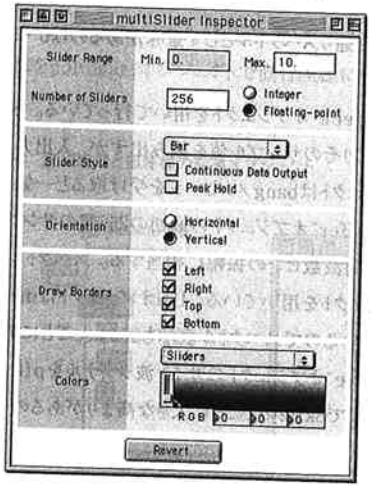
■5-18-18 mulitiSliderオブジェクトのインスペクター



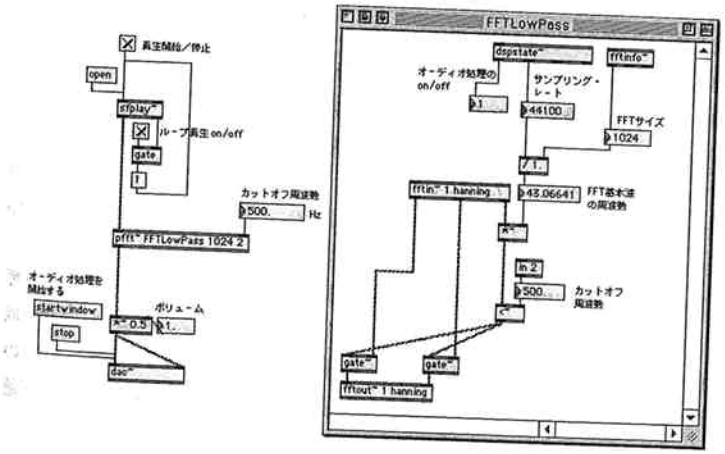
5-19 FFT：スペクトル合成

FFTおよびIFFTが理解できると、今度はさまざまなアイディアによってスペクトルそのもの変形させたり、コントロールしてみたくなる。実際、スペクトルそのものを直接操作することは、音の内部構造に分け入って、音を内側から操作するようなものだ。ここに極めて魅力的なスペクトル合成という領域が広がることになる。ここではスペクトル合成の入門的な事例を具体的に紹介してみたい。

スペクトル・フィルター

スペクトルは入力信号を周波数帯域ごとのエネルギーとして表したものだ。ここで直ぐに思いつくのが、スペクトルの特定の周波数帯域をカットしてフィルター処理を行うことだろう。5-19-1のパッチでは、pfft~オブジェクトに読み込んだサブ・パッチ"FFT LowPass"の中で指定したカットオフ周波数以上の周波数について、エネルギーを0にする処理を行っており、その結果として周波数領域でのローパス・フィルターを実現している。

■5-19-1 スペクトル・ローパス・フィルターの例



サブ・パッチ"FFTLowPass"の処理内容を見てみよう。オーディオ処理を開始すると、fftinfo~オブジェクトはサブ・パッチが読み込まれているpfft~オブジェクトのFFTサイズやホップサイズを報告してくれる。ここではFFTサイズの値を用い、dspstate~オブジェクトが報告する現在のサンプリング・レートを割り算してFFT基本波の周波数を計算している。計算内容は、44100/1024＝43.06...になっている。

次に、このFFT基本波の周波数はfftin~オブジェクトの第3アウトレットから出力される周波数ビンのインデックスと掛け合わされる。つまり0Hzからサンプリング・レートの1/2まで、順に43.06Hzごとに FFT高調波の周波数を出力することになる。この高調波周波数とメイン・パッチから指定されるカットオフ周波数を比較し、もし高調波の周波数の方が小さければ、gate~オブジェクトを開いてfftin~オブジェクトの出力シグナルをfftout~オブジェクトに渡す。そうでなければgate~オブジェクトはシグナル0を出力し、fftout~オブジェクトに渡す。

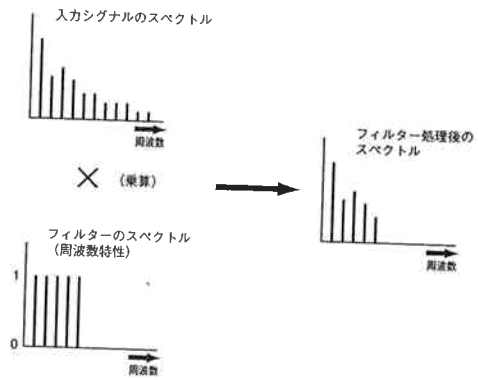
inオブジェクトは、通常はpoly~オブジェクトで用いられるが、pfft~オブジェクトのサブ・パッチ内でも機能し、メイン・パッチからメッセージを受け取る。アーギュメントとして書かれた2はインレット番号を意味する。

このパッチで行っていることは、fftin~オブジェクトの出力シグナルのうち、ある条件を満たすサンプルはそのまま何も手を加えず、それ以外のサンプルは強制的に0にしているということだ。言い換えれば、ある条件（カットオフ周波数より小さいという条件）を満たすサンプルには1を乗算し、それ以外のサンプルは0を乗算していることと同じ意味を持つ。このことを図式的に示したのが5-19-2だ。これは入力シグナルのスペクトルとローパス・フィルターのスペクトル（いままで周波数特性と呼んできたもの）を、個々の周波数ビンごとに乗算していることにほかならない。

スペクトル同士を掛け合わす処理は、時間領域では"畳み込み"と呼ばれるものだ。この例に挙げたフィルターだけでなく、エコー、リバーブといったおなじみの信号処理は、時間領域ではディレイの原理を用いて実現する。そこで行われているのが畳み込みである。

これを周波数領域の処理に置き換えると、まさにスペクトル同士を掛け合わす処理として実現することができる。しかも、スペクトルを直接操作することにより、、時間領域の畳み込み（間接的な方法）が持つ限界を軽々と突破することができる。例えば、今のローパス・フィルターでは、カットオフ周波数より大きな周波数帯は完全に除去され、逆

■5-19-2　スペクトル同士の乗算



にカットオフ周波数より小さな周波数帯は弱められることがない。いわば直角のフィルター特性が得られるのである。このようなフィルターを時間領域においてbiquad~オブジェクトやlores~オブジェクトで実現するこは不可能だろう。

512バンド・パラメトリック・イコライザー

スペクトル同士を掛け合わす畳み込みについて、もう少し明確な例としてパラメトリック・イコライザー・パッチを紹介しよう。パラメトリック・イコライザーは、周波数帯域（バンド）を細かく分けて、それぞれの周波数帯域ごとに入力信号のゲインをコントロールするものだ。

時間領域でパラメトリック・イコライザーを実現しようとすれば、複数のバンドパス・フィルターを並列して周波数帯域ごとのコントロールを行うことになる。当然、数多くのバンドパス・フィルターを同時に使用すればCPUへの負担は大きくなるので、そのバンド数は制限されるだろう。

5-19-3で紹介するパッチは、全周波数帯域を512ものバンドに分けてパラメトリック・イコライザーを実現するものである。pfft~オブジェクトが読み込むサブ・パッチの処理

はいたって簡単で、index~オブジェクトによりbuffer~ EqFuncオブジェクトの波形を順に読み出し、それが0より大きければfftin~オブジェクトの出力シグナルに乗算してfftout~オブジェクトに渡し、そうでなければ0を乗算してfftout~オブジェクトに渡している。index~オブジェクトはサンプル・インデックスによりbuffer~オブジェクトの内容を読み出すオブジェクトであり、ここでは、fftin~オブジェクトの第3アウトレットから出力される周波数ビンのインデックスを用いて読み出しを行っている。つまりインデックス0番～511番までを順番に読み出すことを繰り返す。

buffer~ EqFuncオブジェクトの内容はメイン・パッチ上で作られる。まず、パッチを開いた瞬間にloadbangオブジェクトのトリガーによりbuffer~オブジェクトのサイズが512サンプル分に設定される。なぜ512サンプルなのかについては後述しよう。

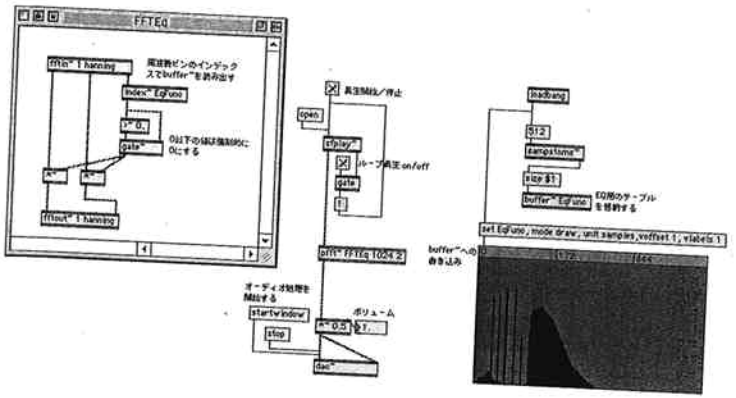
続いてwaveform~オブジェクトに5つのメッセージが送られ初期状態を設定する。まず、set EqFuncメッセージにより、waveform~オブジェクトはbuffer~ EqFuncオブジェクトとリンクする。mode drawメッセージでwaveform~オブジェクトに対するマウス操作が鉛筆ツールによる書き込みに設定され、直接buffer~ EqFuncオブジェクトの内容を書き込めるようになる。unit samplesメッセージは横軸の時間表示単位をサンプル単位にする。voffset 1メッセージは、縦軸の表示範囲のオフセット値を1にする。通常、waveform~オブジェクトの縦軸は−1～1の範囲で表示を行うが、オフセット値を1にすれば0～2の範囲で表示を行う。vlabels 1メッセージはオブジェクトの右側に縦軸の定規を表示するかどうかを設定するもので、1にすると表示を行う。

さて、オーディオ処理を開始し、sfplay~オブジェクトで適当なオーディオ・ファイルを再生し、そのシグナルをpfft~オブジェクトに送る。waveform~オブジェクト上にマウスを持ってくると、ポインターが鉛筆ツールに変わるので、それで自由に書き込みを行う。書き込んだ内容は、buffer~ EqFuncオブジェクトに反映され、pfft~オブジェクトのサブ・パッチ内で読み出されて周波数ビンごとに乗算される。

waveform~オブジェクトの横軸は、0Hzからナイキスト・レートまでの周波数ビンに対応している。その表示はパラメトリック・イコライザーの512本のフェーダーだ。

あるいは、見方を変えれば、これはフィルターの周波数特性のスペクトルであり、pfft~オブジェクトのサブ・パッチでの処理は、スペクトル同士を同じ周波数ビンごとに掛け合わせていることになる。周波数領域でスペクトル同士を掛け合わす畳み込みであれば、6-19-3のように通常では考えられないいびつな形のフィルター特性でも実現す

■5-19-3 スペクトル同士を掛け合わせるパラメトリック・イコライザー



ることが可能になる。

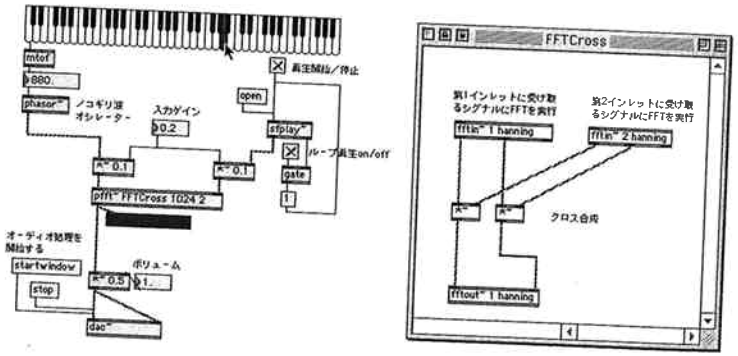
クロス合成

スペクトル同士を掛け合わせる畳み込みで、もう1つ手にすることのできる可能性は、2つの入力シグナルのスペクトル同士を掛け合わせる"クロス合成"だろう。これは時間領域でのシグナルの掛け合わせや、ミックスとは根本的に異なる。なぜならスペクトル同士を掛け合わせると、2つのシグナルの共通した周波数成分のみが強調され、一方に含まれて、他方に含まれないような周波数成分はカットされてしまうからだ。こうして合成されたシグナルは、元になる2つのシグナルのスペクトル上の性格をある程度は引き継ぎながらも、別のスペクトルを持った新たなサウンドとなる。したがって、クロス合成は"音のモーフィング"という興味深い領域の基礎となる。

5-19-4のパッチは最もシンプルなクロス合成の例だ。pfft~オブジェクトが読み込むサブ・パッチ"FFTCross"では、fftin~オブジェクトを2つ用意し、メイン・パッチから受け取る2つのシグナルそれぞれについてFFT処理を行い、FFT処理結果同士を乗算している。

メイン・パッチ上でpfft~オブジェクトの第1インレットにはphasor~オブジェクトから出力されるノコギリ波シグナルを入力し、第2インレットにはsfplay~オブジェクトで再生されるオーディオ・ファイルのシグナルを入力している。オーディオ・ファイルにどのようなものを選ぶかで得られる効果はさまざまであるが、例えば、話し声のオーディオ・ファイルだと、主にphasor~オブジェクトの周波数でピッチ感が決定され、言葉の内容も聞き取れるというボコーダー的なサウンドが得られる。

■5-19-4　シンプルなクロス合成の例



また、このパッチを5-19-5のように変更するとボコーダー的効果はより明瞭になる。サブ・パッチ"FFTCross2"では第2インレットに受け取るシグナルのFFT結果のみ、cartopol~オブジェクトで極座標に変換し、振幅の値のみを用いて第1インレットに受け取るシグナルのFFT結果をコントロールしている。第1インレットに受け取るシグナルがソースであり、それを第2インレットに受け取るシグナルでコントロールする関係だ。結果として、phasor~オブジェクトの出力シグナルによるピッチ感がより明確になる。

■5-19-5　クロス合成によるボコーダー・パッチ