9.2 Linear shift-invariant filters

9.2.1 Linearity

Linear operators are defined by the *principle of superposition*. If a and b are two complex-valued scalars, and $\mathcal{H}$ is an operator that maps an image onto another image of the same dimension, then the operator is linear if and only if

$$\mathcal{H}(a:+b:) = a\mathcal{H}:+b\mathcal{H}: \tag{9.6}$$

We can generalize Eq. (9.6) to the superposition of many inputs

$$\mathcal{H}\left(\sum_k a_k:\right) = \sum_k a_k \mathcal{H}: \tag{9.7}$$

The superposition property makes linear operators very useful. We can decompose a complex image into simpler components for which we can easily derive the response of the operator and then compose the resulting response from that of the components.

It is especially useful to decompose an image into its individual pixels as discussed in Section 8.5.1.

9.2.2 Shift invariance and convolution

Another important property of an operator is shift invariance or homogeneity. It means that the response of the operator does not explicitly depend on the position. If we shift a signal, the output image is the same but for the shift applied. We can formulate this property more elegantly with a *shift operator* S. For 2-D images, for example, the shift operator is defined as

$${}^{mn}SG_{m'n'} = G_{m'-m,n'-n} \tag{9.8}$$

An operator is then *shift-invariant* if and only if it commutes with the shift operator, that is,

$$\mathcal{H}S = S\mathcal{H} \tag{9.9}$$

Note that the shift operator S itself is a shift-invariant operator. An operator that is both linear and shift-invariant is known as a *linear shift-invariant operator* or short *LSI* operator. This important class of operators is also known as *linear time-invariant* or *LTI* operators for time series.

It can be proven [1] that a linear shift-invariant operator must *necessarily* be a convolution operation in the space domain. There is *no* other operator type that is both linear and shift-invariant. Thus, linear

shift-invariant neighborhood operators share all the useful features of convolution that were discussed in Section 8.6.3. They are commutative, associative, and distribute over addition (see also Eq. (9.5)). These properties are very useful for an efficient design of filter operations [CVA2, Chapter 6].

9.2.3 Point spread function

As just discussed in the previous section, an LSI filter can be represented in the space domain as a convolution operation. In two dimensions the image $\boldsymbol{G}$ is convolved with another image $\boldsymbol{H}$ that represents the LSI operator:

$$G'_{mn} = \sum_{m'=0}^{M-1} \sum_{n'=0}^{N-1} H_{m',n'} G_{m-m',n-n'} \tag{9.10}$$

Because for a neighborhood operation $\boldsymbol{H}$ is zero except for a small neighborhood, this operation can also be written as

$$G'_{mn} = \sum_{m'=-R}^{R} \sum_{n'=-R}^{R} H_{-m',-n'} G_{m+m',n+n'} \tag{9.11}$$

In this equation it is assumed that coefficients of $\boldsymbol{H}$ are nonzero only in a (2R+1)×(2R+1) window. Both representations are equivalent if we consider the periodicity in the space domain (Section 8.7.1). The latter representation is much more practical and gives a better comprehension of the operator. For example, the following M×N matrix and 3×3 filter mask are equivalent:

$$\begin{bmatrix} \mathbf{0.} & -1 & 0 & \dots & 0 & 1 \\ 1 & 0 & 0 & \dots & 0 & 2 \\ 0 & 0 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 0 \\ -1 & -2 & 0 & \dots & 0 & 0 \end{bmatrix} \equiv \begin{bmatrix} 0 & -1 & -2 \\ 1 & \mathbf{0.} & -1 \\ 2 & 1 & 0 \end{bmatrix} \tag{9.12}$$

For a D-dimensional signal, the convolution sum can be written with a simplified vector indexing as also used in Section 8.4.4:

$$G'_{\boldsymbol{n}} = \sum_{\boldsymbol{n'}=-\boldsymbol{R}}^{\boldsymbol{R}} H_{-\boldsymbol{n'}} G_{\boldsymbol{n}+\boldsymbol{n'}} \tag{9.13}$$

with $\boldsymbol{n} = [n_1, n_2, \dots, n_D]$, $\boldsymbol{R} = [R_1, R_2, \dots, R_D]$, where $G_{\boldsymbol{n}}$ is an element of a D-dimensional signal $G_{n_1,n_2,\dots,n_D}$. The notation for the sums in this

equation is an abbreviation for

$$\sum_{\boldsymbol{n}'=-\boldsymbol{R}}^{\boldsymbol{R}} = \sum_{n'_1=-R_1}^{R_1} \sum_{n'_2=-R_2}^{R_2} \cdots \sum_{n'_D=-R_D}^{R_D} \tag{9.14}$$

The vectorial indexing introduced here allows writing most of the relations for arbitary dimensional signals in a simple way. Moreover, it can also be used for skewed coordinate systems if $\boldsymbol{n}$ are regarded as the indices of the corresponding *lattice vectors* (see Eq. (8.27), Section 8.4.2).

The filter mask is identical to another quantity known as the *point spread function*, which gives the response of the filter to a point image:

$$P'_{\boldsymbol{n}} = \sum_{\boldsymbol{n}'=-\boldsymbol{R}}^{\boldsymbol{R}} H_{\boldsymbol{n}'} P_{\boldsymbol{n}-\boldsymbol{n}'} = H_{\boldsymbol{n}} \tag{9.15}$$

where

$$P_{\boldsymbol{n}} = \begin{cases} 1 & \boldsymbol{n} = \boldsymbol{0} \\ 0 & \text{otherwise} \end{cases} \tag{9.16}$$

The central importance of the point spread function is based on the fact that the convolution operation is linear. If we know the response to a point image, we can compute the response to any image, as any image can be composed of point images as discussed in Section 8.5.1. With respect to the analysis of time series, the point spread function is known as the *impulse response*, with respect to the solution of partial differential equations as the *Green's function* [2].

9.2.4 Transfer function

The Fourier transform of the convolution mask is known as the *transfer function* of a linear filter. The transfer function has an important practical meaning. For each wave number, it gives the factor by which a periodic structure is multiplied using the filter operation. This factor is generally a complex number. Thus, a periodic structure experiences not only a change in the amplitude but also a phase shift:

$$\begin{aligned} \hat{G}'_{\boldsymbol{v}} = \hat{H}_{\boldsymbol{v}} \hat{G}_{\boldsymbol{v}} &= r_H \exp(\mathrm{i}\varphi_H)\, r_G \exp(\mathrm{i}\varphi_G) \\ &= r_H r_G \exp[\mathrm{i}(\varphi_H + \varphi_G)] \end{aligned} \tag{9.17}$$

where the complex numbers are represented by their magnitude and phase as complex exponentials.

Using the wave number normalized to the Nyquist limit (Eq. (8.34) in Section 8.4.2), the transfer function is given by

$$\hat{h}(\tilde{k}) = \sum_{n'=-R}^{R} h_{n'} \exp(-\pi \mathrm{i} n' \tilde{k}) \tag{9.18}$$

for a 1-D signal and by

$$\hat{h}(\tilde{\boldsymbol{k}}) = \sum_{\boldsymbol{n}'=-\boldsymbol{R}}^{\boldsymbol{R}} H_{\boldsymbol{n}'} \exp(-\pi \mathrm{i} \boldsymbol{n}'^T \tilde{\boldsymbol{k}}) \tag{9.19}$$

for a multidimensional signal. For a nonorthogonal, that is, skewed lattice, the vectorial index $\boldsymbol{n}'$ has to be replaced by the reciprocal lattice vector (Eq. (8.29)), and Eq. (9.19) becomes

$$\hat{h}(\boldsymbol{k}) = \sum_{\boldsymbol{v}=-\boldsymbol{R}}^{\boldsymbol{R}} H_{\boldsymbol{r}} \exp(-2\pi \mathrm{i} \hat{\boldsymbol{r}}_{\boldsymbol{v}}^T \boldsymbol{k}) \tag{9.20}$$

9.2.5 Symmetries

Symmetries play a central rule for linear shift-invariant filters in the processing of higher-dimensional signal processing. This is because of the simplified transfer function of symmetric masks. According to Section 8.6.3, filters of even and odd symmetry have a real and purely imaginary transfer function, respectively. The symmetry of a filter is most generally expressed by:

$$H_{\boldsymbol{R}-\boldsymbol{r}} = \pm H_{\boldsymbol{r}} \tag{9.21}$$

This is a necessary and sufficient condition for a real or imaginary transfer function. Filters normally meet a stronger symmetry condition for each direction d:

$$H_{r_1,\ldots,R_d-r_d,\ldots,r_D} = \pm H_{r_1,\ldots,r_d,\ldots,r_D} \tag{9.22}$$

For separable symmetric filters, the symmetry conditions can be expressed for each 1-D component separately:

$$h_{R_d-r_d} = \pm h_{r_d} \tag{9.23}$$

As the transfer functions of the 1-D components of separable filters are combined multiplicatively, an even and odd number of odd components results in an even and odd filter according to Eq. (9.21) and thus into a real and imaginary transfer function, respectively.

Because of the significance of separable filters for effective computing of convolution operations [CVA2, Section 5.6], we focus on the symmetry of 1-D filters. Besides odd and even symmetry, it is necessary to distinguish filters with an even and odd number of coefficients.

The situation is straightforward for filters with an odd number of coefficients. Then the central coefficient is the center of symmetry and the result of a filter operation is written for the position of this central coefficient. This symmetry is implicitly considered in Eqs. (9.13) and

(9.18) where the central coefficient has the index 0. With this indexing of the filter coefficients, the convolution sum and transfer function of even 1-D filters with $2R+1$ coefficients—also known as *type I FIR filter* [3]—can be expressed as

$$g'_n = h_0 g_n + \sum_{n'=1}^{R} h'_n (g_{n+n'} + g_{n-n'}), \quad \hat{h}(\tilde{k}) = h_0 + \sum_{n'=1}^{R} 2h_{n'} \cos(n'\pi\tilde{k}) \tag{9.24}$$

and for odd filters with $2R+1$ coefficients or *type III FIR filters* as

$$g'_n = \sum_{n'=1}^{R} h'_n (g_{n-n'} - g_{n+n'}), \quad \hat{h}(\tilde{k}) = \mathrm{i} \sum_{n'=1}^{R} 2h_{n'} \sin(n'\pi\tilde{k}) \tag{9.25}$$

For filters with an even number of coefficients, there is no central pixel. The symmetry center rather is inbetween two pixels. This means that the results of a filter operation with such a filter are to be placed on a grid that is shifted by half a pixel distance. Because of this shift between the output pixel and the input pixels, the transfer function of an even filter with $2R$ coefficients *type II FIR filter* is

$$\hat{h}(\tilde{k}) = h_0 + \sum_{n'=1}^{R} 2h_{n'} \cos((n'-1/2)\pi\tilde{k}) \tag{9.26}$$

The transfer function of an odd filter with $2R$ coefficients or *type IV FIR filter* is

$$\hat{h}(\tilde{k}) = \mathrm{i} \sum_{n'=1}^{R} 2h_{n'} \sin((n'-1/2)\pi\tilde{k}) \tag{9.27}$$

The equations for symmetric filters for two and more dimensions are significantly more complex and are discussed in Jähne [4].

9.2.6 LSI operators as least squares estimators

The LSI operators compute a new value at each point in a signal from a linear combination of neighboring points. Likewise, a least squares estimator computes the estimate of a quantity from a linear combination of the input values. Thus it appears that a close relationship should exist between LSI operators and least squares estimators.

We assume that we want to fit a certain function with linear parameters a_p

$$f(\boldsymbol{x}) = \sum_{p=0}^{P-1} a_p f_p(\boldsymbol{x}) \tag{9.28}$$

to the local spatial gray-value variation $g(\boldsymbol{x})$. For 2-D digital signals, the continuous functions $f_p(\boldsymbol{x})$ have to be replaced by matrices $\boldsymbol{F}_p$. All of the following equations are also valid for digital signals but it is more convenient to stay with the continuous case. In the least squares sense, the following error measure $e^2(\boldsymbol{x})$ should be minimized:

$$e^2(\boldsymbol{x}) = \int_{-\infty}^{\infty} w(\boldsymbol{x}') \left(\sum_{p=0}^{P-1} a_p(\boldsymbol{x}) f_p(\boldsymbol{x}') - g(\boldsymbol{x}+\boldsymbol{x}') \right)^2 \mathrm{d}^D x' \tag{9.29}$$

In this integral the window function $w(\boldsymbol{x}')$ has been introduced to limit the fit to a local neighborhood around the point $\boldsymbol{x}$. Therefore, the fit coefficients $a_p(\boldsymbol{x})$ depend on the position. Normally, the window function is an isotropic even function with a maximum at the origin monotonically decreasing with increasing distance from the origin. We further assume that the window function is normalized, that is,

$$\int_{-\infty}^{\infty} w(\boldsymbol{x}') \, \mathrm{d}^D x' = 1 \tag{9.30}$$

For the sake of simpler equations, the following abbreviations will be used in this section:

$$\begin{aligned} \langle f_p g_{\boldsymbol{x}} \rangle &= \int_{-\infty}^{\infty} w(\boldsymbol{x}') f_p(\boldsymbol{x}') g(\boldsymbol{x}+\boldsymbol{x}') \, \mathrm{d}^D x' \\ \langle f_p f_q \rangle &= \int_{-\infty}^{\infty} w(\boldsymbol{x}') f_p(\boldsymbol{x}') f_q(\boldsymbol{x}') \, \mathrm{d}^D x' \end{aligned} \tag{9.31}$$

Setting all derivatives of Eq. (9.29) with respect to the parameters $a_p(\boldsymbol{x})$ zero, the following linear equation system is obtained as the standard least squares solution of the minimization problem:

$$\boldsymbol{a}(\boldsymbol{x}) = \boldsymbol{M}^{-1} \boldsymbol{d}(\boldsymbol{x}) \tag{9.32}$$

with

$$M_{p,q} = \langle f_p f_q \rangle, \; \boldsymbol{a} = [a_0(\boldsymbol{x}), a_1(\boldsymbol{x}), \ldots, a_{P-1}(\boldsymbol{x})]^T \; d_p = \langle f_p g_{\boldsymbol{x}} \rangle$$

The solution of Eq. (9.32) becomes most simplistic if the functions $f_p(\boldsymbol{x})$ are orthogonal to each other, that is, $\langle f_p f_q \rangle = \langle f_p^2 \rangle \delta_{p-q}$. Then the matrix $\boldsymbol{M}$ is diagonal and

$$a_p(\boldsymbol{x}) = \langle f_p g_{\boldsymbol{x}} \rangle \Big/ \left\langle f_p^2 \right\rangle \tag{9.33}$$

This expression can also be written as a convolution integral by using Eq. (9.31) and substituting $\boldsymbol{x}'$ by $-\boldsymbol{x}'$:

$$a_p(\boldsymbol{x}) = \int_{-\infty}^{\infty} w(\boldsymbol{x}') f_p(-\boldsymbol{x}') g(\boldsymbol{x} - \boldsymbol{x}')\, \mathrm{d}^D x' \tag{9.34}$$

This means that the fit coefficient for each point is computed by convolving the windowed and mirrored orthonormal function with the signal.

Example 9.1: Plane fit

As a simple example we discuss the local plane fit, that is, the local approximation of the gray-scale variation by a plane. The fit function is

$$f(\boldsymbol{x}) = a_0 + a_1 x_1 + a_2 x_2, \quad f_0 = 1, f_1 = x_1, f_2 = x_2 \tag{9.35}$$

It is easy to verify that these three functions are orthogonal to each other. Therefore,

$$\begin{aligned} a_0 &= \int_{-\infty}^{\infty} w(\boldsymbol{x}') g(\boldsymbol{x} - \boldsymbol{x}')\, \mathrm{d}^D x' \\ a_1 &= -\int_{-\infty}^{\infty} w(\boldsymbol{x}') x_1' g(\boldsymbol{x} - \boldsymbol{x}')\, \mathrm{d}^D x' \Big/ \int_{-\infty}^{\infty} w(\boldsymbol{x}') {x_1'}^2\, \mathrm{d}^D x' \\ a_2 &= -\int_{-\infty}^{\infty} w(\boldsymbol{x}') x_2' g(\boldsymbol{x} - \boldsymbol{x}')\, \mathrm{d}^D x' \Big/ \int_{-\infty}^{\infty} w(\boldsymbol{x}') {x_2'}^2\, \mathrm{d}^D x' \end{aligned} \tag{9.36}$$

As a special case for 2-D digital signals we take a binomial 3×3 window and obtain

$$\begin{aligned} \boldsymbol{W} &= \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}, \qquad \boldsymbol{F}_0 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \\ \boldsymbol{F}_1 &= 2\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}, \quad \boldsymbol{F}_2 = 2\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \end{aligned} \tag{9.37}$$

The three matrices $\boldsymbol{F}_0$, $\boldsymbol{F}_1$, and $\boldsymbol{F}_2$ are already normalized, that is,

$$\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} W_{m,n} ((F_p)_{m,n})^2 = 1 \tag{9.38}$$

so that the division in Eq. (9.36) is not required. Then the convolution masks to obtain the fit coefficients a_0, a_1, and a_2 are

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}, \frac{1}{8}\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}, \frac{1}{8}\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \tag{9.39}$$

and we end up with the well-known binomial smoothing mask and the Sobel operator for the estimate of the mean and slopes of a local plane fit, respectively.

Thus, the close relationship between LSI operators and least squares fits is helpful in determining what kind of properties an LSI operator is filtering out from a signal.

The case with nonorthogonal fit functions is slightly more complex. As the matrix $\boldsymbol{M}$ (Eq. (9.32)) depends only on the fit functions and the chosen window and not on the signal $g(\boldsymbol{x})$, the matrix $\boldsymbol{M}$ can be inverted once for a given fit. Then the fit coefficients are given as a linear combination of the results from the convolutions with all P fit functions:

$$a_p(\boldsymbol{x}) = \sum_{p'=0}^{P-1} M^{-1}_{p,p'} \int_{-\infty}^{\infty} w(\boldsymbol{x}') f_{p'}(-\boldsymbol{x}') g(\boldsymbol{x}-\boldsymbol{x}')\, \mathrm{d}^D x' \tag{9.40}$$

9.3 Recursive filters

9.3.1 Definition

Recursive filters are a special form of the linear convolution filters. This type of filter includes results from previous convolutions at neighboring pixels into the convolution sum. In this way, the filter becomes directional. Recursive filters can most easily be understood if we apply them first to a 1-D discrete signal, a *time series*. Then we can write

$$g'_n = -\sum_{n''=1}^{S} a_{n''} g'_{n-n''} + \sum_{n'=-R}^{R} h_{n'} g_{n-n'} \tag{9.41}$$

While the neighborhood of the nonrecursive part (coefficients h) is symmetric around the central point, the recursive part is asymmetric, using only previously computed values. A filter that contains only such a recursive part is called a *causal filter*. If we put the recursive part on the left hand side of the equation, we observe that the recursive filter is equivalent to the following difference equation, also known as an *ARMA*(S,R) process (*autoregressive-moving average process*):

$$\sum_{n''=0}^{S} a_{n''} g'_{n-n''} = \sum_{n'=-R}^{R} h_{n'} g_{n-n'} \quad \text{with} \quad a_0 = 1 \tag{9.42}$$

9.3.2 Transfer function and z-transform

The transfer function of such a filter with a recursive and a nonrecursive part can be computed by applying the *discrete-space Fourier transform*