Computational Vision
U. Minn. Psy 5036
Daniel Kersten
Lecture 8: Linear Systems & optics

Outline

Last time

Generative models: intensity-based (e.g. 2D images) and scene-based (e.g. 3D graphics)

Point spread functions, principle of superposition, led to convolution model of image transformation by the optics

Today

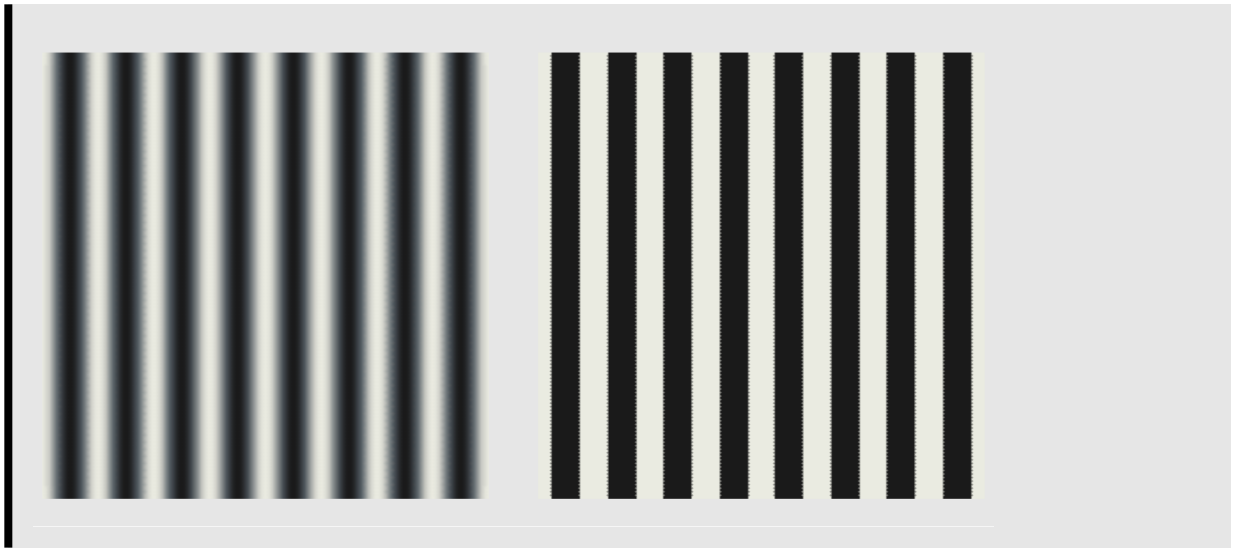
Linear systems analysis: Linear intensity-based image models + linear systems

Fourier synthesis/analysis, and the spatial frequency analysis of images

- Vector/matrix modeling of optical image transformation
- Linear systems
- Eigenfunctions of linear shift-invariant systems
- Spatial frequency analysis

Application: Modulation transfer functions and the human eye

Preview: Sinewaves are a special type of pattern. We'll try to understand why, and why they are useful.



Linear systems analysis

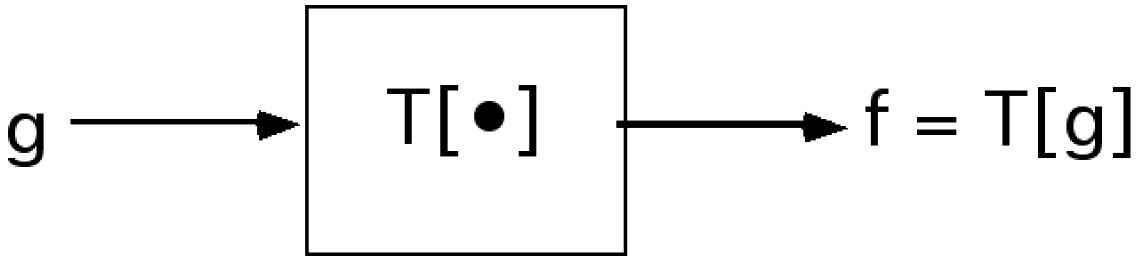
The world of input/output systems can be divided up into linear and non-linear systems. Linear systems are nice because the mathematics that describes them is not only well-known, but also has a mature elegance. On the other hand, it is a fair statement to say that most real-world systems are not linear, and thus hard to analyze...but fascinating if for that reason alone. That nature is usually non-linear doesn't mean one shouldn't familiarize oneself with the basics of linear system theory. Many times a non-linear system has a sufficiently smooth mapping that it can be approximated by a linear one over restricted ranges of parameter values. The assumption of linearity is an excellent starting point--but must be tested. The optics of the eye can be approximated as a linear system over small patches. The retinal and neural networks in primary visual cortex can also be approximated as linear systems.

So exactly what is a "linear system"?

The notion of a "linear system" is a generalization of the input/output properties of a straight line passing through zero. For any matrix **W**, the equation **W.g** == **f** represents a linear system, and so is the one we studied above (**T.g = f**). If **W** is a matrix, **g1** and **g2** are vectors, and a and b are scalars, it is easy to show that:

$$\mathbf{W}.(a\,\mathbf{g}1 + b\,\mathbf{g2}) = a\,\mathbf{W.g1} + b\,\mathbf{W.g2}$$

This is a consequence of the laws of matrix algebra.The idea of a linear system has been generalized beyond matrix algebra. Imagine we have a box that takes inputs such as g, and outputs **f = T[g]**.



The abstract definition of a linear system is that it satsifies:

$$T[a\,\mathbf{g1} + b\,\mathbf{g2}] = a\,T[\mathbf{g1}] + b\,T[\mathbf{g2}]$$

where T is the transformation that takes the sum of scaled inputs g1, g2 (which can be functions or vectors) to the sum of the scaled transformation of g1 and g2. The property, that the output of a sum is the sum of the outputs, is sometimes known as the *superposition principle* for linear systems--the assumption we needed to add up all the PSF contributions earlier. The property that a scaled version of the input results in an output scaled by the same amount (i.e. by a or by b) is called the *homogeneity principle*. The fact that linear systems show superposition is good for doing theory, but as we will see later, it limits the kind of input/output computations that can be done with linear systems, and thus with linear neural network models.

Shift-invariance means that if:

g(x,y) -> f(x,y)

then if you shift the input image over by (a,b) and run it through the transformation, you get the same result as just shifting over the original output image by the same amount:

g(x-a, y-b)→f(x-a, y-b)

Shift-invariance is a good assumption over small optical regions. But once the image gets transduced, it breaks down for visual neural processing due to the large differences between foveal and extra-foveal vision.

Vector/matrix modeling of image transformations: Linear intensity-based

Suppose that **T[ ]** is a linear system, with g1 and g2 input images (on a high-resolution computer screen that you are viewing), and f1, f2 output images (e.g. on your retina). By definition, if

g1 -> f1

and

g2 -> f2,

then a linear combination of the inputs maps to a linear combination of the outputs

a***g1+** b***g2 ->** a***f1 +** b***f2**

where a and b are scalar weights. We now want to understand how to characterize the transformation in terms of the properties of a matrix **T**.

But first let's see the ways in which one can represent the input vector as combinations of other vectors. We'll assume a linear intensity-based generative model of images.

Vector/matrix modeling of optical image transformation: Simple 1D case

■ Output image "response" to a single point of light (pixel)

For the time being, let's pretend our images are one-dimensional, and represented as vectors whose positions indicate pixel location, and whose values are intensities. Suppose that an input image **g** ={g1,g2,...}.

Let an input *test* image (e.g. on a computer test screen) be represented by **ui** = {0,0,0,...,1,...} where all pixels are black (zero) except for the ith one which is bright (1) . If the 4th pixel is bright, for example, the test image is represented by a vector **u4** = {0,0,0,1,0,0,...}:

$$\mathbf{u4}=\begin{pmatrix}0\\0\\0\\1\\0\\0\\0\\0\end{pmatrix}$$

We'll use **bold** for vectors and matrices, and plain for scalars. Let **t4** = {t6, t7, t8,t1, ...} represent the output image (e.g. retinal image), i.e. *point spread function (PSF)*, that results from **u4**

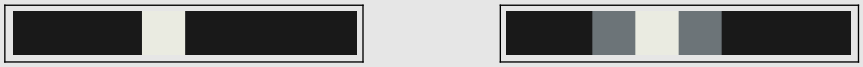
$$\mathbf{t4}=\begin{pmatrix}t6\\t7\\t8\\t1\\t2\\t3\\t4\\t5\end{pmatrix}$$

For example, if there is a little local blurring, t4 might look like this:

$$\mathbf{t4}=\begin{pmatrix}0\\0\\1/4\\1/2\\1/4\\0\\0\\0\end{pmatrix}$$

In[28]:=
```
u4 = {0, 0, 0, 1, 0, 0, 0, 0};
t4 = {0, 0, 1 / 4, 1 / 2, 1 / 4, 0, 0, 0};
GraphicsRow[{ArrayPlot[{u4}, ColorFunction → "GrayTones"],
  ArrayPlot[{t4}, ColorFunction → "GrayTones"]}]
```

Out[30]=



■ Output image "response" to a spatially shifted point of light

What if we now shift the input pattern over a position, so the bright pixel is at the (i+1)th position? E.g.

$$\mathbf{u5}=\begin{pmatrix}0\\0\\0\\0\\1\\0\\0\\0\end{pmatrix}$$

For a system like the eye, we might expect that the output image would be basically the same form, except that it also gets shifted over:

$$\mathbf{t5}=\begin{pmatrix}X\\t6\\t7\\t8\\t1\\t2\\t3\\t4\end{pmatrix}$$

We've stuck an "X" at the slot left open after the shift. There are various ways of modeling the boundaries. One way which is convenient mathematically is to use a circular boundary which works quite well for large images, and where the PSF is local and falls off to zero away from the ith position. Then we wrap the entries around, so in the above case X -> t5.

$$\mathbf{t5}=\begin{pmatrix}0\\0\\0\\1/4\\1/2\\1/4\\0\\0\end{pmatrix}$$

In other words, the image of a point shifted over (t5) has the same form as t4--it is just offset by the shift.

Output image "response" to an arbitrary image g

Let's see how **g** gets transformed to an output image **f**, through a transformation matrix **T**.

We'll generalize things a bit. Let **u1, u2, u3**, ... be the shifted test images where the bright pixel is at location 1, 2, ...:

$$\mathbf{u1} = \begin{pmatrix}1\\0\\0\\0\\0\\0\\0\\0\end{pmatrix}, \mathbf{u2} = \begin{pmatrix}0\\1\\0\\0\\0\\0\\0\\0\end{pmatrix}, \ldots$$

This is the familiar Cartesian basis set for vectors.

Using our linear intensity-based model of image patterns, we can express the (arbitrary) input image **g** as a weighted linear sum of the test images:

$$\mathbf{g} = \begin{pmatrix}g1\\g2\\g3\\g4\\g5\\g6\\g7\\g8\end{pmatrix} = g1\begin{pmatrix}1\\0\\0\\0\\0\\0\\0\\0\end{pmatrix} + g2\begin{pmatrix}0\\1\\0\\0\\0\\0\\0\\0\end{pmatrix} + \ldots = g1\ \mathbf{u1} + g2\ \mathbf{u2} + \ldots$$

The weights are just the intensities of the pixels.

Assuming that the transformation **T[ ]** is linear, then the following holds:

f = **T[g]** = **T**[g1 **u1** + g2 **u2** + ...] = g1 **T[u1]** + g2 **T**[**u2]** + ...= g1 **t1** + g2 **t2** + ...

Each **T[ui], i = 1, 2,...** is the blurry image of the input test images, **ui**.

The transformation **T** can be represented by a matrix that multiplies **g**. Assuming shift-invariance, we can line up all the **ti** vectors in columns to represent the output as matrix **T** times input **g**:

$$\mathtt{f} = \begin{pmatrix}\mathtt{t1} & \mathtt{t8} & \mathtt{t7} & \mathtt{t6} & \mathtt{t5} & \mathtt{t4} & \mathtt{t3} & \mathtt{t2}\\ \mathtt{t2} & \mathtt{t1} & \mathtt{t8} & \mathtt{t7} & \mathtt{t6} & \mathtt{t5} & \mathtt{t4} & \mathtt{t3}\\ \mathtt{t3} & \mathtt{t2} & \mathtt{t1} & \mathtt{t8} & \mathtt{t7} & \mathtt{t6} & \mathtt{t5} & \mathtt{t4}\\ \mathtt{t4} & \mathtt{t3} & \mathtt{t2} & \mathtt{t1} & \mathtt{t8} & \mathtt{t7} & \mathtt{t6} & \mathtt{t5}\\ \mathtt{t5} & \mathtt{t4} & \mathtt{t3} & \mathtt{t2} & \mathtt{t1} & \mathtt{t8} & \mathtt{t7} & \mathtt{t6}\\ \mathtt{t6} & \mathtt{t5} & \mathtt{t4} & \mathtt{t3} & \mathtt{t2} & \mathtt{t1} & \mathtt{t8} & \mathtt{t7}\\ \mathtt{t7} & \mathtt{t6} & \mathtt{t5} & \mathtt{t4} & \mathtt{t3} & \mathtt{t2} & \mathtt{t1} & \mathtt{t8}\\ \mathtt{t8} & \mathtt{t7} & \mathtt{t6} & \mathtt{t5} & \mathtt{t4} & \mathtt{t3} & \mathtt{t2} & \mathtt{t1}\end{pmatrix}\begin{pmatrix}\mathtt{g1}\\\mathtt{g2}\\\mathtt{g3}\\\mathtt{g4}\\\mathtt{g5}\\\mathtt{g6}\\\mathtt{g7}\\\mathtt{g8}\end{pmatrix} = \mathtt{T.g}$$

For mathematical convenience, we are using "circulant" boundaries, i.e. that wrap around. The practical justification is that images are usually big, and if the blur small relative to the size of the image, then the boundaries don't contribute much to the center of the image.

A matrix for which the values along diagonals are the same is called a Toeplitz matrix. A circulant matrix is a special case of a Toeplitz matrix in which each subsequent row is a copy of the previous row but shifted one element to the right.

■ Shift-invariance and symmetric transformations, T

Let's take our model one step further. In addition to shift-invariance, we might also expect an optical system like the eye to show symmetry, i.e. the PSF is radially symmetric, or in 1-D our transformation matrix becomes:

$$\begin{pmatrix} t1 & t2 & t3 & t4 & t5 & t6 & t7 & t8 \\ t2 & t1 & t2 & t3 & t4 & t5 & t6 & t7 \\ t3 & t2 & t1 & t2 & t3 & t4 & t5 & t6 \\ t4 & t3 & t2 & t1 & t2 & t3 & t4 & t5 \\ t5 & t4 & t3 & t2 & t1 & t2 & t3 & t4 \\ t6 & t5 & t4 & t3 & t2 & t1 & t2 & t3 \\ t7 & t6 & t5 & t4 & t3 & t2 & t1 & t2 \\ t8 & t7 & t6 & t5 & t4 & t3 & t2 & t1 \end{pmatrix}$$

One of the benefits of the symmetric shift-invariant properties of the eye's optics is that we can model the transformation as a symmetric toeplitz matrix. As we will see later, these properties suggest an alternative basis set to represent images. Rather than represent an image as a linear combination of points (i.e. the above u's), images will be represented as a linear combination of *spatial sinewave gratings*. We'll see how the symmetric shift-invariant case provides a particularly simple example of Fourier analysis/synthesis.

So in standard *Mathematica* format, the above matrix calculation is written as:

```
{{t1, t8, t7, t6, t5, t4, t3, t2}, {t2, t1, t8, t7, t6, t5, t4, t3},
  {t3, t2, t1, t8, t7, t6, t5, t4}, {t4, t3, t2, t1, t8, t7, t6, t5},
  {t5, t4, t3, t2, t1, t8, t7, t6}, {t6, t5, t4, t3, t2, t1, t8, t7},
  {t7, t6, t5, t4, t3, t2, t1, t8}, {t8, t7, t6, t5, t4, t3, t2, t1}}.
 {g1, g2, g3, g4, g5, g6, g7, g8}
```

```
{g1 t1 + g8 t2 + g7 t3 + g6 t4 + g5 t5 + g4 t6 + g3 t7 + g2 t8,
 g2 t1 + g1 t2 + g8 t3 + g7 t4 + g6 t5 + g5 t6 + g4 t7 + g3 t8,
 g3 t1 + g2 t2 + g1 t3 + g8 t4 + g7 t5 + g6 t6 + g5 t7 + g4 t8,
 g4 t1 + g3 t2 + g2 t3 + g1 t4 + g8 t5 + g7 t6 + g6 t7 + g5 t8,
 g5 t1 + g4 t2 + g3 t3 + g2 t4 + g1 t5 + g8 t6 + g7 t7 + g6 t8,
 g6 t1 + g5 t2 + g4 t3 + g3 t4 + g2 t5 + g1 t6 + g8 t7 + g7 t8,
 g7 t1 + g6 t2 + g5 t3 + g4 t4 + g3 t5 + g2 t6 + g1 t7 + g8 t8,
 g8 t1 + g7 t2 + g6 t3 + g5 t4 + g4 t5 + g3 t6 + g2 t7 + g1 t8}
```

Our transformation matrix T is completely characterized by just one vector (which just gets repeated and shifted). In fact this filtering operation can be done using a convolution operator. **ListConvolve[ ]** gives us the same answer:

```
ListConvolve[{t1, t2, t3, t4, t5, t6, t7, t8},
 {g1, g2, g3, g4, g5, g6, g7, g8}, 1]
```

```
{g1 t1 + g8 t2 + g7 t3 + g6 t4 + g5 t5 + g4 t6 + g3 t7 + g2 t8,
 g2 t1 + g1 t2 + g8 t3 + g7 t4 + g6 t5 + g5 t6 + g4 t7 + g3 t8,
 g3 t1 + g2 t2 + g1 t3 + g8 t4 + g7 t5 + g6 t6 + g5 t7 + g4 t8,
 g4 t1 + g3 t2 + g2 t3 + g1 t4 + g8 t5 + g7 t6 + g6 t7 + g5 t8,
 g5 t1 + g4 t2 + g3 t3 + g2 t4 + g1 t5 + g8 t6 + g7 t7 + g6 t8,
 g6 t1 + g5 t2 + g4 t3 + g3 t4 + g2 t5 + g1 t6 + g8 t7 + g7 t8,
 g7 t1 + g6 t2 + g5 t3 + g4 t4 + g3 t5 + g2 t6 + g1 t7 + g8 t8,
 g8 t1 + g7 t2 + g6 t3 + g5 t4 + g4 t5 + g3 t6 + g2 t7 + g1 t8}
```

When we do a convolution (a correlation, as in ListCorrelate[*kern,list, offset*]), we can specify that *Mathematica* use a circular boundary, and place *kern* in the first row, but aligned relative to *list* by *offset* .

Characterizing a linear system by its response to an orthonormal basis set

■ Orthonormal basis sets

Given an image, we can think of it as being composed of a weighted sum of basic or "basis" images that together make up a "basis set" (e.g. the u's above). Each image is a vector of the same dimensionality as the image, but is presumed fixed. What is a "good" basis set? We have lots (∞) to choose from, and what is best will depend on what we are doing.

The simplest basis set (the Cartesian set used above) corresponds to our standard pixel representation for images, where the ith pixel is represented by a vector that is all zero's except for the ith element which is 1.

Suppose we have a different basis set (e.g. a "Walsh" set defined below). Let's call the ith basis image **w[i]**. Any 8-pixel image **g** can be written:

$$g = g[1]\begin{pmatrix} .3535 \\ .3535 \\ .3535 \\ .3535 \\ .3535 \\ .3535 \\ .3535 \\ .3535 \end{pmatrix} + g[2]\begin{pmatrix} +.3535 \\ -.3535 \\ -.3535 \\ +.3535 \\ +.3535 \\ -.3535 \\ -.3535 \\ +.3535 \end{pmatrix} + \ldots = g[1]\,\mathbf{w[1]} + g[2]\,\mathbf{w[2]} + \ldots$$

If the set $\{\boldsymbol{w}_i\} = \{\mathbf{w[i]}\}$ is *orthogonal* , then **w[i] . w[j]** = 0 for i≠j. Note that **w[i]** is the ith *vector* in the set--not a scalar.

If it is *normal* , then **w[i] . w[i]**=1. (The vector length of each basis vector is 1).

If the set is *complete* , then it spans 8-space in such a way that we can express any 8-d vector as a linear sum of these basis vectors.

Orthonormal (orthogonal and normalized) basis sets make it easier to do calculations.

Are there other orthonormal sets than the Carteisian or Walsh set? Again, there are lots, and again which one we use will depend on the job at hand.

Let $\{w_i\}$= {**w[i]**} be any orthonormal set. Then an arbitrary vector, **g** can be represented as a weighted sum of the orthonormal vectors, each weighted by the amount **g** projects onto w_i i.e, $(g.w_i)$.

The vector $\{g.w_1,\ g.w_2,\ g.w_3\ ...\}$ is sometimes called the spectrum of **g**. We have:

$$g = \sum (g.w_i)\ w_i \qquad (1)$$

In our first example (the above **u**'s), the dot product is really simple and $g.w_i$= g[[i]], i.e. the ith element of vector **g**.

■ Application to an "unknown" system

How can one go about measuring the matrix to characterize a linear system such as the optics of the eye? Or a neural system?

We won't presume symmetry or shift-invariance.

Suppose we have an unknown physical system, which we model as a linear system with matrix **T**:

```
T = Table[RandomReal[], {i, 1, 8}, {j, 1, 8}];
```

Goal: We would like to make a simple set of measurements that could characterize T in such a way that we could predict the output of T to any input.

This is the sort of task that engineers face when wanting to characterize, say a stereo amplifier (as a model linear system), so that the output sound can be predicted for any input sound.What kind of measurements would tell us what **T** is? As we did above, we could just "stimulate" the system with cartesian vectors {1,0,0,0,0,0,0,0},{0,1,0,0,0,0,0,0}, and so forth and collect the responses which would be the columns of **T**. This has two practical problems: 1) for a real physical system, such as your stereo, or a neuron in the eye, this would require stimulating it with a high-intensity audio or light intensity spike, which could damage what you are trying to study; 2) Characterizing the linear system by a matrix **T**, requires n^2numbers, where **n** is the input signal vector length--and **n** can be pretty big for both audio and visual systems. Problem 2) can be solved when **T** is symmetric and/or shift-invariant (because of the redunduncy, O(n) numbers are sufficient). Problem 1) can be addressed by showing that we can characterize **T** with any basis set--so we can pick one that won't blow out the physical system being tested.

As an example, consider the orthonormal set (**v**'s below) of Walsh functions. It has the advantage that the elements that contribute to the "energy", i.e. (the square of the length) are distributed across the vector.

```
Vectorlength[x_] := N[Sqrt[x.x]]
```

```
v1 = {1, 1, 1, 1, 1, 1, 1, 1}; w1 = v1/Vectorlength[v1];
v2 = {1,-1,-1, 1, 1,-1,-1, 1}; w2 = v2/Vectorlength[v2];
v3 = {1, 1,-1,-1,-1,-1, 1, 1}; w3 = v3/Vectorlength[v3];
v4 = {1,-1, 1,-1,-1, 1,-1, 1}; w4 = v4/Vectorlength[v4];
v5 = {1, 1, 1, 1,-1,-1,-1,-1}; w5 = v5/Vectorlength[v5];
v6 = {1,-1,-1, 1,-1, 1, 1,-1}; w6 = v6/Vectorlength[v6];
v7 = {1, 1,-1,-1, 1, 1,-1,-1}; w7 = v7/Vectorlength[v7];
v8 = {1,-1, 1,-1, 1,-1, 1,-1}; w8 = v8/Vectorlength[v8];
```

Consider an arbitrary 1-D image **g**:

```
g = {2,6,1,7,11,4,13, 29};
```

g can be written as the sum of its own projections onto the basis set:

```
(g.w1) w1  + (g.w2) w2  +(g.w3) w3  +(g.w4) w4  +
(g.w5) w5  + (g.w6) w6  +(g.w7) w7  +(g.w8) w8
```

```
{2., 6., 1., 7., 11., 4., 13., 29.}
```

Suppose we now do an "experiment" to find out how **T** transforms the vectors of our basis set:, and we put all of these transformed basis elements into a new set of vectors **newW[[i]]. newW** is a matrix for which each column is the response of **T** to a basis vector.

```
newW = Transpose[{T.w1,T.w2,T.w3,T.w4,T.w5,T.w6,T.w7,T.w8}];
```

Note that **newW** is an 8x8 matrix. So how can we calculate the output of **T**, given **g** without actually running the input through **T**?

By the principle of linearity, we can also calculate the output by finding the "spectrum" of **g**, and then scaling each of the transformed basis elements by the spectrum and adding them up:

$$T.g = T.\left\{\sum (g.w_i)\, w_i\right\} = \sum (g.w_i)\, T.w_i \qquad (2)$$

```
(g.w1) T.w1 + (g.w2) T.w2 + (g.w3) T.w3 + (g.w4) T.w4 +
(g.w5) T.w5 + (g.w6) T.w6 + (g.w7) T.w7 + (g.w8) T.w8
```

```
{26.6453, 30.0773, 42.8788, 29.0873, 48.1378, 42.0447, 22.1362, 54.9271}
```

■ ..but we don't have to run the basis vectors through T each time

Of course, we have already done our "experiment in the lab", so we know what the transformed basis vectors {**T.w1, ...**} are, we stored them as columns of the matrix **newW**. We can calculate what the spectrum (**g.w**i) is, so the output of **T** is:

```
(g.w1) Transpose[newW][[1]] + (g.w2) Transpose[newW][[2]] + (g.w3) Transpos
(g.w4) Transpose[newW][[4]] + (g.w5) Transpose[newW][[5]] + (g.w6) Transpos
(g.w7) Transpose[newW][[7]] + (g.w8) Transpose[newW][[8]]
```

```
{26.6453, 30.0773, 42.8788, 29.0873, 48.1378, 42.0447, 22.1362, 54.9271}
```

The point is that once we've measured the output to the complete set of basis images, we just store those image (just 8 of them in this simple example).

Then we can get the output to any arbitrary input image by computing the spectrum of the input and using those values as weights for the stored images.

The PSF was a special case of the "stored image", but was simpler because of shift-invariance. Here we are not assuming shift-invariance.

Check our answer: If we "go back to the lab" and run the input through **T** we get:

```
T.g
```

```
{26.6453, 30.0773, 42.8788, 29.0873, 48.1378, 42.0447, 22.1362, 54.9271}
```

■ Same thing in more concise notation

Let the basis vectors be the rows of a matrix **W**:

```
W = {w1, w2, w3, w4, w5, w6, w7, w8};
```

So again, we can project **g** onto the rows of **W**, and then reconstitute it in terms of **W** to get **g** back again:

```
(W.g).W
```

```
{2., 6., 1., 7., 11., 4., 13., 29.}
```

```
newW.W.g
```

```
{33.4443, 24.4767, 33.6792, 46.3589, 26.9546, 20.6934, 37.5313, 33.1454}
```

What if the choice of basis set is the set of eigenvectors of T?

We are now going to introduce an elegant idea that will simplify our analysis of linear shift-invariant systems. Our illustration relies on a result from linear algebra that says the eigenvectors of a symmetric matrix are real and orthogonal. In general, one can relax the assumption of symmetry.

Eigenvectors and eigenvalues--short review

Eigenvectors

An eigenvector, **x**, of a matrix, **A**, is vector that when you multiply it by **A**, you get an output vector that points in the same direction as **x**:

$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$

where λ is a scalar that adjusts the length change of **x**.

The *Mathematica* function **Eigenvectors[A]** returns the eigenvectors of matrix **A** as the rows of a matrix, which we'll call **eig**:

```
A = {{1,2},{3,4}};
eig = Eigenvectors[A]
```

$$\begin{pmatrix} \frac{1}{6}\left(-3-\sqrt{33}\right) & 1 \\ \frac{1}{6}\left(-3+\sqrt{33}\right) & 1 \end{pmatrix}$$

We can verify that **eig[[1]]** and **A.eig[[1]]** lie along the same direction by taking the dot product of the unit vectors pointing in the directions of each:

```
normalize[x_] := x/Sqrt[x.x];
normalize[eig[[1]]].normalize[A.eig[[1]]];
N[%]
```

$-1.$

Eigenvalues

The eigenvalues are given by:

```
Eigenvalues[A]
```

$$\left\{\frac{1}{2}\left(5-\sqrt{33}\right), \frac{1}{2}\left(5+\sqrt{33}\right)\right\}$$

Eigenvalues and eigenvector elements do not have to be real numbers. They can be complex, that is an element can be the sum of a real and imaginary number. In *Mathematica*, imaginary numbers are represented by multiples (or fractions) of **I**, the square root of -1:

```
Sqrt[-1]
Sqrt[-1]//StandardForm
```

$$i$$

```
I
```

```
B = {{1,2},{-3,4}};
Eigenvalues[B]
```

$$\left\{\frac{1}{2}\left(5-i\sqrt{15}\right),\ \frac{1}{2}\left(5+i\sqrt{15}\right)\right\}$$

We've seen how linearity provides us with a method for characterizing a linear system in terms of the responses of the system to the basis vectors. The problem is that if the input signals are long vectors, say with dimension 40,000, then this set of basis vector responses is really big--1.6×10^9 numbers.

Construct a symmetric matrix transformation, **T**, with eigenvectors $\{\boldsymbol{w_i}\}$. Then,

$$\boldsymbol{T.w_i}=\lambda_i\ \boldsymbol{w_i}$$

The eigenvectors of a symmetric matrix are orthogonal and real-valued (very nice). Now we use those same eigenvectors of **T** to represent the input **g**. If the elements of the basis set are the eigenvectors of **T**, then the transformation of any arbitrary input vector **g** is given by:

$$\boldsymbol{T.g = T.\left\{\sum (g.w_i)\ w_i\right\} = \sum (g.w_i)\ T.w_i = \sum \alpha_i\ \lambda_i\ w_i} \qquad \textbf{(3)}$$

Where the α_i are the projections of **g** onto each eigenvector. Having the eigenvectors of **T** enables us to express the input and output of **T** *in terms of the same basis set*--the eigenvectors. All **T** does to the input is to scale its projection onto each eigenvector by the eigenvalue for that eigenvector. The set of these eigenvalues, $\{\lambda_i\}$ is sometimes called the *modulation transfer function* because it describes how the amplitude of the eigenvectors change as they pass through **T**.

Linear systems analysis is the foundation of Fourier analysis, and is why it makes sense to characterize your stereo amplifier in terms of frequency response. But your stereo isn't just any linear system--it has the special property that if you input a sound at time t and measure the response, and then you input the same sound again at a later time, you get the same response, except of course that is is shifted in time. It is said to be a shift-invariant system. The eigenvectors of a shift-invariant system are sinusoids. (The eigenvectors of the symmetric matrix are sinusoids, not just because the matrix was symmetric, but also because each row of the matrix was a shifted version of the previous row--the elements along any given diagonal are identical-- a symmetric Toeplitz matrix.)

Sinewave inputs are the eigenvectors of your stereo system. The dimensionality is much higher--if you are interested in frequencies up to 20,000 Hz, your eigenvector for this highest frequency would have least 40,000 elements--not just 8!

This kind of analysis has been applied not only to physical systems, but to a wide range of neural sensory systems. For the visual system alone, linear systems analysis has been used to study the cat retina (Enroth-Cugell and Robson, 1964), the monkey visual cortex, and the human contrast sensivity system as a whole (Campbell and Robson, 1968).

Much empirical analysis has been done using linear systems theory to characterize neural sensory systems, and other neural systems such as those for eye movements. It works wonderfully as long as the linear system approximation holds. And it does do quite well for the lateral eye of the limulus, X-cells and P-cells of the mammalian visual system, over restricted ranges for so-called "simple" cells in the visual cortex, among others. The optics of the simple eye is another example of an approximately linear system. Many non-linear systems can be approximated as linear systems over smooth subdomains.

■ Summary of frequency analysis

In summary, if **T** has n distinct orthogonal eigenvectors, $\mathbf{e}_i$, and known eigenvalues, λ_i, then we have a particularly easy way to calculate the response to an input **g**:

Step 1: Project **g** onto eigenvectors of **T** using the dot product: $\mathbf{g}.\mathbf{e}_i$

Step 2: Scale each $\mathbf{g}.\mathbf{e}_i$ by the eigenvalue of $\mathbf{e}_i$: $\lambda_i \mathbf{g}.\mathbf{e}_i$

Step 3: Scale each $\mathbf{e}_i$ by $\lambda_i \mathbf{g}.\mathbf{e}_i$

Step 4: Sum these up. That's the response of **T** to **g** : $\mathbf{f} = \sum_i (\lambda_i \, \boldsymbol{g}.\boldsymbol{e}_i) \, \mathbf{e}_i$

The Modulation Transfer Function (MTF) of the human eye

Gratings: Eigenfunctions of linear shift-invariant optics

Now a point of light, a face, a square wave grating do not maintain the same form through optical transformation, because aberrations and diffraction blur the edges. However, for a spatially homogeneous optical system, a sine-wave grating does keep the same form (consistent with the shift-invariant properties of a linear system). The figure below shows pictures of a square-wave and a sine-wave (you need 256 gray-levels to see the sine-wave grating). What happens to the form of the two intensity patterns when you blur your eyes?

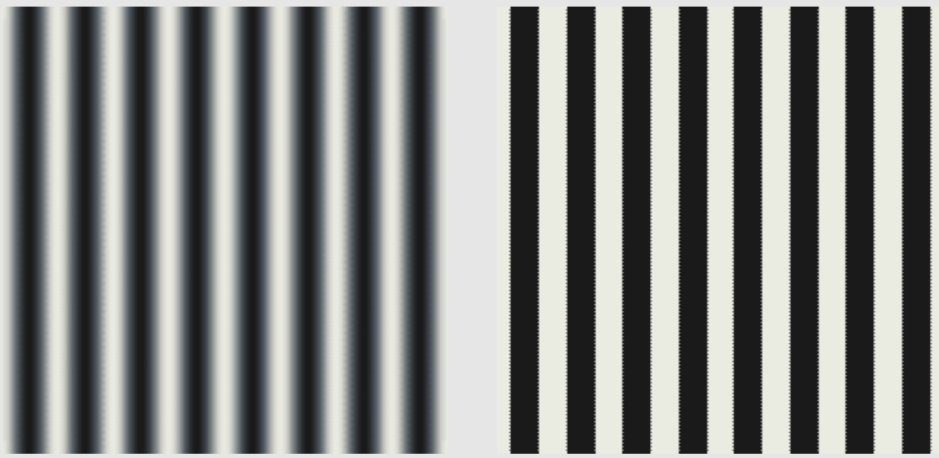
■ Blurring: Sine vs. square

```
Grating[x_, y_, fx_, fy_] := Cos[2 Pi (fx x + fy y)];
Square[x_, y_, fx_, fy_] := Sign[Grating[x, y, fx, fy]];
```

```
gsine = DensityPlot[0.25 * Grating[x, y, 4, 0], {x, -1, 1}, {y, -1, 1},
   PlotPoints → 64, Mesh → False, Frame → False, PlotRange → {-1, 1},
   ColorFunction → "GrayTones"];
```

```
gsquare = DensityPlot[0.25 * Square[x, y, 4, 0], {x, -1, 1},
   {y, -1, 1}, PlotPoints → 64, Mesh → False, Frame → False,
   PlotRange → {-1, 1}, ColorFunction → "GrayTones"];
```

```
Show[GraphicsRow[{gsine, gsquare}]]
```



Sine-wave gratings are eigenfunctions of linear shift-invariant systems. Our complete basis set (a collection of eigenfunctions, $\{\boldsymbol{b}_i\}$) will have to be built out of collection of gratings of various frequencies (combinations of fx and fy produce different orientations) and phases (ϕ). So if we can represent our images in terms of sums of sine-wave gratings, then we can model how a linear shift-invariant optical system distorts the image. This kind of representation of an image is called *Fourier Analysis*. Because we are talking about intensity as a function of space (we'll get to time later), this kind of Fourier analysis is called *spatial frequency analysis* of images.

Fourier analysis

■ Representing images: The formal case, continous variable case -> Fourier transform

A complete formal analysis for continuous images is the analog to the discrete model we introduced above, where frequencies can take on continuous values. The basic formalism is similar, with summation replaced by integration, and complex variables are used (to compactly and efficiently deal with phase relationships between the gratings). In one spatial dimension, the forms are given by:

$$\text{Fourier transform: } F(f_x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \text{image}(x)\, e^{i 2\pi f_x x}\, dx,$$

$$\text{Inverse fourier transform: image}(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} F(f_x)\, e^{-i 2\pi f_x x}\, df_x$$

Mathematica has built-in functions FourierTransform[] and InverseFourierTransform[] to deal with symbolic manipulations for the continuous case. Fourier[] handles the discrete numerical calculations. One theoretical advantage of Fourier transforms is that, in contrast to Fourier series, one is not limited to periodic images.

■ Actual practice: hybrid discrete-continuous

But in actual practice, we might use some discrete collection of continuous gratings (Fourier series) to approximate the image as:

$$\text{image} = \sum_i a_i \boldsymbol{b}_i + b_0$$

$$a_i \boldsymbol{b}_i = a_i \text{Cos}[2\ \text{Pi}\ (\text{fx}_i\ x + \text{fy}_i\ y) + \phi_i]$$

b_0 is the average background light level, and $a_i \left(= \text{a}_\text{i}^{\text{in}}\right)$ is the amplitude of the grating. Often, we talk only about the *contras* t of the grating :

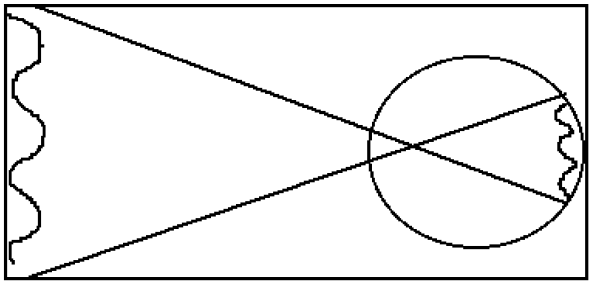
$$\text{contrast} = \frac{a_i}{b_0}.$$

The reason is psychophysical and physiological--the human visual system is largely invariant to average background level--contrast corresponds well to the relative variations in brightness that you see in an image pattern, as well as to the neural variations transmitted from the retina to the brain. So for convenience, we can drop the $b_0$term which is constant, and if there is negligible absorption by the optics, remains unchanged anyway. The set $\{a_i\}$ is the amplitude spectrum of the image, and $\{\phi_i\}$ the phase spectrum. Both are usually plotted as a function of frequency (here indexed by i).

The Modulation Transfer Function (MTF)

■ Characterizing on optical system

In order to characterize an optical system, the idea is to measure how the amplitude (contrast) of the eigenfunctions (sine wave gratings) changes as a function of spatial frequency. The ratio of output grating amplitude to input amplitude (as a function of spatial frequency) is called the *modulation transfer function* of the optical system (e.g. eye):

$a_i^{in}\ \mathbf{b}_i$  $a_i^{out}\ \mathbf{b}_i$

$a_{f_x}^{in} \sin 2\pi f_x x$ $a_{f_x}^{out} \sin(2\pi f_x x + \phi)$

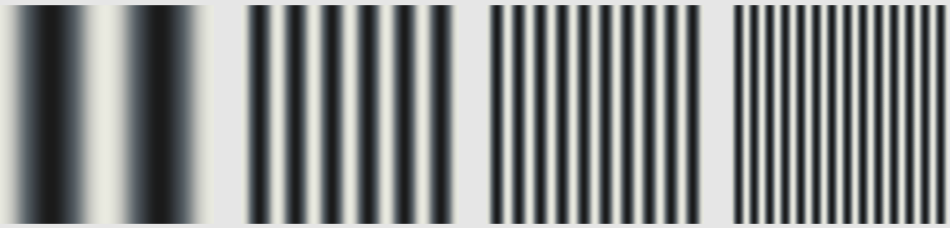
$$MTF_i = \frac{a_i^{out}}{a_i^{in}}$$

Of course, the spatial scale will change with optical minification or magnification. We will assume the scale is the same. We can always put the right scale back in with a suitable mapping x-> scale factor * x.

Suppose the input contrast amplitude is fixed, say $a_i^{in} = 1$. Then a_i^{out} is the MTF. In one dimension, we'll write $a(f) = a_i^{out}$.

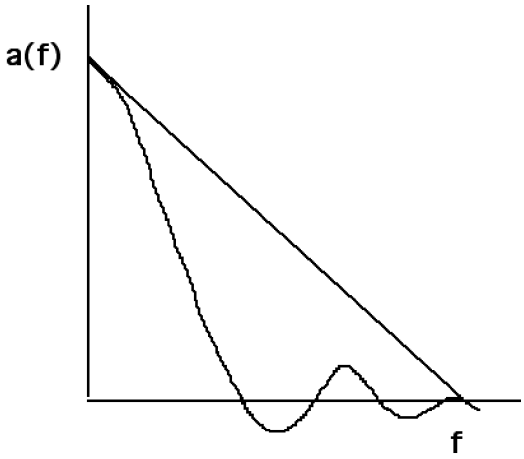
Can you guess what a typical modulation transfer function would do the the following gratings, going from low to high spatial frequencies?

```
g = Table[DensityPlot[0.5` Grating[x, y, fx, 0], {x, -1, 1},
    {y, -1, 1}, PlotPoints → 128, Mesh → False, Frame → False,
    PlotRange → {-1, 1}, ColorFunction → "GrayTones"], {fx, 1, 7, 2}];
Show[GraphicsRow[{g[[1]], g[[2]], g[[3]], g[[4]]}]]
```



■ Some answers

Here are two possible MTFs:



The form of these curves makes intuitive sense. The amplitudes of low spatial frequency gratings remains largely unchanged going through the optics. On the other hand, at high spatial frequencies, the light and dark bars are close together and get smeared out by aberrations. You might think that the negative part of the MTF is a little unlikely. In fact, it is quite common. An example is when your eye is out of focus. The contrast at those spatial frequencies corresponding to negative MTF undergo phase reversal. This is called spurious resolution. You can't really notice this with ordinary images, but can see it with sinewave grating images.

■ Visualizing your own contrast sensitivity function (CSF)

Warning--executing the next cell can take some time.

```
CSF[x_, y_] := 127.5 e^(-y/0.125) Cos[2 π e^(x/4) x] + 127.5;
csfg = DensityPlot[CSF[x, y], {x, 0, 6}, {y, 0, 1}, PlotPoints → 512,
   PlotRange → {1, 254}, Mesh → False, Axes → False, Frame → False,
   ColorFunction → "GrayTones"]
```

There is an apparent fall-off in contrast sensivity for high frequencies; however, there is also a drop-off at low frequencies. We'll focus on the high spatial frequency fall-off and return to the reasons for the low frequency loss later.

■ To re-cap

So the sine waves are the known basis functions, the MTF {ti's} is measured, and the spectrum (Li's) can be calculated from an image using a fourier transform subroutine. Then our discrete-hybrid analog of what we introduced at the very beginning of this lecture says that knowing the human eye's MTF, we can calculate the input image **g**,

$$\mathbf{g} = \sum_i (\mathbf{g}.\mathbf{b}_i)\, \boldsymbol{b}_i = \sum a_i^{in}\, \boldsymbol{b}_i$$

retinal image f, as:

$$\mathrm{f} = \sum_i a_i^{\mathrm{in}} \lambda_i \boldsymbol{b_i}$$

where

$$\lambda_i = \frac{a_i^{out}}{a_i^{in}}, \text{ is the MTF.}$$

■ For the advanced...

For the continuous case, the image at the retina could be represented as an integral of the spectrum, the modulation transfer function, and the basis functions (or eigenfunctions):

$$\mathrm{r(x,y)}=\int g(f_x, f_y)\, T(f_x, f_y)\, b(x, y; f_x, f_y)\, d f_x d f_y$$

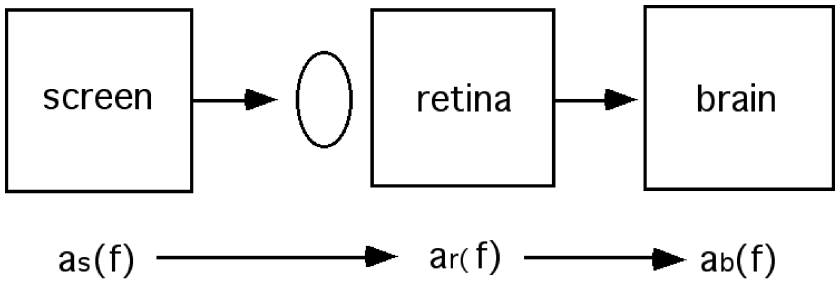
where $L(f_x, f_y)$ is the (complex-valued) spectrum, $T(f_x, f_y)$ the optical transfer function (takes into account phase shifts, that the MTF doesn't), and $\{b(x, y; f_x, f_y)\}$ are the complex-valued eigenfunctions, $\{e^{-2\pi i(\mathrm{x}f_x+\mathrm{y}f_y)}\}$.

The clever experiment of Campbell and Green

■ The idea--bypass the optics to measure the "eye-brain" response

How can we measure the MTF of a real eye? The first measurements were made in the late 1950's and 1960's using several techniques. One of the cleverest was developed by Campbell and Green in 1966.

Let us break the way contrast of a grating gets processed into two part



Our goal is to measure the MTF, but to do this would require getting access to ar(f). This seems like a tough problem. Campbell and Green came up with the following solution. Rather than measuring the MTF directly, they measured two other functions. One is called the contrast sensitivity function (CSF) of the human eye. This is measured by having subjects adjust the contrast of a grating until it is just disappearing, that is where the brain's "response" is always k. One then plots up the reciprocal of contrast (called sensitivity) as a function of spatial frequency. Such a graph is shown below. Using our terminology

$$CSF(f) = \frac{k}{a_s(f)}$$

The second function they measured, we will call the "brain's contrast sensitivity function", or BTF for short. The idea was to present a grating on the retina whose contrast was unaffected by the optics of the eye, and then measure the contrast sensitivity in the same way as for the CSF

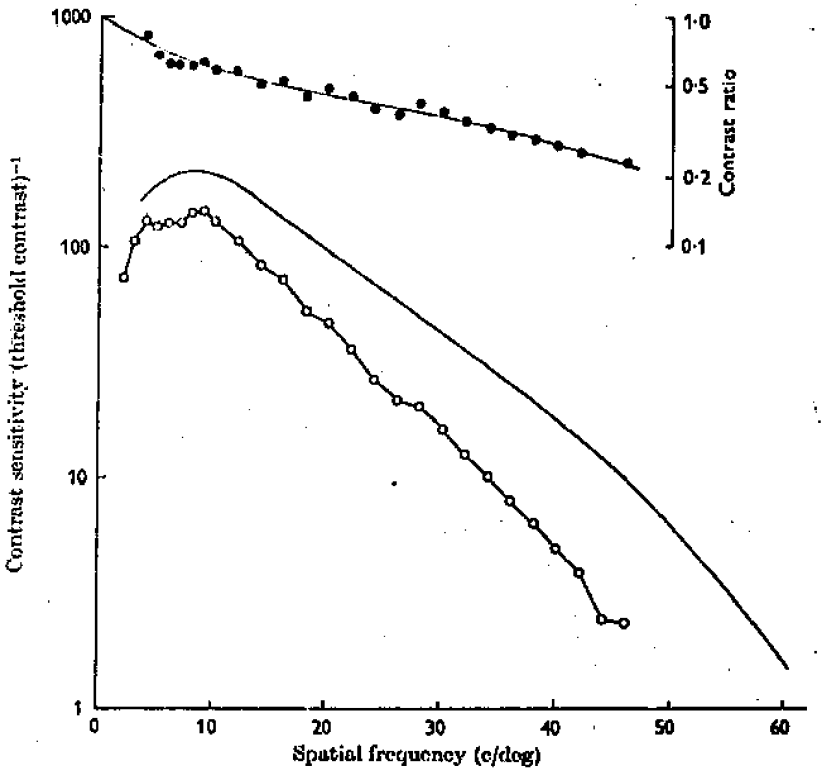
$$\text{"BTF"} = \frac{k}{a_r(f)}$$

$$\text{CSF} = \text{MTF } x \text{ BTF} = \frac{a_r}{a_s} \cdot \frac{k}{a_r} = \frac{k}{a_s}$$

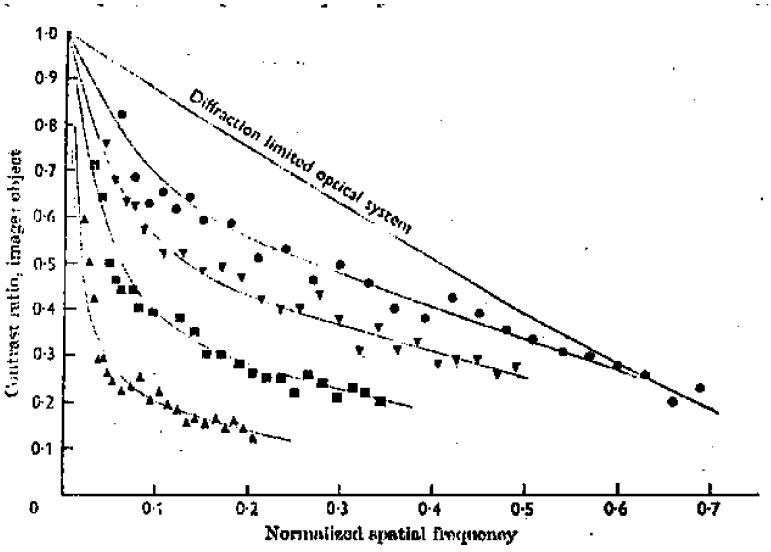
$$\text{MTF} = \frac{a_r}{a_s}$$

If we know the CSF and the BTF, we can get the MTF. But how can the BTF be measured? The solution was to image two points of coherent laser light in the pupil of the eye. These two point sources in the pupil produce a sinusoidal interference pattern on the retina of the eye. In fact, the pupil corresponds to the fourier plane of the retina--the fourier transform of a pair of delta functions is a sinusoid, because of constructive and destructive interference (see Appendix and the cosine/dirac delta transform pairs). Further, the shift theorem says that if the points are moved apart in the pupil, the frequency of the grating gets higher, because the retinal pattern shrinks as the pupil fourier pattern expands. Campbell and Green were able to have subjects modulate the contrast of the interference pattern to find the BTF, or contrast sensitivity as a function of spatial frequency with the optics effectively bypassed

■ The measured CSF and MTF of the human eye



The solid line shows the BTF--we require less contrast to see a grating if the optics are bypassed. The line with the data points shows the CSF. The MTF is the CSF/BTF and is shown in the upper panel

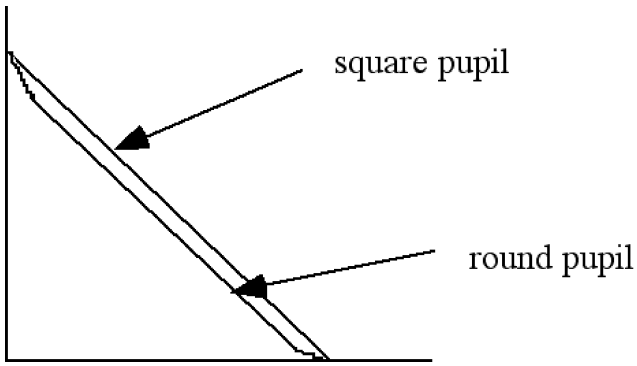


In this figure, the estimated MTFs are plotted for various pupil sizes (2, 2.8,3.8, and 5.8 mm). The curves are normalized so that the diffraction limit would correspond to a frequency of 1. The solid line is the diffraction limit. You can see that for high frequencies, the eye with a 2mm pupil is essentially diffraction limited. However, aberrations (e.g. spherical) greatly affect contrast for big pupils.

The theoretical diffraction limit and receptor spacing

■ Diffraction limit

What is the theoretical diffraction limit? The MTF can be calculated by computing the fourier transform of the Airy disk function. For round and square pupils, the curves look roughly like



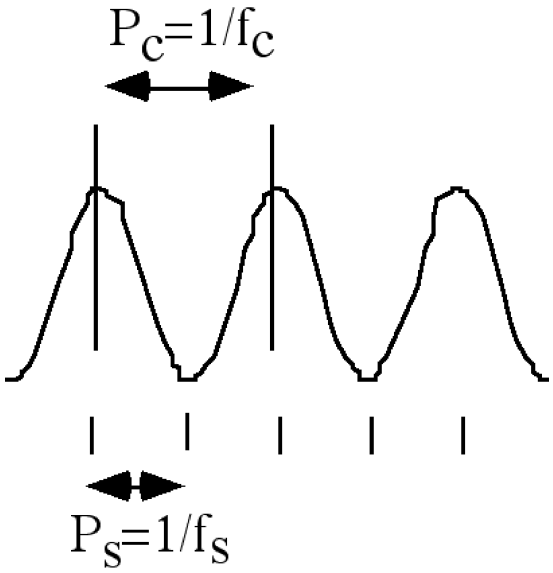
(Note: these are hand sketched, and are meant mainly to show the steady monotonic drop in contrast transfer).

The high frequency cut-off is:

$$f_c = \frac{a}{\lambda} \text{ cycles/radian } = 3,603.6 \text{ c/rad for } a = 2\text{mm},\ \lambda = 555\text{nm} = 63 \text{ c/degree}$$

One of the advantages of spatial frequency analysis of the optics is that it gives us a precise description of the information that is lost--i.e. spatial frequencies higher than the cut-off frequency. (There have been efforts to recover this missing high frequency information in optics using analytic continuation, but here one is again thwarted by noise). The period Pc = .016 deg = 1/63. Now recall that the cone spacing = .008 deg. How well does receptor sampling period, Ps, match the period of the highest frequency in a broad band image spectrum?

The Nyquist limit



This is actually a perfect match (perhaps too perfect?). The Whittaker-Shannon theorem says that one can perfectly reconstruct a continuous band-limited function if the discrete sampling rate, fs, is at least twice that of the highest frequency in the spectrum

$$f_s > 2f_c \qquad P_s < \frac{P_c}{2}$$

The smallest sampling frequency that one can get by with is twice the highest frequency in the spectrum of the signal. This smallest frequency is called the Nyquist rat

$$f_s = f_{Nyquist} = 2 f_c$$

If the sampling frequency is less, then aliasing results. Aliasing produces moiré patterns. (See Williams, 1986; Coletta et al., 1990 for examples of the effects of aliasing in human vision). Is aliasing bad for neural processing of images? Allan Snyder once took me down in the basement of the Physiological Laboratories at the University of Cambridge to show me a Garter snake in a terrarium. With a simple ophthalmoscope, we peered into the tiny eye of the snake. We saw the photoreceptor mosaic in striking detail and regularity--the optics were clearly better than the sampling frequency.

We now have the tools to calculate an upper bound on the information capacity of the eye. The modulation transfer function effectively limits the size of a resolution cell, as specified by diffraction and the Whittaker-Shannon sampling theorem, and photon statistics limit the number of distinguishable levels. There are formulas that combine these two factors to give precise measures of the limits to optical capacity in terms of bits.

■ Why aren't the optics better?

Making the pupil bigger leads to problems with aberrations (spherical).

Making the pupil smaller leads to increasing diffraction blur.

If the pupil was bigger, and aberration could be reduced, we would have to pack the cones more tightly to adequately sample the higher spatial frequencies passed by the optics. It is thought that there may be a physical wave-guide limit of 1-2 micrometers to receptor diameter, across animals of varying acuity (eagle, hawk or human).

Next time

Linear systems and visual neurons

->Multi-resolution, wavelets

->A model of the spatial filtering properties of neurons in the primary visual cortex

Appendices

Generating a Contrast Sensitivity Function (CSF) demo

```
CSF[x_, y_] := 127.5` e^-(y/0.125`) Cos[2 π e^(x/4) x] + 127.5`;
csfg = DensityPlot[CSF[x, y], {x, 0, 6}, {y, 0, 1}, PlotPoints → 512,
  PlotRange → {1, 254}, Mesh → False, Axes → False, Frame → False]
```

The nominal physical contrast is constant across any horizontal straight line; however, the subjective appearance of the boundary between the easily visible and nearly invisible contrast transition appears like an upside-down U-shaped function, ⋂.

■ Model CSF

```
csfpg = Plot[-e^-fx + e^-(fx/2), {fx, 0, 8}, Axes → False]
```

Mathematica has built-in symbolic functions: FourierTransform[ ] and

InverseFourierTransform[ ]

$$\text{Fourier transform: } F(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(t)\, e^{i\omega t}\, dt,$$

$$\text{Inverse fourier transform: } f(t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} F(\omega)\, e^{-i\omega t}\, d\omega$$

■ **1. The fourier transform of the inverse fourier transform of g[f] is g[f].**

g[f]=

```
FourierTransform[InverseFourierTransform[g[w], w, x], x, w]
```

■ **2. Convolution theorem: f*g = InverseFourierTransform[FourierTransform[f[x],x,w] FourierTransform[g[x],x,w],w,x]**

```
InverseFourierTransform[
 FourierTransform[g[x], x, w] FourierTransform[h[x], x, w], w, x]
```

```
InverseFourierTransform[
 FourierTransform[g[x], x, f] FourierTransform[h[x], x, f], f, x]
```

■ **Fourier transform of a delta function is constant**

```
FourierTransform[DiracDelta[x], x, f]
```

$$\frac{1}{\sqrt{2\pi}}$$

■ **What is the Fourier transform of Cos[x]?**

```
FourierTransform[Cos[x], x, f]
```

$$\sqrt{\frac{\pi}{2}}\ \text{DiracDelta}[-1+f] + \sqrt{\frac{\pi}{2}}\ \text{DiracDelta}[1+f]$$

■ **Some functions have the same shape in Fourier domain as in the spatial domain**

```
FourierTransform[Exp[-(x / σ)^2], x, f]
```

$$\frac{e^{-\frac{1}{4} f^2 \sigma^2} \sqrt{\sigma^2}}{\sqrt{2}}$$

Note that as the gaussian gets narrower in space, it gets broader in frequency.

References

Barlow, H. B. (1981). Critical Limiting Factors in the Design of the Eye and Visual Cortex. Proceedings of the Royal Society London B, 212, 1-34.

Coletta, N. J., Williams, D. R., & Tiana, C. L. M. (1990). Consequences of spatial sampling for human motion perception. Vision Research, 30(11), 1631-1648.

Campbell, F. W., & Green, D. (1966). Optical and retinal factors affecting visual resolution. Journal of Physiology (Lond.), 181, 576-593.

Gaskill, J. D. (1978). Linear Systems, Fourier Transforms, and Optics. New York: John Wiley & Sons.

He, S., & MacLeod, D. I. (1996). Local luminance nonlinearity and receptor aliasing in the detection of high-frequency gratings. *J Opt Soc Am A, 13*(6), 1139-1151.

Liang, J., & Williams, D. R. (1997). Aberrations and retinal image quality of the normal human eye. *J Opt Soc Am A, 14*(11), 2873-2883.

Smallman, H. S., MacLeod, D. I., He, S., & Kentridge, R. W. (1996). Fine grain of the neural representation of human spatial vision. *J Neurosci, 16*(5), 1852-1859.

Williams, D. R. (1986). Seeing through the photoreceptor mosaic. Trends in Neuroscience, 9(5), 193-197.