オーディオ信号処理で学ぶDSP

第2回　FIRフィルタを使ってみよう

堀江 誠一

DSPの重要な用途の一つにフィルタ処理がある．一般的なCPUでは，フィルタ処理のような演算を浮動小数点数を用いて処理する．これに対して，DSPは固定小数点数を用いた演算を行うという特徴がある．そこで今回は，フィルタの一つであるFIRフィルタの実装と，C++において固定小数点数を使ったプログラミングの実際について紹介する．（編集部）

フィルタは，DSPにおける主要なアプリケーションの一つです．フィルタだけのためにDSPを使うことは少なくなりましたが，アプリケーションの内部でフィルタが使われるケースは今でも数多くあります．

DSPによるフィルタはアナログ回路と同じく，周波数による信号の選択に使われるほか，設計の自由度を生かして遅延線や位相の制御にも使われます．特にFIRフィルタはアナログ回路では実装が難しい反面，アナログ・フィルタでは実現できない処理を具現化できるためDSP向きです．また，IIRフィルタはアナログ・フィルタの設計手法をそのまま使える上，DSPで実装しても演算量が小さいというメリットがあります．

今回は，FIRフィルタを実装する方法を見てみましょう．

1．FIRフィルタの原理

● インパルス関数とインパルス応答

FIRフィルタとはどのようなフィルタでしょうか．FIRとはFinite Impulse Responseの略，つまり「有限時間でインパルス応答を打ち切る」ということを意味しています．しかしこれは実装の特徴であって，FIRフィルタの一番の特徴とは言えません．ユーザから見たときのFIRフィルタの特徴は，インパルス応答をありのまま設計値として扱うことにあります．

インパルス応答とはシステムにインパルス関数を入力したときの出力のことですが，これでは何のことか分かりにくいですね．分かりやすいように例えると，インパルス応答とは鐘の音です（**図1**）．

お寺の鐘の音というと，「ご〜ん」という除夜の鐘の響きを思い浮かべますが，実際にはその音だけではありません．引っかいたり，引きずり回したり，やすりで削ったりすれば，それぞれ違う音がします．

どのようなことをしたときにどのような音が出るかを想像しろといわれても困ってしまいます．しかし数学的には，鐘のインパルス応答を知っていれば，引っかく，やすりで削るといった入力に対する音も計算可能です．

ではインパルス応答とは何かというと，それは鐘を撞木（しゅもく）で突いたときの音だと思ってかまいません．鐘を撞木で突くと，突いた瞬間だけ鐘に衝撃が与えられます．このように「一瞬だけ与えられる入力」をインパルス関数と呼びます（**図2**）．

鐘を突くと鳴る音は，その鐘のインパルス応答です．この応答を知ることで，いろいろな入力に対する鐘の鳴り方を知ることができます．FIRフィルタとはこのインパルス応答をそのまま係数として実装してしまおうというものです．

● インパルス応答の実装

実装そのものは単純で，あらかじめ配列に格納しておいたインパルス応答と，入力データ列の間でベクトルの内積を計算するだけです．入力データが一つ入ってくるたびに

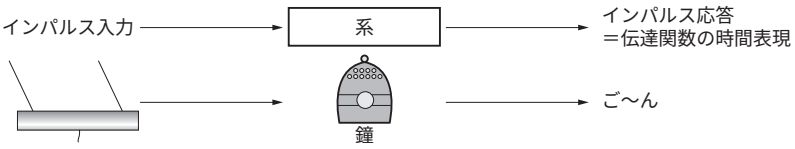

図1
鐘の音とインパルス応答
寺の鐘を突くと，ゴーンという音が鳴る．このとき，撞木から与えられる力は瞬間的なものになる．この力を正規化したものがインパルス関数．系にインパルスを入力すると，伝達関数（の時間表現）が出力となって出てくる．鐘の音は鐘の伝達関数と思っていい

KEYWORD ——DSP，信号処理，FIRフィルタ，インパルス応答，SciLab，固定小数点表記

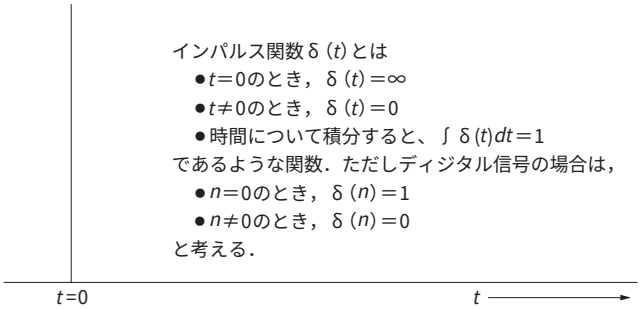

図2　インパルス関数

入力データ列がずれていくので，その都度内積を計算します．こうして得られた内積の列が系の出力です．

どのような入力に対しても，インパルス応答と入力データ列の内積を計算すればその系の出力が分かります[注1]．これがFIRフィルタの原理であり，動作そのものです．サンプルごとに内積を計算するため，FIRフィルタは積とその和の計算の塊となります．ここで使われる積と和の組み合わせ計算が「積和演算」です．

さて，ここまで読んで，不安を感じる方がいるかもしれません．そもそもどうやってインパルス応答を知るのでしょうか．実際に存在する鐘であれば，叩いてみればインパルス応答らしきものは分かります．しかし，希望する周波数特性を持ったフィルタのインパルス応答はどうすれば分かるのでしょうか．

幸いなことに，この点について深く考えなくても簡単に設計を行ってくれるツールがあります．これらのツールは周波数特性を指定するだけでFIRフィルタ用のインパルス応答を計算してくれるため，設計者が深く悩む必要はありません．

2. FIRフィルタの設計と実装

● SciLabを使ってFIRフィルタを設計する

それではFIRフィルタを設計してみましょう．設計に使うのはInterface誌2006年9月号の特集で紹介されたSciLabです．このソフトウェアを使えば，FIRフィルタを比較的簡単に作ることができます．手始めに，遮断周波数5kHzのLPF（ローパス・フィルタ）を作ってみましょう．

注1：ただし線形な系に限る．

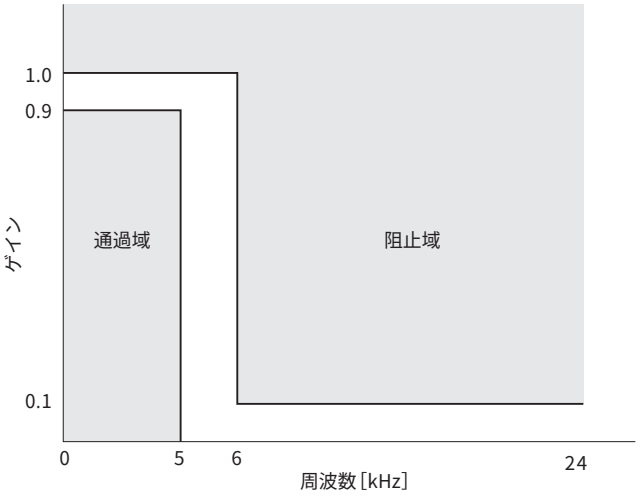

図3　希望するフィルタの特性

希望するフィルタの特性は**図3**のようなものであるとします．遮断周波数は5kHz，通過帯域内のゲインは1.0，阻止域は6kHzから始まり，阻止域内のゲインは0とします．また，サンプル周波数は48kHzとします．遮断周波数と阻止域の開始を48kHzで正規化すると，それぞれ0.1041667，0.125となります．

この仕様をもとに，SciLabでFIRフィルタを設計してみましょう．SciLabにはFIRフィルタを設計してくれる関数がすでに存在しています．この関数は`eqfir()`という名前で，仕様を引き数として与えると，設計したFIRフィルタのインパルス応答をベクトルとして出力します．

図4を見てください．`eqfir()`関数に四つの引き数を与えます．最初の引き数は整数で，設計するFIRフィルタのタップ数（インパルス応答の長さ）を与えます．この例では63を与えているので，出力されるインパルス応答の長さは63個になります．このインパルス応答そのものが，FIRの設計結果となります．

2番目の引き数は，設計帯域の下限周波数と上限周波数をサンプル周波数で正規化した値として与えます．上限と下限を組みとしてベクトルを作り，それを帯域の数だけ重ねて行列にします（**図5**）．3番目の引き数はベクトルで，要素はそれぞれ対応する帯域のゲインを指定します．

4番目の引き数だけはトライ＆エラーが必要になります．これは設計にあたってそれぞれの帯域に与える重みです．重みとはつまり，どのくらい設計目標を重視するかということです．1.0がもっとも重く，目標について妥協を許し