**Fast Convolution**

Version 1.5: Jun 21, 2004 12:00 pm GMT-5

Douglas L. Jones

This work is produced by The Connexions Project and licensed under the Creative Commons Attribution License 

Abstract

Efficient computation of convolution using FFTs.

**Fast Convolution**

1 Fast Circular Convolution

Since,

$$\sum_{m=0}^{N-1} (x(m)(h(n-m)) mod N) = y(n) \text{ is equivalent to } Y(k) = X(k)H(k)$$

$y(n)$ can be computed as $y(n) = \text{IDFT}[\text{DFT}[x(n)] \cdot \text{DFT}[h(n)]]$

Cost

- **Direct**
  - $N^2$ complex multiplies.
  - $N(N-1)$ complex adds.

- **Via FFTs**
  - $3$ FFTs + $N$ multiplies.
  - $N + \frac{3N}{2} \log_2 N$ complex multiplies.
  - $3(N \log_2 N)$ complex adds.

If $H(k)$ can be precomputed, cost is only $2$ FFTs + $N$ multiplies.

2 Fast Linear Convolution

DFT\(^1\) produces circular convolution. For linear convolution, we must zero-pad sequences so that circular wrap-around always wraps over zeros.

To achieve linear convolution using fast circular convolution, we must use zero-padded DFTs of length $N \geq L + M - 1$

\(^1\)http://cnx.org/content/m12032/latest/#DFTequation

http://cnx.org/content/m12022/latest/
Figure 1

Figure 2
Choose shortest convenient $N$ (usually smallest power-of-two greater than or equal to $L + M - 1$)

$$y(n) = \text{IDFT}_N [\text{DFT}_N [x(n)] \text{DFT}_N [h(n)]]$$

**NOTE:** There is some inefficiency when compared to circular convolution due to longer zero-padded DFTs\(^2\). Still, $O\left(\frac{N}{\log_2 N}\right)$ savings over direct computation.

3 Running Convolution

Suppose $L = \infty$, as in a real time filter application, or ($L \gg M$). There are efficient block methods for computing fast convolution.

3.1 Overlap-Save (OLS) Method

Note that if a length-$M$ filter $h(n)$ is circularly convolved with a length-$N$ segment of a signal $x(n)$, the first $M - 1$ samples are wrapped around and thus is incorrect. However, for $M - 1 \leq n \leq N - 1$, the convolution is linear convolution, so these samples are correct. Thus $N - M + 1$ good outputs are produced for each length-$N$ circular convolution.

\(^2\)http://cnx.rice.edu/content/m12032/latest/#DFTequation

http://cnx.org/content/m12022/latest/
The Overlap-Save Method: Break long signal into successive blocks of $N$ samples, each block overlapping the previous block by $M - 1$ samples. Perform circular convolution of each block with filter $h(m)$. Discard first $M - 1$ points in each output block, and concatenate the remaining points to create $y(n)$.

Computation cost for a length-$N$ equals $2^n$ FFT per output sample is (assuming pre-computed $H(k)$) 2 FFTs and $N$ multiplies

\[
\frac{2 \left( \frac{N}{2} \log_2 N \right) + N}{N - M + 1} = \frac{N (\log_2 N + 1)}{N - M + 1} \text{complex multiplies}
\]

\[
\frac{2 (N \log_2 N)}{N - M + 1} = \frac{2N \log_2 N}{N - M + 1} \text{complex adds}
\]

Compare to $M$ multiplies, $M - 1$ adds per output point for direct method. For a given $M$, optimal $N$ can be determined by finding $N$ minimizing operation counts. Usually, optimal $N$ is $4M \leq N_{opt} \leq 8M$.

### 3.2 Overlap-Add (OLA) Method

Zero-pad length-$L$ blocks by $M - 1$ samples.

Add successive blocks, overlapped by $M - 1$ samples, so that the tails sum to produce the complete linear convolution. Computational Cost: Two length $N = L + M - 1$ FFTs and $M$ multiplies and $M - 1$ adds per $L$ output points; essentially the same as OLS method.
Figure 5
Figure 6