', 'normalize' の三つのオプションが入力

[5]窓のシフト幅が 1 で，周波数ビン数が信号長と同じ STFT の場合，複素スペクトログラムを周波数方向に全て足すと時間波形となり，時間方向に全て足すと周波数スペクトルとなる．

表–2　plot 系メソッドのオプションのデフォルト値

	'range'	'trunc'	'normalize'
'plot'	80	0	true
'plotPhase'	40	15	true
'plotReassign'	100	0	true

できる．'range' [dB] は色軸の範囲を表し，例えば F.plot(x,'range',120) なら，図として最大値から最小値までの範囲が 120 dB で表示される．同様に，'trunc' [dB] は色軸を平行移動する．例えば，'trunc' を 10 に設定すると，色軸が小さい方に 10 dB シフトし，最大値から 10 dB の範囲は飽和する．'normalize' は窓の正規化を指定し，true であれば窓を sum(win)/2 で割って正規化する．これにより，正弦波のスペクトログラムのピークが 0 dB になる．これらの三つのオプションは，どの順番で入力しても良く，入力しない場合はデフォルト値が適用される．ただし，関数によってデフォルトが違うので，表–2 にまとめて示す．これらの値は，著者が適当に決めた．

3.7　スパース時間周波数解析

位相の偏微分である瞬時周波数と群遅延を用いて，信号をスパースに表現する時間周波数解析法が提案されている [6]．DGTtool では，上記と同様に

```
F.plotReassign(x)
```

と書けば，スパース時間周波数表現を描画することができる．サンプリング周波数やオプションを追加で入力できるのも共通である．この方法では，見た目がパラメータに大きく依存する．特に，時間周波数ビンの細かさによって解像度が決まるので，窓のシフト幅は小さく，周波数ビン数は大きくすると良い．

描画用メソッドの他にも，スパース時間周波数表現を計算するためのメソッドも実装されており，

```
S = F.reassign(x)
```

によってパワースペクトログラムを再割り当てしてスパースにした結果 S が得られる．他にも，戻り値として瞬時周波数や群遅延を返すこともできるので，help DGTtool/reassign を参照願いたい．なお，瞬時周波数の計算に微分窓 diffWin を用いている [6]．

3.8　パラメータの再設定

パラメータは，DGTtool オブジェクトを生成する際に設定する (3.1 節)．パラメータを変えたいときは，DGTtool オブジェクトを作り直すことを推奨するが，代入による変更も一応実装してある．例えば，周波数ビン数を変えるには F.FFTnum = 1000 などとすれば良い．窓のシフト幅を変える場合は F.shift = 100 のようにすれば良い．なお，redundancy という内部パラメータが冗長度を示しており，変換前の信号の何倍のメモリが変換後に必要かの目安を与えている [3]．この冗長度は，窓のシフト幅と周波数ビン数から自動的に計算されているが，F.redundancy = 8 のように値を代入することで，冗長度がその値になるように周波数ビン数を変更する機能も実装している．

窓を再設定するには，次のようにすれば良い．

```
F.setWindow(windowLength,windowName)
```

入力された変数は直接 getWindow に渡されるので，詳細は help DGTtool/getWindow を参照願いたい．

3.9　位相の変換

時間原点の定義や窓の定義によって，位相の表現が変わる [5, 9, 10]．窓の線形位相成分を除去するには，X = F.makeZeroPhase(X) とすれば良い．また，時間原点の定義は，X = F.changeDGTdef(X) によって変えることができる．なお，undoMakeZeroPhase と undoChangeDGTdef で，それらの効果を打ち消せる．これらのメソッドは，冒頭で述べたように実行速度を優先した結果，使い勝手と精度の点で損をしている．

3.10　静的メソッド

ここまで紹介したメソッドは，DGTtool オブジェクトを生成した後で使えるものである．一方，オブジェクトを生成せずとも使える静的メソッドも実装されていて，外から呼び出せるようになっている．

静的メソッドは，次のように使用する．

```
out = DGTtool.methodName(in1,in2,...)
```

例えば，DGTtool.getWindow で窓を計算できる．他にも，DGTtool.zeroPad で多チャンネル信号のゼロ埋めができる．それらメソッドの一覧および詳細は，help DGTtool および doc DGTtool で確認願いたい．

参考文献

[1] 矢田部浩平, “[連載講座：短時間フーリエ変換入門] 第一回：連続信号と離散信号,” 日本音響学会誌, **77**(4), 262–269 (2021).
[2] 矢田部浩平, “[連載講座：短時間フーリエ変換入門] 第二回：離散フーリエ変換,” 日本音響学会誌, **77**(5), 331–338 (2021).
[3] 矢田部浩平, “[連載講座：短時間フーリエ変換入門] 第三回：短時間フーリエ変換,” 日本音響学会誌, **77**(6), 396–403 (2021).
[4] 矢田部浩平, “[連載講座：短時間フーリエ変換入門] 第四回：信号の再構成と窓関数,” 日本音響学会誌, **77**(7), 463–470 (2021).
[5] 矢田部浩平, “[連載講座：短時間フーリエ変換入門] 第五回：実装における諸注意,” 日本音響学会誌, **77**(8), (2021).
[6] 矢田部浩平, “[連載講座：短時間フーリエ変換入門] 第六回：時間周波数領域のスパース表現,” 日本音響学会誌, **77**(9), (2021).
[7] DGTtool (MATLAB による STFT および DGT のシンプルな実装例), https://github.com/KoheiYatabe/DGTtool または https://doi.org/gjzj
[8] 矢田部浩平, “音響信号処理における位相復元,” 電子情報通信学会 Fundamentals Review, **15**(1), 25–36 (2021).
[9] 矢田部浩平, 升山義紀, 草野翼, 及川靖広, “位相変換による複素スペクトログラムの表現,” 日本音響学会誌, **75**(3), 147–155 (2019).
[10] K. Yatabe, Y. Masuyama, T. Kusano and Y. Oikawa, “Representation of complex spectrogram via phase conversion,” *Acoust. Sci. & Tech.*, **40**(3), 170–177 (2019).