21M.361 Composing with Computers I (Electronic Music Composition)
Spring 2008

21M.361: Composing with Computers I (Electronic Music Composition)

Peter Whincop

Spring 2008 OCW

[All lab notes are being significantly revised for next term to reflect upgrades in software, different organization, and the incorporation of my own notes on DSP.]

Lab 3.1: Peak Opening, Saving, Importing; Basic Editing; Regions; DSP; Convolution

Peak Basics:

1. Peak is a stereo audio editor, and in general a far simpler program than Pro Tools. I personally use it only for convolution, and these days, less so, since there is a free/shareware way of doing it on a Mac (using Audacity and Soundhack). Convolution is what we'll eventually be focusing on in this module. Peak is not a mixer, but you can have numerous soundfiles open at a time. The clipboard is shared among the files, but undos (command-z) are specific to each soundfile. You can rotate between soundfiles using command-` (just like on a Mac or PC).

2. Make sure no other audio app is open when you start Peak. Check the Digidesign soundcard is being used by Peak by going Audio>Sound Out>CoreAudio..., and click on Hardware Settings.... Make sure both input and output devices are set to Digidesign HW. If this is not an option, then some other app must have grabbed the soundcard. So quit that app (or the Digidesign Core Manager), and start Peak again.

3. To open a document, use command-o, File>Open, or the leftmost icon on the toolbar. (If you can't see the toolbar, select it under the Windows menu.) The file menu has a list of the most recently used documents.

4. Peak is a destructive editor, meaning that saving a document writes over the previous. In Pro Tools, saving saves a session, and doesn't mess with the files, but in Peak it does. So, when you open a document, immediately Save As (shift-command-s, or File>Save As).

5. Just like in Pro Tools, where you should label files and regions descriptively, do the same in Peak, or keep a written list of what your soundfiles are, how you arrived at them, etc.

6. There are two views in Peak: the upper one is a view of the whole soundfile. You cannot select regions in this, but placing the cursor in it will start playing from there, and move the view frame—the white frame in the upper view—to the appropriate place. The lower view is the detail of what is in the view frame and it is here you do your editing. In the lower view, placing the cursor above the middle of the left channel will add a tiny 'L', and only the left channel will be selected when selecting. Idem, right channel. This is a fairly useless feature as far as I can tell.

7. There is a volume control, barely noticeable, on the transport. (If you can't see the transport, select it under the Windows menu.)

8. You can select just like in Pro Tools; shift-click will extend the boundaries. Command-a selects all. Sometimes you might think you've just placed the cursor somewhere, but in fact you have selected a tiny region. You will know this because playback will be just a blip. You can scroll in the usual way, or using control-left or right.

9. Use the transport controls on the toolbar or on the transport window to play etc., or use the space bar to play and pause, and return to return to zero. If nothing is selected, playback will begin at the cursor, and play until the end of the file; if a region is selected, only that region will play, and the region will remain intact after playback (unlike in Pro Tools). Playback with resume at the beginning of the selection next time you hit play.

10. You can create a named region using shift-command-r (or clicking on the appropriate icon, to the left of the stop button on the toolbar (not the transport). This region information is saved with the soundfile. Similarly, you can place markers using the icon to the left of the region, or by typing command-m. In both cases, you can move them either by hand or by double-clicking on their triangle. Peak tries to guess what you want a region or marker to be called.

11. You can create a new document from selected material (which may or may not be a named region) using control-n (NB. not command-n) or File>New>Document from Selection. It will be untitled, so save it. If you create a new document from an entire region, the region markers will be copied to the new document, and a plain marker will be left in the original document, with the region's name.

12. Windows>Contents will bring up a little window with all open documents listed. Clicking on their triangles will list their regions; double-clicking on one will highlight that region.

13. You can change the horizontal scale with the circled + and - buttons on the tool bar, or by using control-[or], just as in Pro Tools. Control-up or down alters the vertical scale; when it is normal, you should see 100–0–100 to the left of the waveform.

14. There are a number of other ways of navigating using the keyboard, most of which can be found under the Action menu. Fit selection to screen, use shift-command-]. Zoom all the way out use shift-command-[. Zoom at sample level use shift-left and zoom to sample level (end) shift-right.

15. The usual cut, copy, delete (like cut, but doesn't write over the clipboard), and paste (which inserts, not writes over, unless you have selected a region) operations work.

16. Under the Action menu is Go To. Typing the left arrow takes the cursor to the start of the selection, idem for the right. Command-g allows you to set a time-point for the cursor. You can also take the cursor to any named locations, such as region and loop endpoints, and markers. Speaking of which, you can convert a region or selection into a loop (for looped playback) using shift-command-- (minus). Look up nudging.

17. Under the File menu is Import Dual Mono. This would be used to import .L and .R pairs generated by Pro Tools. If you Save a Dual Mono file, it will just write over those two mono files; you have to use Save As to save it as a stereo file. When importing, select the left soundfile, and it will find the right one automatically. (If necessary, you can uncheck Auto Import from Dual Mono under the Options menu, but if you do, please return it to checked.)

18. Peak will import mp3s; just Open them as you would a normal soundfile. Peak can also read straight from audio CDs. It does not write mp3s, but can write CDs from Playlists, which we haven't covered here.

DSP (Digital Signal Processing) Basics:

1. Any DSP operation is done to the selected region; if no region is selected, the operation is done to the entire soundfile.

2. Change Pitch... is obvious; don't forget that Preserving Duration will have the advantage of not altering the length of the region, but the sound quality will be fairly poor if the interval is wide.

3. Change Gain...: check with clipguard before increasing gain. +6dB means x2 amplitude (in crude terms); it is a logarithmic scale.

4. Mix...: Copy a region to the clipboard, place the cursor at the point of the (same or a different) soundfile where you want the contents of the clipboard to be mixed, and select DSP>Mix.... The percentage is not clearly explained in the little window; 50% means the mix will be 1/2 (clipboard)–1/2 (selection), and 33% means the mix will be 1/3 (clipboard)–2/3 (selection). Selecting 100% is the same as pasting without inserting, i.e., writing over.

Convolution:

I explained what this is in lab. To carry it out in Peak, copy the impulse region to the clipboard, and select the region (or all/none) of the (same or different) soundfile you want to convolve with it. There is no dialog window for convolution. It is an expensive operation, so convolving a minute-long sample with a 20-minute one might take a long time, or cause the computer to run out of memory, or even freeze or crash Peak. Typically, for our misusing purposes, we would save up to 10s to the clipboard to convolve with a soundfile of any length.

Convolution basically imparts the frequency characteristics, amplitude envelope, and reverberant properties of a short 'impulse' region onto a longer region. Or, the frequencies (or bands of frequencies) of two sounds, over time, are reinforced if they are shared, and diminished if they are not so shared. It's a little like old-fashioned vocoding, the technique used to make robot sounds in 60s and 70s movies. If I have, say, the opening E-flat major chord of Beethoven's Eroica symphony, and convolve it with me speaking, it will sound as if I am speaking Eroica chords. There are enough frequencies in common (and noise) for it to be a successful convolution. (Voice can be thought of as shaped noise.)

Vocoding would take, say, the generic robot sound, or the Eroica chord, and pass it through a bank of, say, twelve bandpass filters. The energy of the resulting filtered signal would then be used to drive the bandpass/reject filtered frequency bands of, say, my voice. So the frequencies in my voice that the generic sound also had would come through, and those not in common wouldn't. Convolution effectively uses 4096, or some other large number, of equally-spaced frequency bands to do more or less the same thing, not just twelve crudely shaped filters

More technically, convolution can be understood both in the time domain—what we are more familiar with, the amplitude (energy) of a signal over time, in other words, the waveform—and in the frequency domain—the Fourier transform of the time domain representation. The Fourier transform is the spectral content of the sound, the energy of each frequency (band), at every (pre-determined and equal) slice of time. We are working with sampled information,

so our data is discrete.

I've included below excerpts from an excellent book by Garreth Loy, named Musimathics (what an annoying title), vol. 2. It explains how, in the time domain, convolution—which is a familiar mathematical operation—builds an array of the two sample sets multiplied and added. Effectively, the short (impulse) sample is replicated at every sample point of the long sample, scaled by the amplitude of long sample's sample (the word 'sample' is getting confusing now), and placed at that sample's position. See the simple diagram below, taken from the Csound Book. It shows how delay, echo, and even reverb, can be described using convolution. (In general, reverb is achieved using allpass filters—filters that pass all frequencies through, but with different phases, which effectively reinforces patterns of frequencies through constructive and destructive interference—but that's a different story.) The Musimathics excerpt show how the discrete convolution formula—a kind of 0,5–1,4–2,3–3,2–4,1–5,0 type of scheme—which does not suggest at first sight the kind of array I described, connects with the replication of the short sample scaled at every sample point of the long sample. Convolution is commutative, but we always treat the small sample as the impulse, the one that scales and displaces copies of the long soundfile.

Convolution used as reverb involves recording a delta (a click, a single amplitute=1 at time=0, and 0 everywhere else) in a space, say a concert hall. Though this might not seem intuitive, a delta is actually an instance of white noise, thus containing all frequencies at equal energy. So the frequencies recorded would be those resulting from reflections (echoes) in the space, and the resulting sound is the impulse for a reverb. Reverberation impulses are fairly complex, and would be near impossible to create them in the time domain. They can be created algorithmically, though. Convolve such an impulse with my voice recorded in a clean (almost anechoic space) and it will sound as if I am talking in that reverberant space.

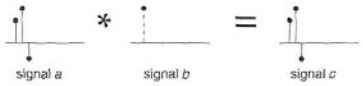

Figure 26.7 Convolution of a signal (*a*) with a single impulse (*b*).

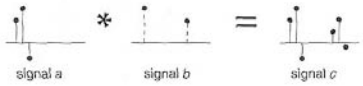

Figure 26.8 Convolution of a signal (*a*) with two widely spaced impulses (*b*).

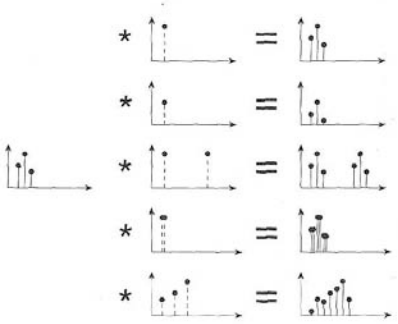

Figure 26.9 Convolution of a signal (*a*) with a variety of impulses (*b*).

Source: *The Csound Book: Perspectives in Software Synthesis, Sound Design, Signal Processing, and Programming*. Edited by Richard Boulanger. Cambridge, MA: MIT Press, 2002. ISBN: 9780262522618.

4 Convolution

It is not surprising that the greatest mathematicians have again and again appealed to the arts in order to find some analogy to their own work. They have indeed found it in the most varied arts, in poetry, in painting, and in sculpture, although it would certainly seem that it is in music, the most abstract of all the arts, the art of number and of time, that we find the closest analogy.
—Havelock Ellis

4.1 Rolling Shutter Camera

Convolution lies at the heart of modern digital audio. The quickest and most intuitive introduction to convolution that I've found comes by way of an antique rolling shutter camera. When photography was first being developed early in the twentieth century, some cameras used a rolling shutter instead of an iris to control exposure of the film. A narrow slit was cut in a roll of opaque paper attached between two spring-loaded rollers that when released by the camera's trigger caused the slit to scroll quickly across the film plate between the lens and the film, exposing it to light and registering the image on the film (see figure 4.1).

Because the travel time of the shutter was not instantaneous, photographing fast-moving images led to image-skewing motion artifacts. In the famous photograph shown in figure 4.2, the slit moved from bottom to top while the camera operator swiveled to track the car going by. Because of motion artifacts and other problems, the rolling shutter was eventually supplanted by the iris shutter.

How is this relevant? The rolling shutter camera implemented a form of convolution. Suppose we had such a camera today and that we altered the loop of opaque paper so that there were two narrow slits, one some distance from the other. Now we retake the picture of the car driving horizontally past the camera. As the first slit passes across the film plate, it registers the car's image at one location on the film, and as the second slit passes over the film plate, it registers the car's image at a position proportional to how much farther the car has traveled in the meantime. The film registers the sum of the first image of the car and the second (time-shifted) image of the car—a double exposure with the exposures very close together in time. The slits are effectively sampling and time-shifting the image, depending upon when they pass between the film and the lens. This is basically what convolution is all about: time shifting and combining signals.

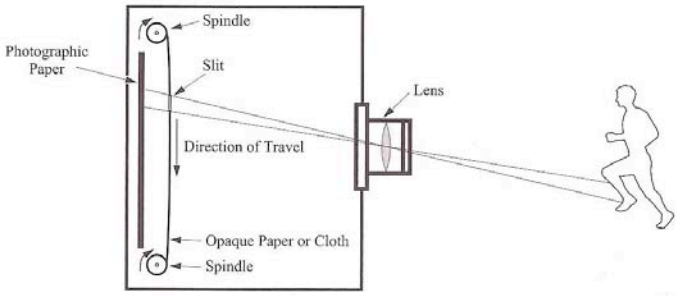

Figure 4.1
Camera with a slit to expose film.

Figure 4.2
"Voyage en auto; Papa à 80 km à l'heure, mars 1913." Photograph by Jacques Henri Lartigue. © Ministère de la Culture—France/AAJHL.

Loy, Gareth. *Musimathics*. Vol. 2. Cambridge, MA: MIT Press, 2007. ISBN: 9780262122856.

4.2 Defining Convolution

The formula for *convolution* is

$$h(n) = \sum_{m=0}^{n} f(m)g(n-m), \qquad \textit{Convolution} \quad (4.1)$$

where m and n are integers. The function g can be likened to the slit in the rolling shutter camera, and the function f can be likened to the image being recorded. The convolution operation scrolls function g past function f rather like the slit is scrolled past the film plate in the rolling shutter camera.

Let's try an example to see how equation (4.1) works. Say that the length of two functions f and g are both $N=4$, so that their defined values lie in the range of 0 to $N-1$. For this example, the actual defined values of the functions don't matter because we're just trying to get a feel for the abstract pattern of the convolution operation. Let's also say that values lying outside of the defined range of functions f and g are equal to zero. We can specify this mathematically for function f by writing

$$f(n) = \begin{cases} 0, & n \geq N, \\ f(n), & 0 \leq n < N, \\ 0, & n < 0. \end{cases} \qquad \textit{Convolution Range Rule} \quad (4.2)$$

Equation (4.2) says that values of f outside the defined range $0 \leq n < N$ are equal to zero, and values within this range are its defined values. We must also apply the convolution range rule to function g.

We start off by seeing what the convolution operation would calculate for $N=0$, then continue by setting n to larger and larger values. Computing the first few elements of equation (4.1) for increasing n we have

$$n = 0 \qquad h(0) = f(0)g(0)$$

$$n = 1 \qquad h(1) = f(0)g(1) + f(1)g(0)$$

$$n = 2 \qquad h(2) = f(0)g(2) + f(1)g(1) + f(2)g(0)$$

$$n = 3 \qquad h(3) = f(0)g(3) + f(1)g(2) + f(2)g(1) + f(3)g(0)$$

Note that as n increases, convolution recruits increasing numbers of terms from f and g into the equation. Here's a visualization: suppose we mark out the two functions on the sides of two boards (figure 4.3), separated by a roller. The figure illustrates the computation of h for $n=3$. As the boards roll past each other, we form the product of f and g, and then sum the products.

Loy, Gareth. *Musimathics*. Vol. 2. Cambridge, MA: MIT Press, 2007. ISBN: 9780262122856.

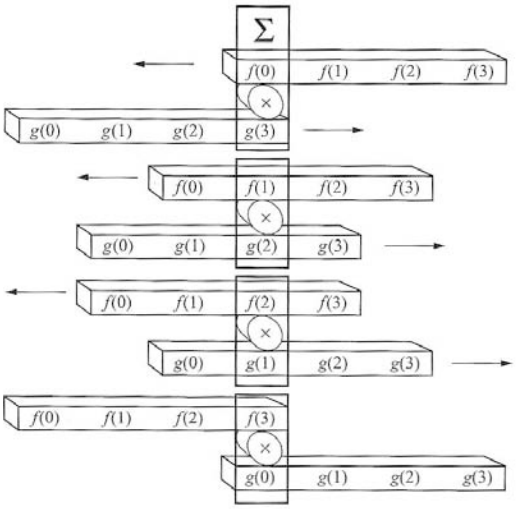

Figure 4.3
Convolution step $h(3)$.

But what happens when $n = 4$ or more? Since this is beyond the range of the functions, we invoke the convolution range rule and substitute zero for undefined values, which yields

$$
\begin{aligned}
n = 4 \quad h(4) &= f(0)g(4) + f(1)g(3) + f(2)g(2) + f(3)g(1) + f(4)g(0) \\
&= (f(0) \cdot 0) + f(1)g(3) + f(2)g(2) + f(3)g(1) + (0 \cdot g(0)) \\
&= f(1)g(3) + f(2)g(2) + f(3)g(1) \\
n = 5 \quad h(5) &= f(0)g(5) + f(1)g(4) + f(2)g(3) + f(3)g(2) + f(4)g(1) + f(5)g(0) \\
&= f(2)g(3) + f(3)g(2) \\
n = 6 \quad h(6) &= f(0)g(6) + f(1)g(5) + f(2)g(4) + f(3)g(3) + f(4)g(2) + f(5)g(1) + f(6)g(0) \\
&= f(3)g(3) \\
n = 7 \quad h(7) &= f(0)g(7) + f(1)g(6) + f(2)g(5) + f(3)g(4) + f(4)g(3) + f(5)g(2) + f(6)g(1) \\
&\quad + f(7)g(0) \\
&= 0
\end{aligned}
$$

For all values $n \geq 7$, $h(n) = 0$, so we might as well stop at $n = 6$.

In general if the length of f is N_f and the length of g is N_g, then the length of the convolved function h is $N_h = N_f + N_g - 1$. This rule works even when $N_f \neq N_g$.

As a shorthand for equation (4.1), mathematicians notate convolution this way:

$$h(n) = f(\cdot) * g(\cdot), \qquad \textit{Convolution Operator} \quad (4.3)$$

where $*$ is the convolution operator,[1] and the notation $f(\cdot)$ and $g(\cdot)$ can be translated to mean *all required values* of functions f and g. The centered dot notation, $f(\cdot)$ and $g(\cdot)$, is used in equation (4.3) to allow the convolution operation unlimited indexing range. Since $f(\cdot) * g(\cdot)$ is itself a function of n, sometimes convolution is written $(f * g)(n)$.

4.3 Numerical Examples of Convolution

The following numerical examples will be a lot clearer if the format for representing the steps of convolution is rotated to show terms that are being summed in columns, as in traditional arithmetic. Here is the convolution of functions f and g:

$f(0)g(0)$	$f(0)g(1)$	$f(0)g(2)$	$f(0)g(3)$	...	...	...	...
	$f(1)g(0)$	$f(1)g(1)$	$f(1)g(2)$	$f(1)g(3)$	...	...	...
		$f(2)g(0)$	$f(2)g(1)$	$f(2)g(2)$	$f(2)g(3)$	...	...
			$f(3)g(0)$	$f(3)g(1)$	$f(3)g(2)$	$f(3)g(3)$	...
			...	...	...	...	...
$h(0)$	$h(1)$	$h(2)$	$h(3)$	$h(4)$	$h(5)$	$h(6)$	...

In this format, increasing values of n run across the page instead of down the page. One value of the output function h is formed by summing each column. If the product of two functions would yield zero because of the application of the convolution range rule, the cell is marked with ellipses (. . .) to indicate that its calculation is skipped.

Here is a concrete example of convolution.

$$h(n) = f(\cdot) * g(\cdot) = \{1, 2, 3, 4, 5\} * \{1, 2, 3\}. \quad (4.4)$$

	1	2	3				
		2	4	6			
			3	6	9		
				4	8	12	
					5	10	15
$h(n) =$	1	4	10	16	22	22	15

Here is the same example with the terms f and g transposed.

$$h(n) = g(\cdot) * f(\cdot) = \{1, 2, 3\} * \{1, 2, 3, 4, 5\}. \tag{4.5}$$

	1	2	3	4	5			
		2	4	6	8	10		
			3	6	9	12	15	
$h(n) =$	1	4	10	16	22	22	15	

Notice that equations (4.4) and (4.5) produce the same result, indicating that

Convolution is commutative.

4.3.1 Impulse Function

The next pair of examples reveals a useful insight into convolution. The *unit impulse function* $\delta(n)$ is defined as a single 1 located at $n = 0$, and all other indexed values are 0, written as follows:

$$\delta(n) = \begin{cases} 1, & n = 0, \\ 0, & n \neq 0, \end{cases} \qquad \textit{Unit Impulse Function} \tag{4.6}$$

where n is an integer.

In the first example, let $f(n) = \{1, 2, 3, 4, 5\}$, and let $g(n)$ be the unit impulse function $\delta(n)$. Then the convolution of $f(n)$ and $g(n)$ is as follows:

$$h(n) = f(\cdot) * g(\cdot) = \{1, 2, 3, 4, 5\} * \{1, 0, 0, 0, 0\}. \tag{4.7}$$

1	0	0	0	0				
	2	0	0	0	0			
		3	0	0	0	0		
			4	0	0	0	0	
				5	0	0	0	0
$h(n) = 1$	2	3	4	5	0	0	0	0

As we see, equation (4.7) simply copies f to h unchanged. In the next example, let $g(n) = \{0, 1, 0, 0, 0\}$. We have shifted the location of the 1 in the impulse function one element to the right.

$$h(n) = f(\cdot) * g(\cdot) = \{1, 2, 3, 4, 5\} * \{0, 1, 0, 0, 0\}. \tag{4.8}$$

0	1	0	0	0	0				
		2	0	0	0	0			
			3	0	0	0	0		
				4	0	0	0	0	
					5	0	0	0	0
$h(n) = 0$	1	2	3	4	5	0	0	0	0

Equation (4.8) copies f to h shifted right by one place. Last, we scale the impulse function g by 1/2.

$$h(n) = f(\cdot) * g(\cdot) = \{1, 2, 3, 4, 5\} * \{0, 0.5, 0, 0, 0\}. \tag{4.9}$$

0	0.5	0	0	0	0				
		1	0	0	0	0			
			1.5	0	0	0	0		
				2	0	0	0	0	
					2.5	0	0	0	0
$h(n) = 0$	0.5	1	1.5	2	2.5	0	0	0	0

Convolving a function f with an impulse function just copies f.

Convolving f with a shifted impulse function makes a shifted copy of f.

Convolving f with a scaled impulse function scales f.

Convolving f with a scaled, shifted impulse function scales and shifts f.

Figures 4.4–4.9 illustrate convolutional shifting for some interesting wave types.

$f(x) * g(x) = h(x)$

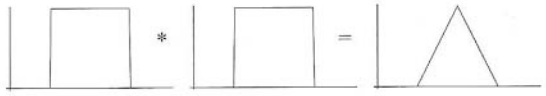

Figure 4.4
Convolution of two rectangular windows.

$f(x) * g(x) = h(x)$

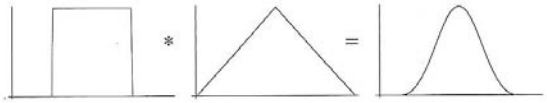

Figure 4.5
Convolution of window with triangular function.

$f(x) * g(x) = h(x)$

Figure 4.6
Convolution of exponential decay with rectangular window.

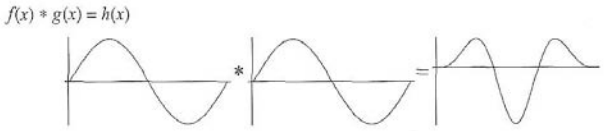

Figure 4.7
Convolution of two sine waves.

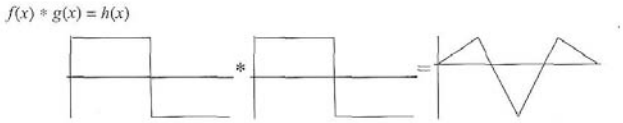

Figure 4.8
Convolution of two square waves.

Figure 4.9
Convolution of impulse train with exponential decay.

4.3.2 Impulse Train

Consider the following example. An *impulse train function* is a periodic impulse function with period equal to the distance between impulses. Let f be an impulse train function with a period of 2, $f = \{1, 0, 1, 0, 1\}$, and convolve it with $g = \{1, 2, 3, 4, 5\}$:

$$f(\cdot) * g(\cdot) = \{1, 0, 1, 0, 1\} * \{1, 2, 3, 4, 5\}. \tag{4.10}$$

1	2	3	4	5				
	0	0	0	0	0			
		1	2	3	4	5		
			0	0	0	0	0	
				1	2	3	4	5
1	2	4	6	9	6	8	4	5

The impulse train function adds a shifted copy of the signal for each impulse in the impulse train function (see figure 4.9). Recalling the rolling shutter camera example, when we added an extra

slot to the opaque screen in the camera, we changed its function from a single impulse to an impulse train (with two pulses) and the camera produced the sum of the original and the delayed image, analogous to equation (4.10).

Let's modify the impulse train function so that each successive impulse is half its previous value: $f = \{1, 0, 0.5, 0, 0.25\}$. Then

$$f(\cdot) * g(\cdot) = \{1, 0, 0.5, 0, 0.25\} * \{1, 2, 3, 4, 5\}. \tag{4.11}$$

1	2	3	4	5				
	0	0	0	0	0			
		0.5	1	1.5	2	2.5		
			0	0	0	0	0	
				0.25	0.5	0.75	1	1.25
1	2	3.5	5	6.75	2.5	3.25	1	1.25

The scaled impulse train function sums a scaled and shifted copy of each impulse in the impulse train function.

Looking at this result, we can see why convolution is sometimes described as a smearing operation: by summing multiple delayed copies, convolution can render the original pattern less distinct. In fact, our ears are highly sensitive to this smearing effect: the walls of a room transmit delayed and scaled reflections to our ears from a sound source, so reverberation is a form of convolution. If we snap our fingers or create another impulsive sound in a room, the sum of all delayed and scaled reflections is called the room's *impulse response*. Speech is more intelligible in a bedroom than in a cathedral because the impulse response of a bedroom is shorter and less pronounced than the impulse response of a cathedral; hence there is less acoustical smearing in a bedroom.

Convolving with an impulse train adds a shifted original to the output for each impulse.

As many copies will be superimposed as there are nonzero impulses.

Copies will be shifted (delayed) by the position of the impulse in the impulse train function.

Shifted copies will be scaled by the amplitude of the impulses.

Shifted, scaled copies of the original are summed, smearing the result.

4.3.3 Convolution as Echo and Reverberation

Echoes are scaled and delayed copies of a source signal, and reverberation is a set of echoes generated recursively. So echoes and reverberation can be seen as forms of convolution (see volume 1, section 7.13).

If we convolve the impulse response of a good-sounding hall with a sound recorded elsewhere, it will be as if the sound had been recorded in that hall. Mathematically, the process is like equation (4.11): many scaled time-delayed copies of the sound are summed, based on the impulse

response of the hall. So, convolving the impulse response of a room with another time domain signal applies the room's response to that signal. Thus, if one has the impulse response of a good concert hall, one can process recordings made in a nonreverberant room to sound as if they had been recorded in the concert hall.

This procedure works best if the sound being convolved is fairly dry (that is, with a short, mild impulse response), but it works surprisingly well even if the source signal is reverberant. Unfortunately, the convolution operation is computationally intensive, so using convolution to create artificial reverberation is prohibitively expensive on all but the fastest computers.[2] For more cost-effective approaches to creating artificial reverberation, see section 9.6.