Advanced Digital Signal Processing: **Lecture 1**

Lecture date: **9-1-2004**

Lecturer: **Udayan Kanade** . email: udayan@codito.com

ADSP Mailing List: adsp@codito.com

ADSP web site: www.codito.com/udayan/~adsp/

Acknowledgements: **Aditya Thakur**

Convoluted Ants:

We first solved the problem of "communist ants" : given 2 ant hills, each having some amount of sugar, the 'richer' ant hill gives 10% (arbitary communist rule) of the difference between the sugar levels to the other ant hill.

If we assume that the 2 hills can communicate and let each other know of their resp. sugar levels, the the solution is obvious: the hill with more sugar calculates the difference using the 'arithmetic ants' and sending it to the other hills.

Assuming the ant hills can't communicate (because say, the hills are a great distance from each other) or the 'arithmetic ants' have gone on vacation (hence the difference cannot be calculated), brings us to the more interesting solution (suggested by the 'dsp ants') . If each ant hill sends 10% of its sugar to the other hill, then the problem is solved.

Example:	A	B
Original:	20	30
Amount sent:	2	3
Final Amount:	21	29

Yippeeee !!

This can be generalized to 'n' ant hills, each ant hill have 'm' neighbour hills and a corresponding rule to distribute sugar to each of its neighbours.

We immediately notice that this method is "scatter convolution". Wow! , a true natural wonder (recent reports tell us that the 'dsp ants' were handsomely rewarded).

Taking a hint from their communist friends, the capitalist ants also devised a similar solution for their problem: each ant hill is supposed to steal from its neighbour hill. Thus, each hill goes to its neighbour hill and steals a certain amount (governed by the capitalist rule) and collects and adds all the loot at the end of the day.

From this analogy it is easy (and fun) to see that the scatter and gather are equivalent if we 'flip' the kernel (extra brownie points for anyone who finds the origin of the word ‘kernel’) i.e. suppose the communist ants give 10% to left hill and 20% to right hill and keep the rest, this is equivalent to the capitalist ants stealing 10% from their right hill and 20% from their left hill.

Properties:

Next we reviewed some of the properties of convolution using the ant analogy:

1) Commutative: If we swap the communist rule with the original sugar amounts and perform the same scatter operation, then the result obtained will be the same. This point brings us to the reason why the scatter approach is used to define convolution: the scatter convolution is commutative as opposed to the gather.

   A non-rigorous proof for the latter follows:
   Assuming that scatter operation is commutative and represented by *,
   We prove that the gather operation (represented by #) is not.

   $$\begin{aligned} A\#B &= A * \mathrm{flip}(B) \\ &= \mathrm{flip}(B) * A \\ &= \mathrm{flip}(B) \# \mathrm{flip}(A) \\ &\neq B\#A \end{aligned}$$

   QED
2) Associative: Suppose we use rule B on the first day and then apply rule C to the result obtained , the final result obtained at the end of the 2^{nd} day is the same as using the dsp ants to apply the rule B to rule C on the 1^{st} day and then applying this combined rule to the original amount on the 2^{nd} day.
   i.e. $(A*B)*C = A*(B*C)$ where A is the original amount.

   Combining the first 2 rules allows us to write the expression A*B*C or B*C*A (or any such permutation) and they all represent the same result. This a prime example of mathematical symbols hiding the beauty of the underling operation.

3) Identity: The identity in this case is each ant hill keeping all the sugar for itself.

4) Linearity: $k(A*B) = kA*B$ and $A*(b+c) = A*b + A*c$

Strength Matters!

We define the strength operator on a vector as
$S(A) = \Sigma a_i$

Things you need to know about strength:

1) It’s a field.
2) It’s a scalar function (maps a vector to a scalar).
3) It’s linear.
4) It isn’t shift invariant. This is because for an operation to be shift invariant both the input and the output should have the same number of components. In the case of the strength operator, since the output is a scalar, it’s not shift invariant.
5) Strength Theorem:
   $S(f*x) = S(f)\, S(x)$

   This shows us that if the kernel adds to 1, then the convolution operator conserves strength.
   But if this not true, say the kernel strength is less than 1 (to compensate for loss during the transportation of the sugar due to hungry ants), then after the scatter operation the strength of the resultant sugar will be less than that of the original , and the strength theorem tells us how much was the loss.

When convolution goes bad: see deconvolution (seedy convolution!) :

Deconvolution is when, given the signal and the output, we have find out the response of the system or equivalently given the output and the response of the system, we have to figure out the input.

Real-world applications of deconvolution:

Examples when you are given the output and the response:

1) given a ‘blurred’ image, you obtain the blurring kernel, and using deconvolution, get the original un-blurred image. Such a technique is commonly used to correct the defects (due to the curvature) in telescope lens.

Examples when you are given the output and the input:

2) Pre and post equalization is used in the telephone system to compensate for the distortion caused by the noise in the channel (found out by using deconvolution) .
3) To compensate for the response in large theaters the sound output is modified.
4) Deconvolution can be used to get better quality pirated VCDs !

Example showing procedure for deconvolution:

Output : 5 10 20 100 50 20 10
Kernel: 0.1 0.8 0.1

Assuming that the system is causal,

$5 = 0*0.1 + 0*0.8 + 0.1*x = 0.1*x$
Thus, $x0 = 50$.

$10 = 0*0.1 + 0.8*x0 + 0.1*x1 = 0.8*50 + 0.1*x1$
$x1 = 10 - 0.1*x1 - 0.8*x0 = -300$

Generalizing,

$x = y - f\,D(x)$
where y is the current output,
f is the kernel function
D(x) is delay x

Taking Z-transform ,
$X = Y - z^{-1} F X$
$X = Y / (1 + z^{-1} F)$

This equation shows us that deconvolution follows the feedback network.
Further given a FIR convolution system the corr. deconvolution is a simple IIR system (all pole system) with the zeroes getting mapped to the poles.
Thus, though a FIR system is stable, its inverse IIR system can be unstable if the pole lies outside the unit circle.