EE416: Introduction to Image Processing and Computer Vision

Il Yong Chun
Department of Electrical and Computer Engineering, the University of Hawai‘i, Mānoa

November 1, 2021

6 "Optimal" linear filtering via stochastic approach

All of the major topics up until this point were essentially deterministic:

- imaging system, Fourier analysis, sampling
- interpolation and decimation
- contrast enhancement
- edge detection and image sharpening
- image denoising
- image segmentation

This chapter presents material related to these motivating applications and describes some image statistical models that one can apply in a variety of contexts.

6.1 Random variables, vectors, processes

6.1.1 Random variables

The defining characteristic of a real-valued random variable X is its **cumulative distribution function** (**cdf or CDF**), given by

$$F_X(x) \triangleq \mathbb{P}\{X \leq x\}, \quad \forall \in \mathbb{R}.$$

Note that a cdf/CDF has the following **end conditions**: $\lim_{t\to\infty} F_X(t) = 1$ and $\lim_{t\to-\infty} F_X(t) = 0$. For **continuous random variables**, the cdf is differentiable, and we define a corresponding **probability density function** (**pdf**):

$$p_X(x) \triangleq \frac{\mathrm{d}F_X(x)}{\mathrm{d}x}.$$

By the fundamental theorem of calculus, it follows from the end conditions above that

$$\mathbb{P}\{X \in \mathcal{B}\} = \int_{\mathcal{B}} p_X(x)\,\mathrm{d}x.$$

This holds for all the subsets $\mathcal{B}$ of $\mathbb{R}$ that one would ever need in practice (and lots more). In particular

$$F_X(x) \triangleq \mathbb{P}\{X \leq x\} = \int_{-\infty}^{x} p_X(x')\,\mathrm{d}x'$$

Caution: earlier in the course x denoted a spatial coordinate, but now it denotes a random variable!

Some important continuous random variables:

- uniform
- Gaussian or Normal
- Laplacian (double-sided exponential, for impulsive noise) or Laplace distribution

Example: The pdf of a Gaussian random variable with mean μ and variance σ^2:

$$p_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}(x-\mu)^2/\sigma^2}$$

Some important discrete random variables:

- Bernoulli
- Binomial
- Poisson

Example: The **probability mass function** (**PMF**) of a Poisson random variable with mean $\lambda > 0$:

$$\mathbb{P}\{X = l\} = e^{-\lambda}\lambda^k/k!, \quad k = 0, 1, \ldots.$$

Properties of random variables. The following definitions apply to both real- and complex-valued random variables:

- **mean** or **expected value**: $\mu_X \triangleq \mathbb{E}[X] = \int x p_X(x)\,\mathrm{d}x$
- expectation of a function of a RV [wiki]: $\mathbb{E}[g(X)] = \int g(x) p_X(x)\,\mathrm{d}x$
- variance: $\mathrm{Var}(X) \triangleq \mathbb{E}[|X - E[X]|^2] = \mathbb{E}[|X|^2] - |\mathbb{E}[X]|^2$
- correlation:[1] $\mathbb{E}[XY^*] = \int\int xy^* p_{X,Y}(x,y)\,\mathrm{d}x\,\mathrm{d}y$
- covariance: $\mathrm{Cov}(X,Y) \triangleq \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])^*] = \mathbb{E}[XY^*] - \mathbb{E}[X]\mathbb{E}[Y^*]$

The expected value $\mathbb{E}[X]$ of a random variable is also called its **ensemble average** because it is the "average" (weighted by the pdf $p_X(x)$) of all possible values of the random variable. In contrast the **sample mean** of a collection of numbers $x_1, \ldots, x_N$ is given by the **average**: $\frac{1}{N}\sum_{n=1}^{N} x_n$. One must be careful to distinguish between the mean of a random variable and the sample mean of a collection of numbers. In Matlab, the `mean` function computes the sample mean, not the ensemble mean.

We call variance, correlation, and covariance **2nd-order properties** because they depend on the square or products of two terms.

Independent random variables. We say X and Y are **independent random variables** iff their **joint distribution** factors into the product of **marginal distributions**:

$$p_{X,Y}(x,y) = p_X(x) p_Y(y).$$

Conditional random variables. The **conditional probability** is a measure of the probability of an event occurring, given that another event has already occurred. If the event of interest is A and the event B is known or assumed to have occurred, "the conditional probability of A given B" or "the probability of A under the condition B" is written as $\mathbb{P}(A|B)$. Here are important definitions and properties related to the conditional property:

- conditional pdf/PMF of X given the occurrence of value y of Y: $p_{X|Y}(x|y) = \frac{p_{X,Y}(x,y)}{p_Y(y)}$.
- random variables X, Y are independent iff the conditional distribution of X given Y, for all possible realizations of Y, equal to the unconditional distribution of X, i.e.,

$$p_{X|Y}(x|y) = p_X(x), \quad \forall x, y \text{ with } f_Y(y) > 0.$$

- conditional expectation of X given Y: $\mathbb{E}[X|Y] = \int_{-\infty}^{\infty} x p_{X|Y}(x|y)\,\mathrm{d}x$. See properties in [wiki].

[1]If $X = X_r + \mathrm{i}X_i$ and $Y = Y_r + \mathrm{i}Y_i$ are complex, then the integral for correlation is four-dimensional: $\mathbb{E}[XY^*] = \int\int \int\int (a + \mathrm{i}b)(c - \mathrm{i}d) p_{X_r,X_i,Y_r,Y_i}(a,b,c,d)\,\mathrm{d}a\,\mathrm{d}b\,\mathrm{d}c\,\mathrm{d}d$.

Deterministic vs. random. Let X and Z be random variables, and let $f : \mathbb{R} \to \mathbb{R}$.

- Is $Y = f(X)$ is a random variable? $\boxed{??}$
- Is $\mu = \mathbb{E}[X]$ a random variable? $\boxed{??}$
- Is $\hat{X} = \mathbb{E}[X|Z]$ a random variable? $\boxed{??}$
- Is $\hat{X} = \mathbb{E}[X|Z = \text{a particular value of } Z]$ a random variable? $\boxed{??}$

Properties of correlation/covariance.

- $\mathrm{Cov}(X,Y) = \mathbb{E}[XY^*] - \mu_X \mu_Y^*$
- $\mathrm{Cov}(X,Y) = \mathrm{Cov}^*(Y,X)$
- $\mathrm{Cov}(X,X) = \mathrm{Var}(X)$
- $\mathrm{Cov}(aX + b, cY + d) = ac^* \mathrm{Cov}(X,Y)$
- **Cauchy-Schwarz inequality**: $|\mathrm{Cov}(X,Y)| \leq \sigma_X \sigma_Y, \quad |\mathbb{E}[XY^*]| \leq \sqrt{\mathbb{E}[|X|^2]\mathbb{E}[|Y|^2]}$
- If X and Y are **independent random variables** then $\mathbb{E}[XY] = \mu_X \mu_Y$ so $\mathrm{Cov}(X,Y) = 0$.
- The reverse is *not true* in general (uncorrelated does not ensure independence); and exception is Gaussian distributed random variables.
- Covariance is an **bilinear operation**: $\mathrm{Cov}(\sum_n X_n, \sum_m Y_m) = \sum_i \sum_j \mathrm{Cov}(X_n, Y_m)$

6.1.2 Random vectors

A finite collection of random variables (over a common probability space) is called a **random vector**, often denoted $\mathbf{X} = (X_1, \ldots, X_N) \in \mathcal{X}$, where each X_i is a random variable and typically $\mathcal{X} = \mathbb{R}^N$ or $\mathcal{X} = \mathbb{C}^N$ for real or complex random vectors respectively.

The defining characteristics of a random vector is its **cumulative distribution function** (**CDF**), given by

$$F_{\mathbf{X}}(x_1, \ldots, x_N) = \mathbb{P}\{X_1 \leq x_1, \ldots, X_N \leq x_N\}.$$

Note that a CDF has the following **end conditions**: $\lim_{t_n \to \infty} F_{\mathbf{X}}(t_1, \ldots, t_N) = 1$ and $\lim_{t_n \to -\infty} F_{\mathbf{X}}(t_1, \ldots, t_N) = 0$ for $n = 1, \ldots, N$ and for any $\{t_n \in \mathbb{R}\}$.

For **continuous random vectors**, the CDF is differentiable, and we define a corresponding **probability density function** (**pdf**) by

$$p_{\mathbf{X}}(\mathbf{x}) \triangleq \frac{\partial^N F_X(x_1, \ldots, x_N)}{\partial x_1 \cdots \partial x_N}.$$

By the fundamental theorem of calculus, it follows from the end conditions above that

$$\mathbb{P}\{\mathbf{X} \in \mathcal{B}\} = \int_{\mathcal{B}} p_{\mathbf{X}}(\mathbf{x}) \, \mathrm{d}\mathbf{x}.$$

This holds for all the subsets $\mathcal{B}$ of $\mathcal{X}$ that one would ever need in practice (and lots more).

We say $X_1, \ldots, X_N$ are **independent random variables** iff

$$\mathbb{P}\{X_1 \leq x_1, \ldots, X_N \leq x_N\} = \prod_{n=1}^{N} \mathbb{P}\{X_n \leq x_n\}, \quad \{\forall x_n \in \mathbb{R} : n = 1, \ldots, N\}.$$

In particular, the elements of a continuous random vector are **independent** iff their pdf is **separable**:

$$p_{\mathbf{X}}(\mathbf{x}) = p_{\mathbf{X}}(x_1, \ldots, x_N) = \prod_{n=1}^{N} p_{X_n}(x_n), \quad \{\forall x_n \in \mathbb{R} : n = 1, \ldots, N\}.$$

Properties of random vectors.

- The **mean** or **expected value** of a random vector is defined by:

$$\mathbb{E}[\mathbf{X}] = \int_{\mathcal{X}} \mathbf{x} p_{\mathbf{X}}(\mathbf{x}) \, \mathrm{d}\mathbf{x}.$$

- The expectation of a function of a random vector is

$$\mathbb{E}[g(\mathbf{X})] = \int_{\mathcal{X}} g(\mathbf{x}) p_{\mathbf{X}}(\mathbf{x}) \, \mathrm{d}\mathbf{x}.$$

- The $N \times N$ **correlation matrix** of a length-N random vector (column vector) is

$$\mathbf{R}_{\mathbf{X}} = \mathbb{E}[\mathbf{X}\mathbf{X}^H], \text{ i.e., } [\mathbf{R}_{\mathbf{X}}]_{i,j} = \mathbb{E}[x_i x_j^*] = \int_{\mathcal{X}} x_i x_j^* p_{\mathbf{x}}(x_1, \ldots, x_N) \, \mathrm{d}\mathbf{x},$$

  where $(\cdot)^H$ denotes Hermitian transpose (for complex-valued random vectors).
- The $N \times N$ **covariance matrix** of two length-N random vectors is

$$\mathbf{C}_{\mathbf{X},\mathbf{Y}} = \mathrm{Cov}(\mathbf{X}, \mathbf{Y}) = \mathbb{E}[(\mathbf{X} - \mathbb{E}[\mathbf{X}])(\mathbf{Y} - \mathbb{E}[\mathbf{Y}])^H], \text{ i.e., } [\mathbf{C}_{\mathbf{X},\mathbf{Y}}]_{i,j} = \mathrm{Cov}(X_i, Y_j)$$
$$\mathbf{C}_{\mathbf{X},\mathbf{X}} = \mathbf{R}_{\mathbf{X}} - \mathbb{E}[\mathbf{X}]\mathbb{E}[\mathbf{X}]^H, \text{ i.e., } [\mathbf{C}_{\mathbf{X},\mathbf{X}}]_{i,j} = \mathrm{Cov}(X_i, X_j)$$

  If the elements of a random vector are **independent**, then its covariance matrix is **diagonal**, e.g., $\mathbf{C}_{\mathbf{X},\mathbf{X}} = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_N^2)$, where σ_n^2 is a variance for X_n, $\forall n$.
  Covariance matrices are positive semidefinite, which we write $\mathrm{Cov}(\mathbf{X}) \succeq \mathbf{0}$, i.e., $\mathbf{z}^T \mathrm{Cov}(\mathbf{X})\mathbf{z} \geq 0$ for a non-zero vector $\mathbf{z} \in \mathbb{R}^N$.

We call correlation and covariance **2nd-order properties of random vectors**.
Example: A particularly important type of continuous random vector $\mathbf{X}$ has a multivariate Gaussian or normal distribution, denoted

$$\mathbf{X} \sim \mathcal{N}(\mu, \mathbf{C})$$

where $\mu = \mathbb{E}[\mathbf{X}]$ denotes the mean vector of $\mathbf{X}$ and $\mathbf{C} = \mathrm{Cov}(\mathbf{X})$ denotes the covariance matrix of $\mathbf{X}$. For the usual case where $\mathbf{X}$ is real, its pdf is given by

$$p_{\mathbf{X}}(\mathbf{x}) = \frac{1}{\sqrt{\det(2\pi\mathbf{C})}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mu)^H \mathbf{C}^{-1}(\mathbf{x} - \mu)\right).$$

Note that if $\mathbf{C}$ is a $N \times N$ matrix, then $\det(\alpha\mathbf{C}) = \alpha^N \det(\mathbf{C})$.

6.1.3 Random processes (in 2D discrete space)

Notation.

- X_s is a pixel at position $s = (s_1, s_2) \in \mathbb{Z}^2$
- S denotes the set of 2D Lattice points where $S \subset \mathbb{Z}^2$

Definitions.

- Mean $\mu_s = \mathbb{E}[X_s]$
- Autocorrelation $R_{s,r} = \mathbb{E}[X_s X_r^*]$
- Autocovariance $C_{s,r} = \mathbb{E}[(X_s - \mu_s)(X_r - \mu_r)^*] = R_{s,r} - \mu_s \mu_r^*$
- A process is said to be **second order stationary** if $\mathbb{E}[X_s]$ and $\mathbb{E}[X_s X_r^*]$ exist for all $s \in S$ and $r \in S$.

- A second order stationery random process is said to be **wide sense stationary** (**WSS**) if for all $s \in \mathbb{Z}^2$

$$\begin{aligned} \mu_s &= \mu_{(0,0)} \\ C_{r,r+s} &= C_{(0,0),s} \end{aligned} \tag{1}$$

Properties of WSS random processes.

- Shift invariance: $R_{s,(0,0)} = \mathbb{E}[X_s X^*_{(0,0)}] = \mathbb{E}[X_{s+r} X^*_r]$, $\forall r \in \mathbb{Z}^2$.
  In particular a shifted version $Y_s = X_{s+s_0}$ for any $s_0 \in \mathbb{Z}^2$ of a WSS random process X_s has the same-order properties as the original: $\mu_{Y_s} = \mu_{X_s}$ and $R_{Y_s} = R_{X_s}$
- Variance: $C_{s,(0,0)} = \text{Var}(X_s)$ and $R_{s,(0,0)} = \text{Var}(X_s) + |\mu_s|^2$, $\forall s \in \mathbb{Z}^2$.
- Hermitian symmetry: $R_{-s,(0,0)} = R^*_{s,(0,0)}$.

  Proof. $R_{-s,(0,0)} = R_{(0,0)-s,(0,0)} = R_{(0,0),s} = R^*_{s,(0,0)}$. □
- Cauchy-Schwarz: $|R_s| \le R_{s,(0,0)}$.

Note that mathematically a WSS random process is defined for all $s \in \mathbb{Z}^2$, in part so that (1) is well defined. In practice, we work with a finite-sized image and it would be imprecise to say that finite-sized image is a WSS random process; instead it would be more precise to say that such a finite-sized image is a portion of an infinite extent WSS random process.

6.1.4 2D Power spectral density of WSS processes

It is also important to consider the behavior of WSS random processes in the frequency domain. A naive attempt at frequency-domain analysis would be to try to analyze the DSFT of $x[m,n]$. However, WSS processes have infinite extent, and for almost all realizations the DSFT of $x[m,n]$ will not exist! (Of course the DSFT of any finite rectangular portion of $x[m,n]$ will always exist, but that DSFT will not generally have the properties of primary interest.)

So we will never refer to the "DSFT $X(e^{i\mu}, e^{i\nu})$" of a WSS process!

Instead we consider the DSFT of the **autocorrelation function** of a WSS random process, which is called its **power spectrum** or **power spectral density**:

$$\boxed{S_X(e^{i\mu}, e^{i\nu}) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} R_{(m,n),(0,0)} e^{-i(m\mu + n\nu)}}$$

This relationship is made rigorous by the **Wiener-Khinchin theorem** [1].
Some notes:

- The wider the power spectrum, the more high frequency fluctuations the random process has on average, which typically means that neighboring values are less correlated.
- Because the autocorrelation function of a WSS random process is Hermitian symmetric, the power spectrum is *real*.
- Power spectra are always *nonnegative*.

Example: A zero-mean WSS **white random process** $X[m,n]$ with variance σ^2 has a constant (flat) power spectrum:

$$R_{(m,n),(0,0)} = C_{(m,n),(0,0)} = \sigma^2 \delta[m,n] \overset{\text{DSFT}}{\Longleftrightarrow} S_X(e^{i\mu}, e^{i\nu}) = \sigma^2.$$

If $X[m,n]$ has nonzero mean $\mu_{(0,0)}$, then

$$C_{(m,n),(0,0)} = \sigma^2\delta[m,n] \overset{\text{DSFT}}{\Longleftrightarrow} \sigma^2, \quad \text{(We do not have a name for the DSFT of } C_{(m,n),(0,0)}.\text{)}$$

$$R_{(m,n),(0,0)} = \sigma^2\delta[m,n] + |\mu_{(0,0)}|^2 \overset{\text{DSFT}}{\Longleftrightarrow} S_X(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) = \sigma^2 + |\mu_{(0,0)}|^2(2\pi)^2 \sum_{k,l=-\infty}^{\infty} \delta(\mu - k2\pi, \nu - l2\pi)$$

Note: the term "white" implies that the process has a constant power spectrum (except possibly for a Dirac impulse at 0), and saying that it has a power spectrum means that it is WSS. So we usually say "white noise" rather than "WSS white noise" because WSS is implied.

6.2 Minimum mean square error filter for "optimal" image denoising

This section considers image restoration problems. Our aim is to restore an image $\mathbf{x}$ from its corrupted version $\mathbf{y}$, by finding an "optimal" linear FIR filter that is designed to minimize **mean square error** (**MSE**). The source of corruption includes additive noise, non-additive noise, linear distortion, nonlinear distortion, etc.

Notations.

- Filters uses input window of $\mathbf{y}$ to estimate each output pixel x_s.
- $x_s^\star$ denotes the estimate of x_s.
- $W(s)$ denotes the window about the pixel location s.

The estimate $x_s^\star$ is a function of $\mathbf{y}_{W(s)}$:

$$x_s^\star = f(\mathbf{y}_{W(s)}).$$

- The function $f(\mathbf{y}_{W(s)})$ is designed to produce a **minimum mean square error** (**MMSE**) estimate of $\mathbf{x}$.
- If $f(\mathbf{y}_{W(s)})$ is linear, then it is linear space invariant filter.
  If $f(\mathbf{y}_{W(s)})$ is nonlinear, then it is nonlinear space invariant filter.
- This filter can reduce the effects of all types of corruption.

Optimality properties of linear filter.

- If $\mathbf{x}$ and $\mathbf{y}$ are jointly Gaussian, then MMSE filter is linear:

$$x_s^\star = \mathbb{E}[x_s|\mathbf{y}_{W(s)}] = \mathbf{A}\mathbf{y}_{W(s)} + b.$$

- If $\mathbf{x}$ and $\mathbf{y}$ are not jointly Gaussian, then MMSE filter is generally not linear:

$$x_s^\star = \mathbb{E}[x_s|\mathbf{y}_{W(s)}] = f(\mathbf{y}_{W(s)}).$$

  However, the MMSE linear filter can still be very effective!

Even we know the exact distributions of $\mathbf{x}$ and $\mathbf{y}$, direct numerical evaluation of the conditional expectation is computationally expensive since it often requires multidimensional integration. The next section formulates problems of finding MMSE linear filters with the machine learning perspective.

6.2.1 Formulation of MMSE linear filter

Notations and definitionts.

- $W(s)$: window about the pixel location s;
  P: number of pixels in $W(s)$
  $W(s) = [s, s + r_1, \ldots, s + r_{P-1}]$, where $r_1, \ldots, r_{P-1}$ are index neighbors.
- $\mathbf{z}_s$: column vector containing pixels of $y_{W(s)}$
  $\mathbf{z}_s^T = [(y_{W(s)})_1, (y_{W(s)})_2, \ldots, (y_{W(s)})_P]$
- θ: column vector containing filter parameters
  $\theta = \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_P \end{bmatrix}$

Objectives.

- Linear filtering is given by $x_s^\star = \mathbf{z}_s^T \theta$
- MSE is given by
$$\text{MSE} = \mathbb{E}[|x_s - x_s^\star|^2] = \mathbb{E}[|x_s - \mathbf{z}_s^T\theta|^2]. \tag{2}$$
- The MMSE filter parameters $\theta^\star$ are given by
$$\theta^\star = \underset{\theta}{\text{argmin}}\ \mathbb{E}[|x_s - \mathbf{z}_s^T\theta|^2].$$
  How do we solve this problem?

Patch-based filtering.

- The subset S_0 of image pixels
  - $S_0 \subset S$
  - S_0 contains $N_0 < N$ pixels
  - S_0 usually does not contain pixels on the boundary of the image
  - $S_0 = [s_1, \ldots, S_{N_0}]$
- $\mathbf{Z} = \begin{bmatrix} \mathbf{z}_{s_1}^T \\ \vdots \\ \mathbf{z}_{s_{N_0}}^T \end{bmatrix} \in \mathbb{R}^{N_0 \times P}$
- $\mathbf{x} = \begin{bmatrix} x_{s_1} \\ x_{s_2} \\ \vdots \\ x_{s_{N_0}} \end{bmatrix} \in \mathbb{R}^{N_0}$; $\mathbf{x}^\star = \begin{bmatrix} x_{s_1}^\star \\ x_{s_2}^\star \\ \vdots \\ x_{s_{N_0}}^\star \end{bmatrix} \in \mathbb{R}^{N_0}$.
- $\mathbf{x} \approx \mathbf{x}^\star = \mathbf{Z}\theta$; called "patch-based" perspective in modern image processing and computer vision literature [2, 3].

6.2.2 Least squares linear filter

From (2), we expect that

$$\text{MSE} = \mathbb{E}[|x_s - \mathbf{z}_s^T\theta|^2] \approx \frac{1}{N_0}\sum_{s \in S_0} |x_s - \mathbf{z}_s^T\theta|^2 = \frac{1}{N_0}\|\mathbf{x} - \mathbf{Z}\theta\|_2^2.$$

So we aim to solve the following optimization problem:

$$\theta^\star = \underset{\theta}{\operatorname{argmin}} \ \|\mathbf{x} - \mathbf{Z}\theta\|_2^2. \tag{3}$$

The solution $\theta^\star$ is the **least squares** (**LS**) estimate of θ, and the estimate $\mathbf{x}^\star = \mathbf{Z}\theta^\star$ is known as the **LS filter**.

Solution to (3). We first rewrite the optimization problem (3):

$$\begin{aligned}\min_{\theta} \frac{1}{N_0}\|\mathbf{x} - \mathbf{Z}\theta\|_2^2 &= \min_{\theta} \frac{1}{N_0}(\mathbf{x} - \mathbf{Z}\theta)^T(\mathbf{x} - \mathbf{Z}\theta) \\ &= \min_{\theta} \frac{1}{N_0}\left(\mathbf{x}^T\mathbf{x} - 2\theta^T\mathbf{Z}^T\mathbf{x} + \theta^T\mathbf{Z}^T\mathbf{Z}\theta\right) \\ &= \min_{\theta} \theta^T\frac{\mathbf{Z}^T\mathbf{Z}}{N_0}\theta - 2\theta^T\frac{\mathbf{Z}^T\mathbf{x}}{N_0} \end{aligned} \tag{4}$$

Observe that the first and second terms in (4) can be rewritten by

- $\mathbf{R}^\star_{\mathbf{z},\mathbf{z}} \triangleq \frac{\mathbf{Z}^T\mathbf{Z}}{N_0} = \frac{1}{N_0}\left[\mathbf{z}_{s_1}, \mathbf{z}_{s_2}, \dots \mathbf{z}_{s_{N_0}}\right]\begin{bmatrix}\mathbf{z}_{s_1}^T \\ \mathbf{z}_{s_2}^T \\ \vdots \\ \mathbf{z}_{N_0}^T\end{bmatrix} = \frac{1}{N_0}\sum_{n=1}^{N_0}\mathbf{z}_{s_n}\mathbf{z}_{s_n}^T$

  The matrix $\mathbf{R}^\star_{\mathbf{z},\mathbf{z}}$ is an estimate of the covariance matrix of $\mathbf{z}_s$:

$$\mathbb{E}[\mathbf{R}^\star_{\mathbf{z},\mathbf{z}}] = \mathbb{E}\left[\frac{1}{N_0}\sum_{s=1}^{N}\mathbf{z}_s\mathbf{z}_s^T\right] = \mathbb{E}[\mathbf{z}_s\mathbf{z}_s^T] = \mathbf{R}_{\mathbf{z},\mathbf{z}}$$

- $\mathbf{r}^\star_{\mathbf{z},x} \triangleq \frac{\mathbf{Z}^T\mathbf{x}}{N_0} = \frac{1}{N_0}\left[\mathbf{z}_{s_1}, \mathbf{z}_{s_2}, \dots \mathbf{z}_{s_{N_0}}\right]\begin{bmatrix}x_{s_1} \\ x_{s_2} \\ \vdots \\ x_{N_0}\end{bmatrix} = \frac{1}{N_0}\sum_{n=1}^{N_0}\mathbf{z}_{s_n}^T x_{s_n}.$

  The vector $\mathbf{r}^\star_{\mathbf{z},x}$ is an estimate of the cross correlation between $\mathbf{z}_s$ and x_s:

$$\mathbb{E}[\mathbf{r}^\star_{\mathbf{z},x}] = \mathbb{E}\left[\frac{1}{N_0}\sum_{s=1}^{N}\mathbf{z}_s x_s\right] = \mathbb{E}[\mathbf{z}_s x_s] = \mathbf{R}_{\mathbf{z},x}$$

Using the above observations, we rewirte (4) as follows:

$$\theta^\star = \underset{\theta}{\operatorname{argmin}} \ \theta^T\mathbf{R}^\star_{\mathbf{z},\mathbf{z}}\theta - 2\theta^T\mathbf{r}^\star_{\mathbf{z},x} \tag{5}$$

Taking the gradient of the cost function in (5) gives

$$\nabla_\theta\left(\theta^T\mathbf{R}^\star_{\mathbf{z},\mathbf{z}}\theta - 2\theta^T\mathbf{r}^\star_{\mathbf{z},x}\right) = 2\mathbf{R}^\star_{\mathbf{z},\mathbf{z}}\theta - 2\mathbf{r}^\star_{\mathbf{z},x}$$

(see [The Matrix Cookbook]), and setting $\nabla_\theta = \mathbf{0}$ gives the solution

$$\theta^\star = \left(\mathbf{R}^\star_{\mathbf{z},\mathbf{z}}\right)^{-1}\mathbf{r}^\star_{\mathbf{z},x}.$$

Training. The vector $\theta^\star$ is usually estimated from "training" data.

- **Training data** generally consists of image pairs $(\mathbf{x}, \mathbf{y})$ where $\mathbf{y}$ is the measured data and $\mathbf{x}$ is the ground truth image.
  Should be typical of what you might expect.
- **Testing data** (consisting of image pairs $(\mathbf{x}, \mathbf{y})$) is used to evaluate the effectiveness of the filters. Should never be taken from the training data set.
- Training vs testing
  - Performance on training data is always better than performance on testing data.
  - As the amount of training data increases, the performance on training and testing data both approach the best achievable performances.

Comments.

- Wiener filter is the MMSE linear filter.
- Wiener filter is optimal in the sense of MSE, but is not always good.
  - Linear filters blur edges.
  - Linear filters work poorly with non-Gaussian noise.
- Nonlinear filters can be designed using the same methodologies (e.g., DnCNN [4, 5] and convolutional autoencoders [3, 5–7])

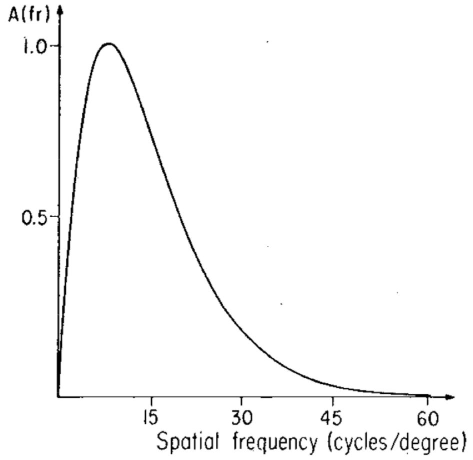
6.3 Minimum weighted mean square error filtering for human visual system

In general, MSE is **NOT** a good quality criteria for images, but sometimes it is OK. For achromatic images, it is best to choose $\mathbf{x}$ and $\mathbf{y}$ in gamma corrected space or perceptually uniform lightness space. A better metric error is based on the human visual system (HVS), called **human visual system error (HVSE)**:

$$\text{HVSE} = \|\mathbf{H}(\mathbf{x} - \mathbf{x}^\star)\|_2^2 = (\mathbf{x} - \mathbf{x}^\star)^T \mathbf{H}^T \mathbf{H}(\mathbf{x} - \mathbf{x}^\star) = \|\mathbf{x} - \mathbf{x}^\star\|_{\mathbf{W}}^2$$

where $\mathbf{H}$ is a filter that implements the **contrast sensitivity function (CSF)** for the HVS, $\mathbf{W} = \mathbf{H}^T\mathbf{H}$, and $\|\cdot\|_{\mathbf{W}}^2$ denotes weighted ℓ_2-norm. Note that $\|\cdot\|_{\mathbf{W}}^2$ is a quadratic norm.

CSF. CSF is the contrast sensitivity measured as a function of the spatial frequency in cycles per degree [8]:



- Bandpass function
- High-Frequency cut-off primarily due to optics of eye.
- Low-frequency cut-off due to neural response.
- Accurate measurement of CSF requires specialized techniques.

Minimum HVSE estimate $\mathbf{x}^\star$. Consider the following HVSE minimization problem:

$$\min_{\mathbf{x}^\star} \mathbb{E}\left[\|\mathbf{x}-\mathbf{x}^\star\|_{\mathbf{W}}^2\right] = \min_{\mathbf{x}^\star} \mathbb{E}\left[\|\mathbf{H}(\mathbf{x}-\mathbf{x}^\star)\|_2^2\right] = \boxed{??}$$

where $\mathbf{v} = \mathbf{H}\mathbf{x}$. Observe next that

$$\begin{aligned}\operatorname*{argmin}_{\mathbf{v}^\star} \mathbb{E}\left[\|\mathbf{v}-\mathbf{v}^\star\|_2^2\right] &= \mathbb{E}\left[\|\mathbf{v}-\mathbb{E}[\mathbf{v}|\mathbf{y}]\|_2^2\right] = \mathbb{E}\left[\|\mathbf{H}\mathbf{x}-\mathbb{E}[\mathbf{H}\mathbf{x}|\mathbf{y}]\|_2^2\right] = \mathbb{E}\left[\|\mathbf{H}(\mathbf{x}-\mathbb{E}[\mathbf{x}|\mathbf{y}])\|_2^2\right] \\ &= \mathbb{E}\left[\|\mathbf{x}-\mathbb{E}[\mathbf{x}|\mathbf{y}]\|_{\mathbf{W}}^2\right]\end{aligned}$$

So $\mathbf{x}^\star = \mathbb{E}[\mathbf{x}|\mathbf{y}]$ minimizes the error measure $\|\mathbf{x}-\mathbf{x}^\star\|_{\mathbf{W}}^2$.

- The estimate $\mathbf{x}^\star$ that can achieve minimum HVSE is

$$\mathbf{x}^\star = \mathbb{E}[\mathbf{x}|\mathbf{y}].$$

  - This is the same as for MSE (see §6.2)!
  - The conditional expectation minimizes any quadratic norm of the error.
  - This is also true for non-Gaussian images.
- Let $\mathbf{x}^\star = \mathbf{A}\mathbf{y} + \mathbf{b}$ be the MMSE linear filter.
  - This filter is also the minimum HVSE linear filter.
  - This is also true for non-Gaussian images.

6.4 Performance measures.

To quantify both the degree of degradation as well as the improvements due to restoration algorithms, various performance measures are used in the literature. Here are some examples.

- The **MSE** between a restoration $x^\star[m,n]$ and an original $M \times N$ image $x[m,n]$ is often defined to be the per-pixel average squared error:

$$\text{MSE} = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} |x^\star[m,n] - x[m,n]|^2 .$$

  Note that this is a *spatial* average, not an ensemble average (expectation).
- If we have a statistical model for the images, i.e., we treat $x[m,n]$ as random process, then often we define MSE in terms of ensemble average:

$$\text{MSE} = \mathbb{E}\left[|x^\star[m,n] - x[m,n]|^2\right]$$

  Note that this MSE definition could be a function of pixel locations m, n, but typically we use this MSE in the context of WSS random process models, in which case the MSE is not a function of m, n.
- A slight inconvenience of MSE is that its units are the square of the original image units. In photographic imaging, where the intensities typically are arbitrary units, this is a minor issue. But in applications like X-ray CT, where the pixel values have physically meaningful units, it can be preferable to use error measures that are expressed in those very units. One simple solution is to take the square root of MSE, known as the **root mean-squared error** (**RMSE**):

$$\text{RMSE} = \sqrt{\text{MSE}}.$$

- In cases where the image units are arbitrary, it is preferable to report error measures that are invariant to the choice of units by normalizing. There are a multiple definitions of **normalized root mean square error** (**NRMSE**) in the literature, so when reporting results it is wise to specify which version was used. A typical definition for a single realization is:

$$\text{NRMSE} = \frac{\text{RMSE}}{\sqrt{\frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}|x[m,n]|^2}},$$

and for case with random process models:

$$\text{NRMSE} = \frac{\text{RMSE}}{\sqrt{\mathbb{E}\left[|x[m,n]|^2\right]}}.$$

- No matter which NRMSE definition is used, usually we define the **signal-to-noise ratio** (**SNR**) as its reciprocal, often in dB:

$$\text{SNR} = 20\log_{10}\frac{1}{\text{NRMSE}}.$$

- The **peak SNR** or **PSNR** is also popular, usually in dB:

$$\text{PNSR} \triangleq 20\log_{10}\frac{\max_{m,n}|f(m,n)|}{\text{RMSE}}.$$

- The **blur SNR** or **BSNR** is defined in terms of the noisy, blurry image $y[m,n]$ as follows:

$$\text{BSNR} \triangleq 10\log_{10}\frac{\frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}|y[m,n]-\mu_y|^2}{\sigma^2},$$

where the mean value of the (noiseless) blurred image is $\mu_y = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\mathbb{E}[y[m,n]]$ and σ^2 denotes the noise variance. Adding a constant value to an image does not change the BSNR.
- All of the above are "squared error" performance measures, which are convenient for analysis. Worst-case errors and ℓ_1 errors can also be of interest:

$$\|\mathbf{x}^\star - \mathbf{x}\|_\infty = \max_{m,n}|x^\star[m,n]-x[m,n]|, \qquad \frac{1}{MN}\|\mathbf{x}^\star-\mathbf{x}\|_1 = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}|x^\star[m,n]-x[m,n]|.$$

Clearly there are a variety of possible performance measures. Because there are so many variations on the definition of SNR and NRMSE, one should always define which one is used. Although "squared error" and SNR-type performance measures are used frequently in image processing and computer vision, increasing SNR does not necessarily mean that the image is "better" for a given purpose. Other approaches try to use perceptual metrics or metrics that relate to how well a human observer can perform a task (such as detecting a tumor) from the image.

Homework (due by 11/12 11:55 PM; Upload your solution and Matlab codes to Laulima/EE416(F21)/Assignments)

Prob. 1. Let $\mathbf{y} \in \mathbb{R}^N$ be a vector containing the pixels in an image window. We model $\mathbf{y}$ as

$$\mathbf{y} = \mathbf{a}x + \mathbf{n}$$

where x is scalar-valued Gaussian random variable with zero-mean and variance σ^2, $\mathbf{a} \in \mathbb{R}^N$ is a deterministic column vector, and $\mathbf{n}$ is an independent Gaussian random vector of correlated noise with

distribution $\mathcal{N}(0, \mathbf{C_n})$, in which $\mathbf{C_n} \in \mathbb{R}^{N \times N}$ is a positive definite covariance matrix.
Our objective is to estimate x from the observation $\mathbf{y}$. To do this, we want to find a MMSE linear estimator for x given by

$$x^\star = \mathbf{y}^T \theta$$

where $\theta \in \mathbb{R}^N$ is a vector of filter coefficients.
In addition, define the covariance matrix of $\mathbf{y}$ by

$$\mathbf{C_y} = \mathbb{E}[\mathbf{y}\mathbf{y}^T]$$

and the cross-covariance column vector of $\mathbf{y}$ and x by

$$\mathbf{r}_{\mathbf{y},x} = \mathbb{E}[\mathbf{y}x].$$

(a) Derive an expression for the MSE given by $\mathbb{E}[\|x - x^\star\|^2]$ in terms of $\mathbf{C_y}$, $\mathbf{r}_{\mathbf{y},x}$, σ^2, and θ.
(b) Use the expression from (a) to compute the value of θ that produces the MMSE estimate of x.
(c) Derive $\mathbf{C_y}$ in terms of $\mathbf{a}$, σ^2, and $\mathbf{C_n}$.
(d) Derive $\mathbf{r}_{\mathbf{y},x}$ in terms of $\mathbf{a}$, σ^2, and $\mathbf{C_n}$.
(e) Use the above results to calculate a closed-form expression for $x^\star$.

Prob. 2. Consider a non-linear estimation problem for which we are trying to estimate the value of a scalar x_n from a vector of observations $\mathbf{y}_n$ using nonlinear estimator give by

$$x_n^\star = f(\mathbf{y}_n, \theta)$$

where $\theta \in \mathbb{R}^P$ is a parameter vector that controls the behavior of the nonlinear estimator.
Suppose that training data pairs are given by the form $(x_n, \mathbf{y}_n)$.[2] The data is partitioned into two sets. The first set, $n \in S_1$, contains $L = |S_1|$ pairs, and is used for training purposes. The second set, $n \in S_2$, contains $M = |S_2|$ pairs, and is used for testing purposes.
Using these data, we define the *training MSE* as

$$\mathrm{MSE}_1(\theta) = \frac{1}{L} \sum_{n \in S_1} \|x_n - f(\mathbf{y}_n, \theta)\|_2^2,$$

the *testing MSE* as

$$\mathrm{MSE}_2(\theta) = \frac{1}{M} \sum_{n \in S_2} \|x_n - f(\mathbf{y}_n, \theta)\|_2^2,$$

and the expected MSE as

$$\mathrm{MSE}_3(\theta) = \mathbb{E}[\|x_n - f(\mathbf{y}_n, \theta)\|_2^2].$$

Based on these error measures, we define the following two estimates for the parameter vector:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}}\ \mathrm{MSE}_1(\theta)$$
$$\theta^\star = \underset{\theta}{\operatorname{argmin}}\ \mathrm{MSE}_3(\theta)$$

(a) Between the two quantities, $\mathrm{MSE}_2(\hat{\theta})$ and $\mathrm{MSE}_2(\theta^\star)$, which would you expect to be smaller? Justify your answer. (Hint: compare $\mathbb{E}[\mathrm{MSE}_2(\hat{\theta})]$ and $\mathbb{E}[\mathrm{MSE}_2(\theta^\star)]$.)
(b) What is the limitation of using $\mathrm{MSE}_2(\theta^\star)$?
(c) Approximately how large should L be in order for $\hat{\theta}$ to be useful?

[2] Assume that each training data pair is independent, and each pair has a identical distribution.

(d) Sketch the plots of $\mathrm{MSE}_1(\hat{\theta})$, $\mathrm{MSE}_2(\hat{\theta})$, and $\mathrm{MSE}_2(\theta^\star)$ as a function of the amount of training data L.

Prob. 3 (Computer project). Your objects are *1)* to learn a linear filter that minimize MSE loss (3) with training noisy-clean image pairs and *2)* to investigate your learned filter to a test noisy image. In Laulima/EE416(F20)/Resources/Computer projects, you can download 4 noisy-clean image pairs: training image pairs used the 'couple', 'hill', and 'lena' images, and a test image pair used the 'barbara' image. You are also given the patch extraction function, a data matrix where each column is a vectorized extracted patch stacked vectorized patch = my_im2col(image, kernel size, 1). Set your filter size as 5×5, i.e., kernel size = [5,5].
(a) Write a code that constructs the matrix $\mathbf{Z}$ in §6.2.1 from the 3 noisy training images.
(b) Write a code that constructs the vector $\mathbf{x}^\star$ in §6.2.1 from the 3 clean training images. To properly construct the vector $\mathbf{x}^\star$, you are required to throughly understand how the given function my_im2col extracts and order patches from an image.
(c) Write a code to learn the filter parameter $\theta^\star$ that minimizes MSE loss (3), by using LS solution in §6.2.2. Construct a 2D 5×5 filter from $\theta^\star$.
(d) Use your 2D convolution codes implemented in Prob. 2 of Ch. 2, and test the performance of the learned filter in (c) to the noisy test image. Note that the solution in (c) did not consider the filter rotation defined in 2D convolution studied in Ch. 2, so you will need to consider this in applying your learned filter. Compute the PSNR value of your output image. In addition, denoise the noisy test image with the 5×5 moving average filter.
(e) Compute PSNR values from the 2 denoised images in (d), setting the max value as 255 in your PSNR calculations. Compare performances both quantitatively and qualitatively, i.e., which denoised image has higher PSNR value and looks more appealing.

References

[1] L. Cohen, "Generalization of the Wiener-Khintchine theorem," *IEEE Signal Process. Lett.*, vol. 5, no. 11, pp. 292–294, Nov. 1998.

[2] I. Y. Chun and J. A. Fessler, "Convolutional dictionary learning: Acceleration and convergence," *IEEE Trans. Image Process.*, vol. 27, no. 4, pp. 1697–1712, Apr. 2018, doi: 10.1109/TIP.2017.2761545.

[3] ——, "Convolutional analysis operator learning: Acceleration and convergence," *IEEE Trans. Image Process.*, vol. 29, pp. 2108–2122, 2020, doi: 10.1109/TIP.2019.2937734.

[4] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3142–3155, Feb. 2017.

[5] I. Y. Chun, Z. Huang, H. Lim, and J. A. Fessler, "Momentum-Net: Fast and convergent iterative neural network for inverse problems," early access in *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020, doi: 10.1109/TPAMI.2020.3012955.

[6] I. Y. Chun, X. Zheng, Y. Long, and J. A. Fessler, "BCD-Net for low- dose CT reconstruction: Acceleration, convergence, and generalization," in *Proc. Med. Image Computing and Computer Assist. Interven.*, Shenzhen, China, Oct. 2019, pp. 31–40, doi: 10.1007/978-3-030-32226-7_4.

[7] I. Y. Chun and J. A. Fessler, "Deep BCD-net using identical encoding-decoding CNN structures for iterative image recovery," in *Proc. IEEE IVMSP Workshop*, Zagori, Greece, Jun. 2018, pp. 1–5, 10.1109/IVMSPW.2018.8448694.

[8] J. Mannos and D. Sakrison, “The effects of a visual fidelity criterion of the encoding of images,” *IEEE Trans. Inf. Theory*, vol. 20, no. 4, pp. 525–536, Jul. 1974.