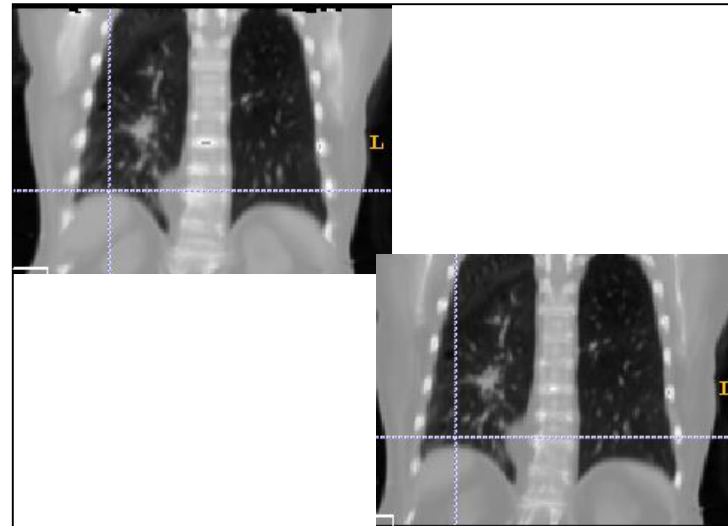
The Math Needed to Understand Image Processing

- Representation of images as Taylor series
  - Thus computation of image derivatives
- **Invariant operators: to shift, rotation, scale**
- **Shift-invariant, linear operators**
  - **Image representations consistent with these operators**
  - **Representation of images via orthogonal basis functions, esp. sinusoids (Fourier basis functions)**
  - **Convolution**
  - **Point and line spread functions and other convolution kernels**
  - **Understanding convolution and derivatives via Fourier basis functions**
- Representation of images as pixels or voxels: understanding sampling effects

Discrete Representations of Images



One image is a deformed version of the other

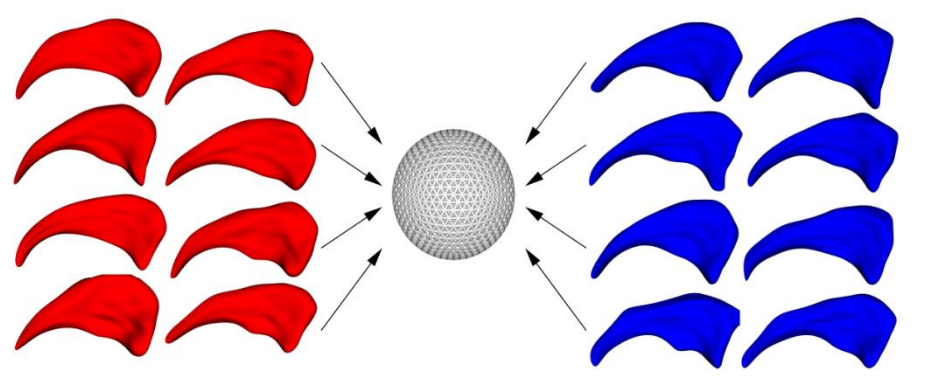
- Sampled
  - Pixels
  - Pixel displacements
    - Need for interpolation of intensities
  - Limiting damage of sampling will be a topic later
- Parametrized
  - **Global:** $\mathbf{I(x,y) = \sum_{k=1}^{n} a_k \Psi^k(x,y)}$; evaluable at any (x,y) the image representation is the n-vector <u>a</u>
    - For an error-free representation n = number of pixels or voxels
    - Frequently you want n << number of pixels or voxels, so issue is which representation allows the smaller values of n
  - Later topic: local over patches: linear combination of local basis
- Interpolation from sampled to parametrized via the $\mathbf{\Psi^k}$

Basis Functions for Parametrized Discrete Representations of Images

- Global vs. local:
  - **Global: $\mathbf{I(x,y) = \sum_k a_k \Psi^k(x,y)}$**; the representation is the vector $\underline{a}$
  - Later topic: local over patches: linear combination of local basis
- What basis functions $\Psi^k(x,y)$?
  - Choices
    - The pixel representation: for it $\mathbf{\Psi^k(x,y)}$ = 1 at k^{th} pixel, 0 elsewhere
      - This is very (too) local
    - We have seen Taylor as too local with $\Psi^k(x,y)$ = an *image* formed by a polynomial whose degree is non-decreasing as k increases: e.g., $(x-x_0)(y-y_0)$
    - Other choices specialized to operators being applied

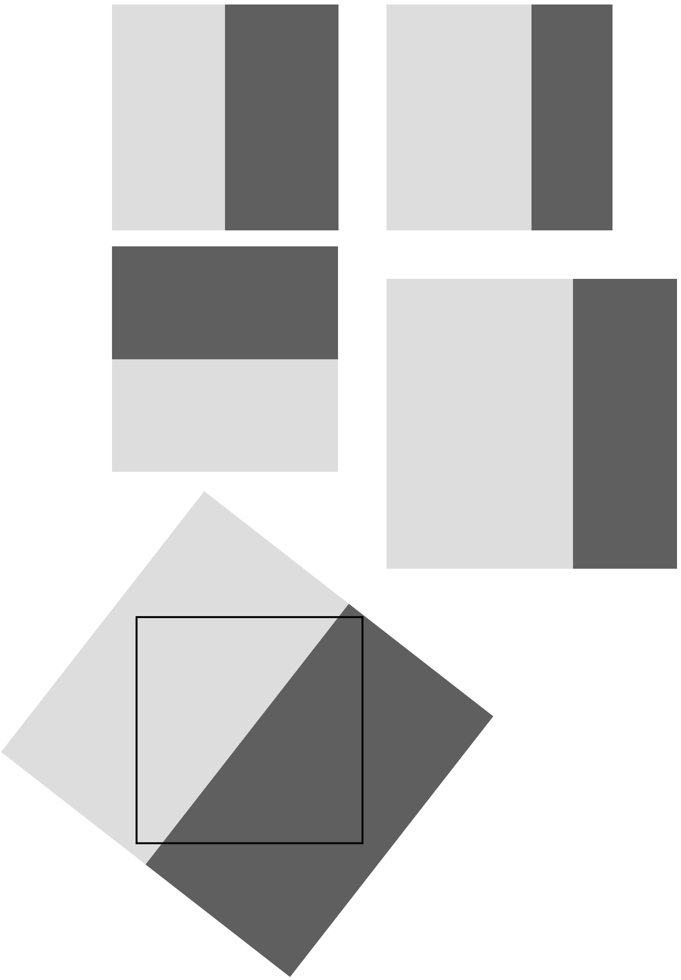
Basis Functions for Parametrized Discrete Representations of Images$^{cont.}$

- $\mathbf{I(x,y) = \sum_k a_k \Psi^k(x,y)}$;
- What basis functions $\Psi^k(x,y)$?
  - Issues
    - Ease of applying processing operators, e.g., shift-invariant, linear
    - How to handle different levels of detail?
    - How to handle level(s) of locality?
    - How get good approximation with few basis functions?
- Ideas are extendable to objects
    - Of boundary with $\Psi^\kappa(\theta,\phi)$
    - Of interior



Invariant operators: to shift, rotation, scale

- Let T be an operator on images
  - so let image J = T o I
  - Example of a T: averaging over a rectangle centered at each pixel
- Let G be a geometric operation on an image, such as shifting (translation), rotation, and scale change
  - so let image J = G o I
- T is said to be G-invariant (equivariant in math terminology) iff
  $\forall I \ [T \circ G \circ I = G \circ T \circ I]$
  - equivalently, $\forall I \ [T \circ I = G^{-1} \circ T \circ G \circ I]$
- “Invariant” is called “equivariant” in the math literature



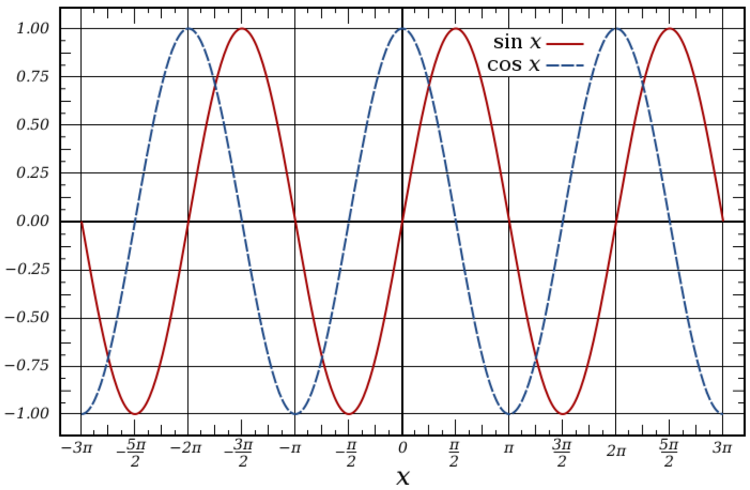
Basis Functions for Linear Operators

- Let **T** be a set of linear operators T_j on image functions $I(\underline{x})$
- Consider a basis $\{\Psi^k(\underline{x}), k=1, \ldots\}$ for functions $I(\underline{x})$
  - $I(\underline{x}) = \sum_k a_k \Psi^k(\underline{x})$
  - Let $J_j(\underline{x})$ be I processed by T_j, i.e., $J_j(\underline{x}) = T_j(I(\underline{x}))$
    - $J_j(\underline{x}) = \sum_m b_{jm} \Psi^m(\underline{x})$
    - But by linearity $T_j(I(\underline{x})) = T_j(\sum_k a_k \Psi^k(\underline{x})) = \sum_k a_k T_j(\Psi^k(\underline{x}))$
    - Let $T_j(\Psi^k(\underline{x})) = \sum_m c_{mjk} \Psi^m(\underline{x})$; there is cross-talk between the basis functions when T_j is applied
    - So $T_j(I(\underline{x})) = \sum_k a_k \sum_m c_{mjk} \Psi^m(\underline{x}) = \sum_m [\sum_k c_{mjk} a_k] \Psi^m(\underline{x})$
    - Thus $b_{jm} = \sum_k c_{mjk} a_k$; matrix multiplications $\underline{b}^j = C^j \underline{a}$
      - the el'ts of $\underline{a}$ and the rows & columns of C^j run over the basis functions
      - Inefficient and hard to understand
    - **We would thus like to avoid crosstalk**

Basis Functions for Linear Operators

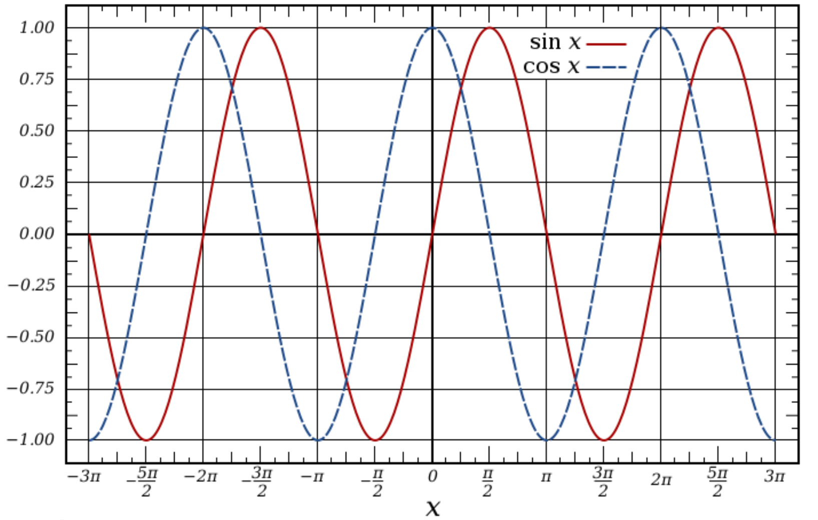
- Let T be a linear operator on images $I(\underline{x})$
- Consider a set of eigenimages $\Psi^k(\underline{x})$ of T
  - Definition of eigen-image of T: $T(\Psi^k(\underline{x})) = \lambda_k \Psi^k(\underline{x})$; no cross-talk
  - The $\Psi^k(\underline{x})$ span the space of $I(\underline{x})$
  - So if $I(\underline{x}) = \sum_k a_k \Psi^k(\underline{x})$, $T(I(\underline{x})) = \sum_k \lambda_k a_k \Psi^k(\underline{x})$
    - Really simple; no cross-talk between the basis functions
      - After application of T, a_k becomes $\lambda_k a_k$
    - But still need a separate eigen-analysis for each $T \in \mathbf{T}$
      - For one important class of operators, the eigenfunctions are the same

Basis Functions for Shift-Invariant Linear Operators in 1D



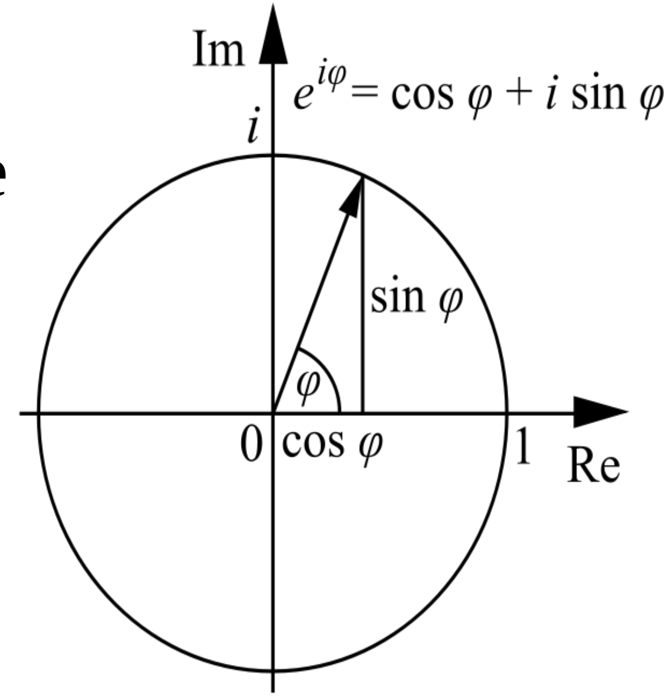
- $T(A\cos(2\pi\nu x) + B\sin(2\pi\nu x))$
  $= C\cos(2\pi\nu x) + D\sin(2\pi\nu x))$
  - Argument of T is a phase-shifted sinusoid of some amplitude and frequency ν
    - Because $A\cos(2\pi\nu x) + B\sin(2\pi\nu x) = |(A,B)|\cos(2\pi\nu x-\phi_{AB})$, where $\phi_{AB} = \tan^{-1}(B/A)$
    - Similarly, $C\cos(2\pi\nu x) + D\sin(2\pi\nu x)) = |(C,D)|\cos(2\pi\nu x-\phi_{CD})$
  - When a sinusoid of some frequency ν is input, the output is a sinusoid of the same frequency!
    - But with a modified amplitude and phase shifted

Basis Functions for Shift-Invariant Linear Operators



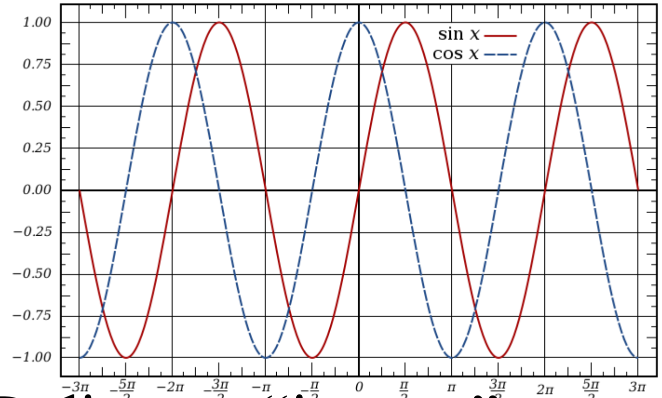
- When a sinusoid $E_{in} \cos(2\pi\nu x-\phi_{in})$, of some frequency ν is input, the output is a sinusoid of the same frequency: $E_{out} \cos(2\pi\nu x-\phi_{out})$,
  - Expressing the sinusoids in a complex form, that is, $\exp(-i2\pi\nu x) = \cos(2\pi\nu x) - i \sin(2\pi\nu x)$, these functions are eigenfunctions of all shift-invariant, linear operators
- Consider $\exp(-i2\pi\nu x)$ for $\nu = k/N$ with $k = 0, 1, 2, \ldots, N/2$
  - These basis functions span the space of 1D discrete "images" when $x = 0, 1, 2, \ldots, N-1$, with N even
  - These basis functions are called the discrete Fourier basis in 1D

Exp(iϕ) as unit circle and capturing sinusoids



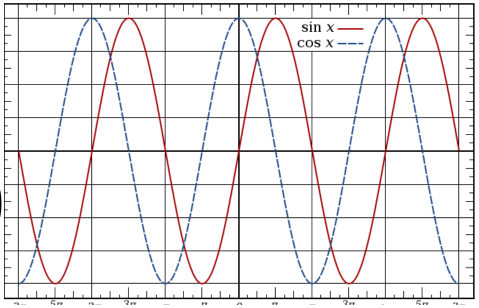
- exp(iϕ) is the point at angle ϕ on the unit circle in 2D complex space
- Trig shows that Re(exp(iϕ)) = cos(ϕ) and Im(exp(iϕ)) = sin(ϕ)
  - i.e., exp(iϕ) = cos(ϕ) + i sin(ϕ)
- Taylor series shows the same:
  - Taylor series for exp(iϕ) is $\Sigma_{k=0}^{\infty} (i\phi)^k / k! =$ $\Sigma_{k=0}^{\infty} (-1)^k \phi^{2k} / (2k)! +$ $i\,\Sigma_{k=0}^{\infty} (-1)^k \phi^{2k+1} / (2k+1)! =$ cos(ϕ) + i sin(ϕ)
- exp(-iϕ) = exp(i(-ϕ)) = cos(ϕ) - i sin(ϕ)
- cos(ϕ) = ½(exp(iϕ)+exp(-iϕ));
  sin(ϕ) = ½(exp(iϕ)-exp(-iϕ))/i

Discrete Basis Eigenfunctions for Shift-Invariant Linear Operators in 1D



- $\exp(-i2\pi\nu x)$ for $\nu = k/N$ with $k = 0, 1, 2, \ldots, N/2$
  - These basis functions span the space of 1D discrete "images" when $x = 0, 1, 2, \ldots, N-1$, with N even
- $I(x) = \Sigma_{k=-N/2+1}^{N/2} A_k \exp(-i2\pi(k/N)x)$ with $k = 0, 1, 2, \ldots, N/2$ ν_k
  - $= \Sigma_{k=1}^{N/2-1} [(A_k+A_{-k}) \cos(2\pi(k/N)x) - i (A_k-A_{-k})\sin(2\pi(k/N)x))] + A_0 \cos(-2\pi(0/N)x) + A_{N/2} \cos(-2\pi((N/2)/N)x)$
    - $(A_k + A_{-k})$ must be real, and $(A_k - A_{-k})$ must be imaginary, so $A_{-k} = A_k{}^*$, $k= 1, 2, \ldots, N/2-1$
    - A_0 and $A_{N/2}$ must be real, i.e., have phase 0
    - $A_0 \cos(-2\pi(0/N)x) + A_{N/2} \cos(-2\pi((N/2)/N)x) = A_0 + A_{N/2} \cos(\pi x)$

Discrete Basis Eigenfunctions for Shift-Invariant Linear Operators in 1D



- $I(x) = \Sigma_{k=1}^{N/2-1} [(A_k+A_{-k}) \cos(2\pi(k/N)x)$
  $-i\,(A_k-A_{-k}) \sin(2\pi(k/N)x))]+$
  $A_0 \cos(-2\pi(0/N)x) + A_{N/2} \cos(-2\pi((N/2)/N)x) =$
- $2\Sigma_{k=1}^{N/2-1} [\mathrm{Re}(A_k) \cos(2\pi(k/N)x) +$
  $\mathrm{Im}(A_k) \sin(2\pi(k/N)x))]+$
  $A_0\, 1 + A_{N/2} \cos(-\pi x) =$
- $2\Sigma_{k=1}^{N/2-1} [|A_k| \cos(2\pi(k/N)x- \phi(\nu_k))] +$
  $A_0\, 1 + A_{N/2} \cos(-\pi x)$,
  where $\phi(\nu_k)= \tan^{-1}(\mathrm{Im}(A_k) / \mathrm{Re}(A_k))$
  - For $k\neq 0$ or $N/2$, amplitude is $2\,|A_k|$, and phase is $\tan^{-1}(\mathrm{Im}(A_k) / \mathrm{Re}(A_k))$
  - For $k=0$ or $N/2$ phase is 0 or π

Basis Functions for Shift-Invariant Linear Operators in *M* dimensions

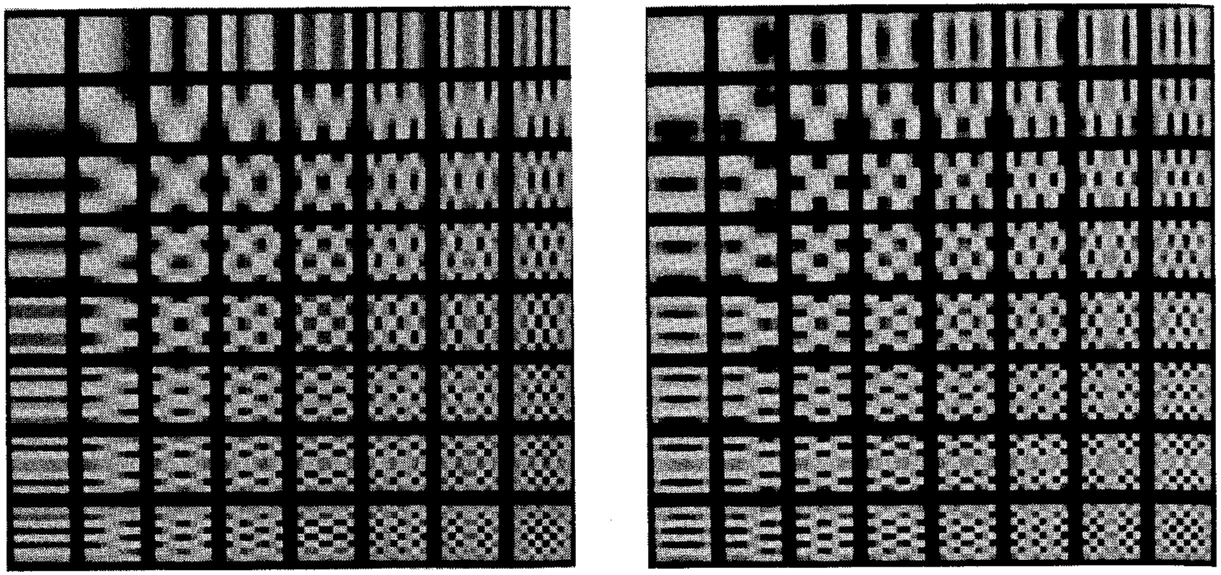
- When a sinusoid $E_{in} \exp(-i(2\pi \underline{\nu} \bullet \underline{x}))$,
  of some frequency $\underline{\nu} = (\nu_{x_1}, \nu_{x_2}, \ldots, \nu_{x_M})$ is input,
  - then the output is a sinusoid of the same frequency:
  - $E_{out} \exp(- i2\pi \underline{\nu} \bullet \underline{x})$, but E_{out} can be complex
  - This is the eigen-condition: $\lambda_\nu = E_{out} / E_{in}$
    - Note λ_ν may be complex; its magnitude changes the sinusoidal amplitude, and the ratio of its imaginary to real parts changes the sinusoidal "phase"
- $\exp(- i2\pi \underline{\nu} \bullet \underline{x}) = \Pi_{j=1}^{M} \exp(- i2\pi\, \nu_{x_j} \times x_j)$
  for $\nu_{x_j} = k_j/N_j$ with $k_j = 0, \pm 1, \pm 2, \ldots, \pm N_j/2-1, N_j/2$
  - These separable basis functions span the space of 1D discrete "images" when $x_j = 0, 1, 2, \ldots, N_j-1$, with N_j even
  - These basis functions are called the discrete Fourier basis in M dimensions

Discrete Basis Eigenfunctions for Shift-Invariant Linear Operators in *M*-D

- $\exp(-i2\pi\underline{\nu}\bullet\underline{x})$ for $\nu_{x_j} = k_j/N_j$ with each $k_j = 0, \pm 1, \pm 2, \ldots, \pm N_j/2-1, N_j/2$
- $I(x) = \sum_{j=1}^{M} \sum_{k_j=-N_j/2+1}^{N_j/2} A_{\underline{k}} \prod_{j=1}^{M} \exp\left(-i2\pi\left(k_j / N_j\right)x\right)$
  with $\underline{k} = (k_1, k_2, \ldots, k_M)$
- $(A_{\underline{k}} \exp(-i2\pi\underline{\nu}_{\underline{k}}\bullet\underline{x}) + A_{-\underline{k}} \exp(+i2\pi\underline{\nu}_{\underline{k}}\bullet\underline{x})) =$
  $(A_{\underline{k}}+A_{-\underline{k}})(\cos(2\pi\Sigma_i(k_i/N_i)x_i)$
  $-i\ (A_{\underline{k}}-A_{-\underline{k}})(\sin(2\pi\Sigma_i(k_i/N_i)x_i)$
  - $(A_{\underline{k}} + A_{-\underline{k}})$ must be real, and $(A_{\underline{k}} - A_{-\underline{k}})$ must be imaginary,
    so $A_{-\underline{k}} = A_{\underline{k}}^*$, $\underline{k}$: all indices ≥ 0 , not all either 0 or $N_j/2$
- For $\underline{k}$ with each component either 0 or $N_j/2$,
  - $A_{\underline{k}}$ must be real.

2D Basis Functions

- Separability: Products of $\Psi^{k_1}(x)$ and $\Psi^{k_2}(y)$
  - Each factor has its own frequency

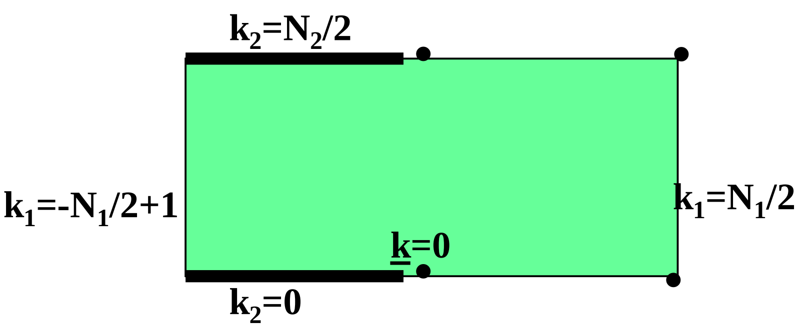


(a) Cosine (b) Sine

- Diagram shows N=8
- Different sub-panels show differing level of detail, by x and by y
- Indeed, the union of the sine and cosine basis forms an equivalent basis to the negative exponentials

Amplitude and Phase for Fourier Eigenimages in *M*-D

- $\underline{k}^{th}$ term for eigenvector decomposition:
  - $(A_{\underline{k}} \exp(-\, i2\pi\underline{\nu}_{\underline{k}} \bullet \underline{x}) + A_{-\underline{k}} \exp(+\, i2\pi\underline{\nu}_{\underline{k}} \bullet \underline{x})) =$
    $2\,\mathrm{Re}(A_{\underline{k}})(\cos(2\pi\Sigma_i(k_i/N_i)x_i)$
    $-i\, 2\,\mathrm{Im}(A_{\underline{k}})(\sin(2\pi\Sigma_i(k_i/N_i)x_i)$ when $\underline{k} \neq$ a tuple of 0s or $N_i/2$'s
  - $=2|A_{\underline{k}}|\cos(2\pi\Sigma_i(k_i/N_i)x_i - \phi(\underline{\nu}_{\underline{k}}))$,
    where $\phi(\underline{\nu}_{\underline{k}}) = \tan^{-1}(\mathrm{Im}(A_{\underline{k}}) / \mathrm{Re}(A_{\underline{k}}))$
  - In 2D there are complex conjugate pairs for every frequency pair but 4: (k_1=0 or N/2, k_2=0 or N/2). Thus, at those frequency pairs, in 2D
    - You get a magnitude and a phase for (k_1,k_2) and another magnitude and phase for $(-k_1,k_2)$ if $k_1 > 0$ or at $(k_1,-k_2)$ if $k_1 = 0$
  - For $\underline{k}$ with both components either 0 or $N_j/2$,
    - $A_{\underline{k}}$ must be real, so phase = 0 or π.
- To summarize, in fig. there is a mag and phase everywhere shaded but on the heavy lines and dots and a signed real on the dots.
  - (Each N/2 can equally well be –N/2)



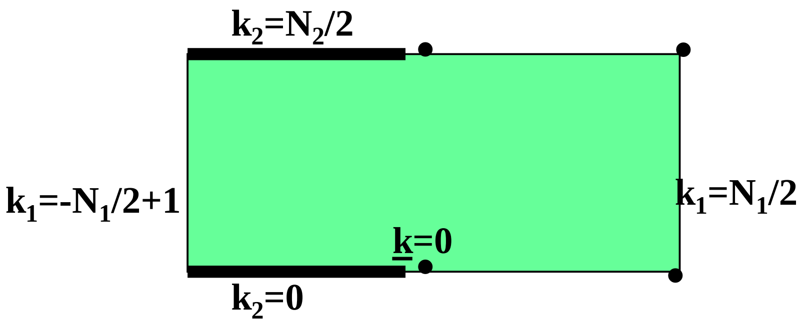
Reconstruction of an Image from Amplitudes and Phases for in N×N 2-D

- For x = 0 to N-1

  For y = 0 to N-1

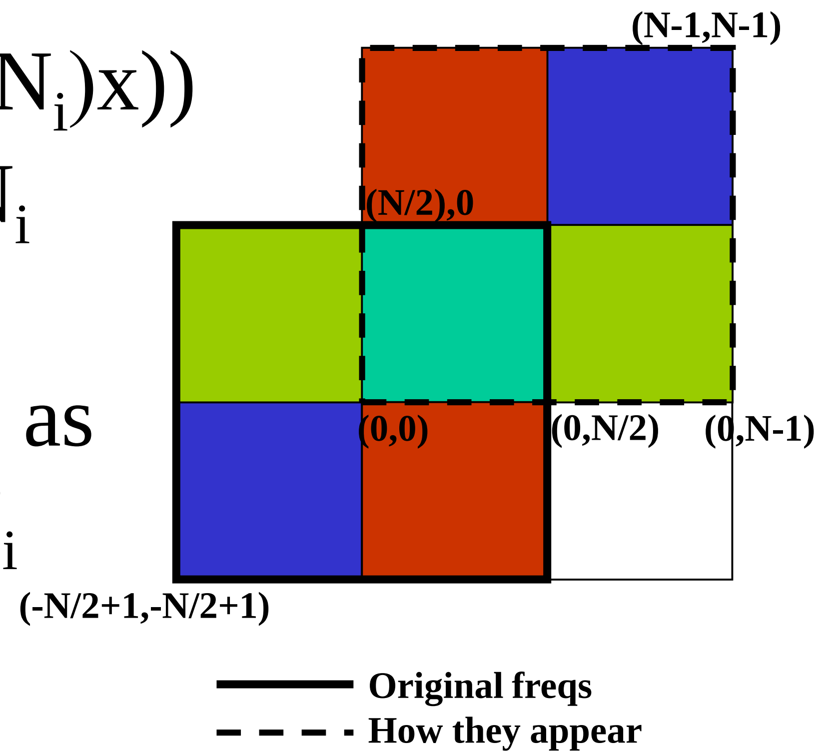
  $I_{reconstr}(x,y) = A(0,0) + A(N/2,0)\cos(\pi x) + A(0,N/2)\cos(\pi y) +$
  $A(N/2,N/2)\cos(\pi(x+y)) +$
  $\Sigma_{(k1,k2) \text{ in green region}}\ 2\ \text{Amplitude}(k1,k2)\cos(2\pi(k1\,x + k2\,y)/N - \text{Phase}(k1,k2))$

  – The complex A(i,j) items are normally stored in slot (k1,k2)



Relocation of frequencies 2 dimensions due to periodicity of the basis functions

- Each sinusoid $\exp(-i(2\pi\Sigma_i(k_i/N_i)x))$ is periodic in k_i with period N_i
- Thus the coefficient for frequency $(-k_i/N_i)$ is the same as that for frequency $(-k_i + N_i)/N_i$
  - That is, in frequency space all but the first quadrant can get transplanted, as in the figure
    - Dashed is how the FFT algorithm you could use displays the coefficient results it computes for an $N \times N$ input image I
    - Amplitudes and phases in the bottom half of the dashed square imply the rest of the solidly surrounded region



Basis Functions Orthogonality for Shift-Invariant Linear Operators

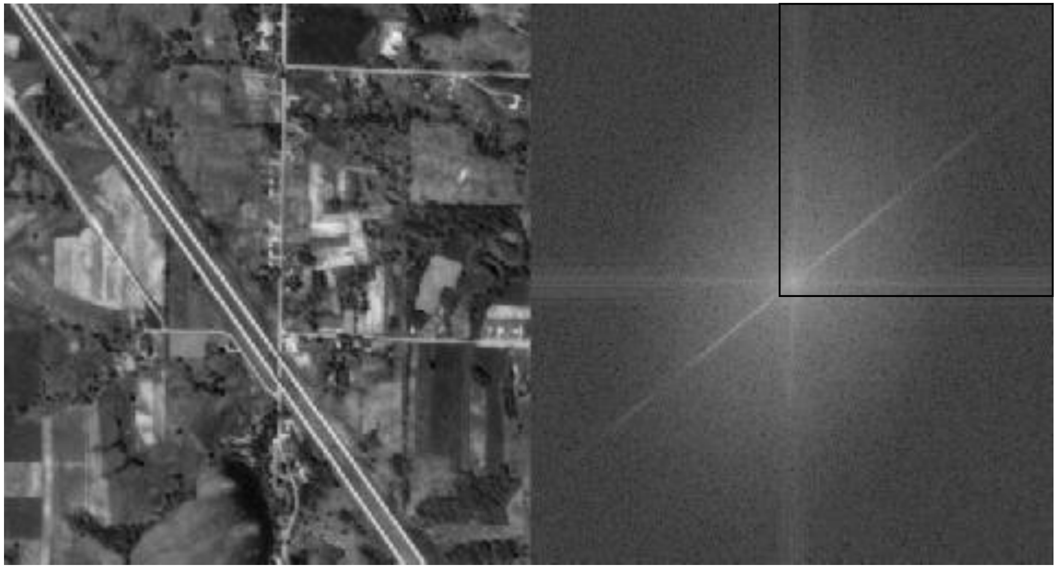
- Thm: The eigenimages $\Psi^j(\underline{x}_m)$ of shift-invariant linear operators are orthogonal
  - That is, in M-D $\sum_m \Psi^j(\underline{x}_m)\, \Psi^k(\underline{x}_m) = 0$ if $j \neq k$
- The $\Psi^j(\underline{x}_m)$ can be normalized in Euclidean length so that $\sum_m \Psi^j(\underline{x}_m)\, \Psi^j(\underline{x}_m) = 1$
- With these orthonormal $\Psi^j(\underline{x}_m)$, the representation $\underline{a}$ of $I(\underline{x}_m)$ is computed(!) by $a_j = \sum_m I(\underline{x}_m)\, \Psi^j(\underline{x}_m)$
  - With n= # of a_j and N being number of pixels or voxels, this O(nN) arithmetic operations for all n a_j is way faster (when n not << N) than the $O(n^3)$ needed when the basis is not orthogonal
  - Separability yields rows then columns appl'n: $O(n^{1/M}N)$

The Coefficients of the Discrete Fourier Basis Functions (Sinusoids)

- The basis functions are $\exp(-\, i2\pi\underline{\nu}\bullet\underline{x}) = \Pi_{j=1}^{M} \exp(-\, i2\pi\, \nu_{x_j}\, x_j)$; note separability
  - Or equivalently sines and cosines
  - Or equivalently cosines with amplitude and phase
- The coefficients of the representation of the discrete image $I(\underline{x})$ is called the "Discrete Fourier Transform" of I
  - We write this $\mathcal{F}(I)$
- Due to orthogonality and separability, $\mathcal{F}(I)(\underline{\nu}) = \mathcal{F}(\text{columns of } \mathcal{F}(\text{rows of } I))$
- $\mathcal{F}(\text{row of } I)(\nu_{\underline{x}_j}) = \text{row of } I \bullet [\text{constant} \times \exp(-\, i2\pi\, \nu_{x_j}\, \underline{x})]$
  - Same for column

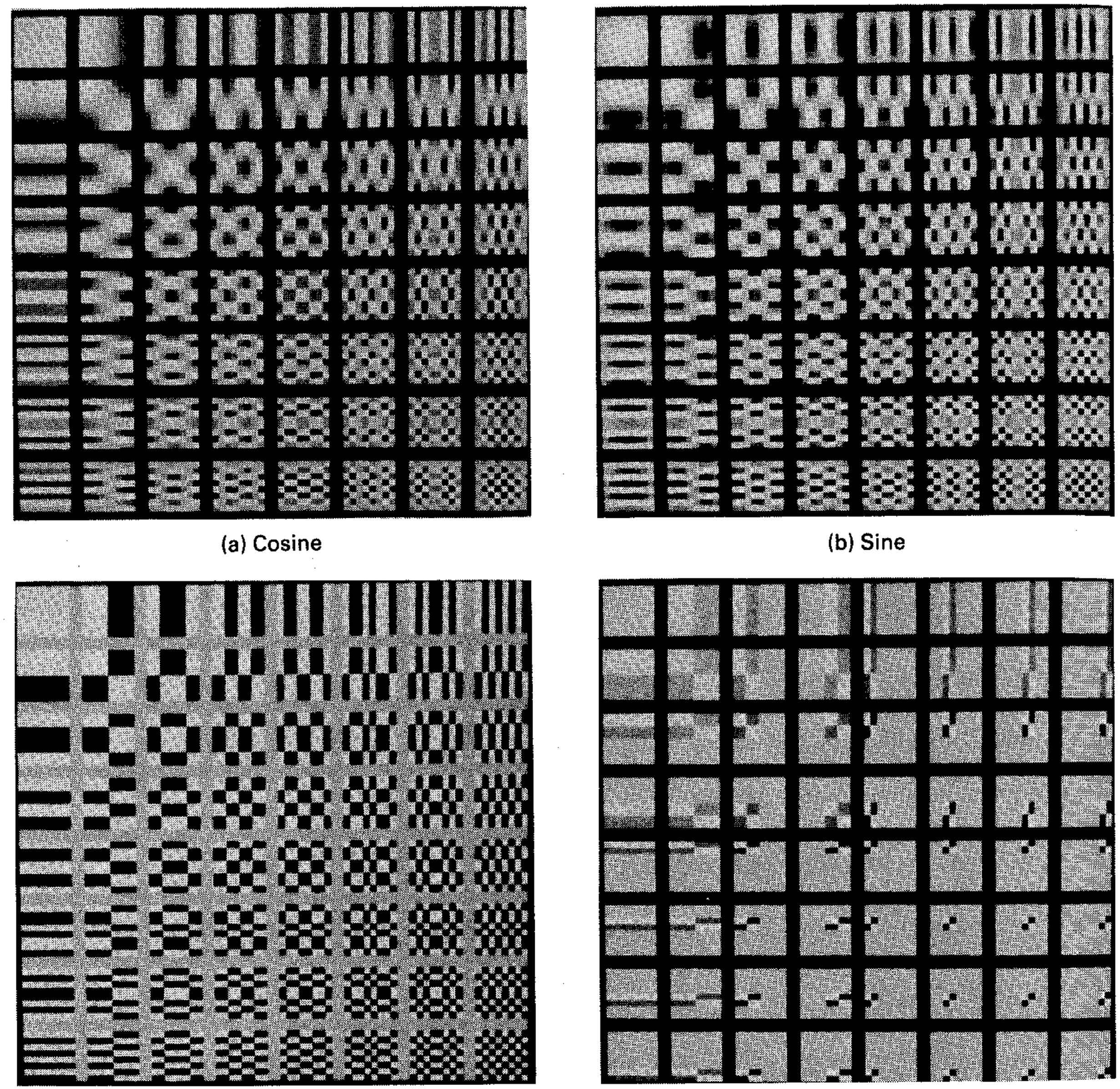
What you see when visualizing the DFT (coeffs of Fourier basis functions)

- Magnitude and phase images vs. frequencies: low to high ≥0
  - Not normally how the computation output appears, since it is done via complex exponentials at negative and positive frequencies:
    - cos(ϕ) = ½(exp(iϕ)+exp(-iϕ))
    - sin(ϕ) = ½(exp(iϕ)-exp(-iϕ))/i
- Edges and bars appear in the magnitude image as lines orthogonal to the edge or bar
  - when displayed centered at freq. (0,0)



- Major position changes are reflected in the phase image!

Alternative 2D Orthogonal Basis Functions
Different sub-panels show differing level of detail



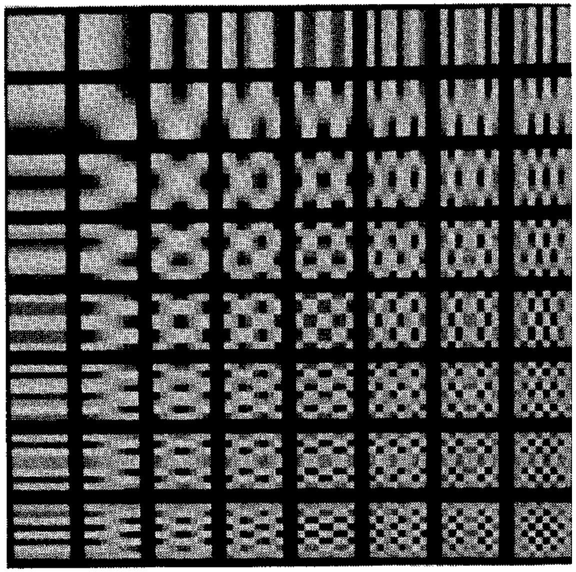
(a) Cosine (b) Sine (c) Hadamard (d) Haar

Properties of Alternative 2D Basis Functions

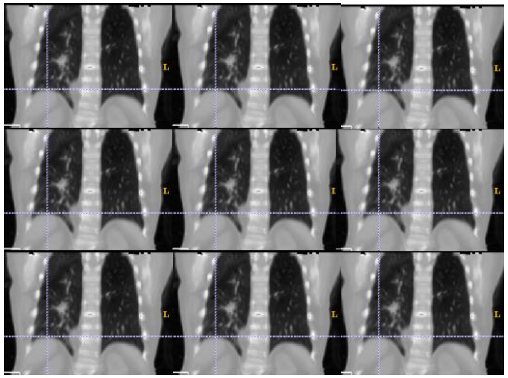
- They typically are chosen to be orthogonal
  - So image representation is pretty fast
- They are typically chosen to be separable
  - So image representation is even faster
- We will see yet another property of sinusoids that often yields a further speedup; only few of the alternatives (Hadamard) have this property
- Not eigenimages of shift-invariant linear operators
  - so crosstalk prevents simple understanding and computation of application of operators

Global Sinusoidal Basis Functions: the wraparound property

- The basis functions are cyclic: across image boundary, right to left and bottom to top
- Thus images represented via Fourier coefficients are cyclic
  - Shift-invariance is wrt to shifts that have this cyclic effect
  - The required value of n can be lowered if the image is adjusted to be smooth across these boundaries
- The coefficients $\underline{a} = \mathcal{F}(I)$ are cyclic in the frequencies $\nu_{x_j} = k_j/N_j$, with N_j entries in dim j

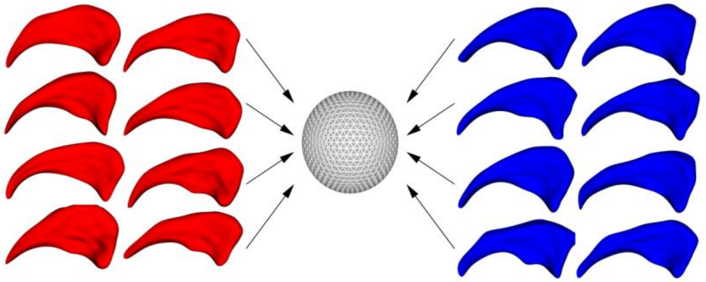


(a) Cosine

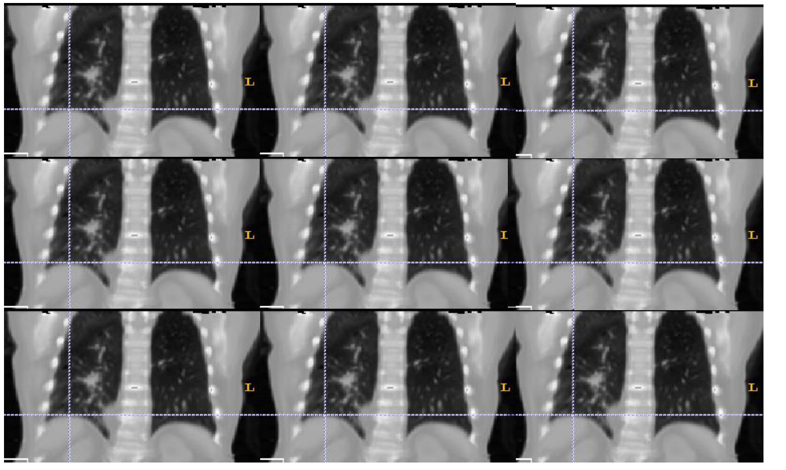


Global Sinusoidal Basis Functions: Details

- On $[0, 2\pi)^n$: Fourier basis functions
  - Three equivalent forms:
    - sine, cosine; negative exponential; amplitude, phases
  - For sine, cosine form, coefficients are real and frequencies ≥0
  - For negative exponential form, coefficients are complex and frequencies are corresponding positive and negative
  - For amplitude, phase form, "coefficients" are an amplitude >0 and an angle
- Different frequencies correspond to different levels of detail) in each parameter
- Generalizes for object to functions on the sphere $[0, \pi] \times [0,2\pi)$: spherical harmonics



Linear Shift-Invariant Operators



- Assume input image I and output image J are cyclic
- When input is $I_{shift}(\underline{x}) = I(\underline{x}-\underline{\Delta x})$, output $J(\underline{z}) = T(I_{shift}(\underline{x}))|_{\underline{x}=\underline{z}} = T(I(\underline{x}))|_{\underline{x}=\underline{z}-\underline{\Delta x}}$
- Theorem: for all such T, there exists a kernel (or a limit of kernels) $h(\underline{x})$ such that if $J = T \circ I$,
  - for continuous images:
    output $J(\underline{z}) = T(I(\underline{x}))|_{\underline{x}=\underline{z}} = \int_{\underline{x}} h(\underline{z}-\underline{x})\, I(\underline{x})\, d\underline{x}$
  - For discrete images $T(I(\underline{x}_q)) = \Sigma_{\underline{x}_m} h(\underline{x}_q-\underline{x}_m)\, I(\underline{x}_m)$
    - With indices wrapping around
- This operation is called "convolution" of h with I
  - Written $h{*}I$ (so you need to use $\times$ for multiplication)

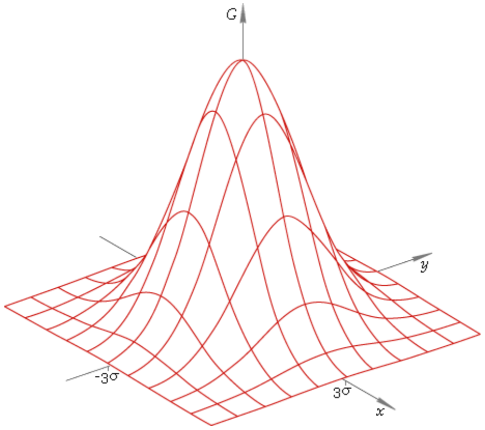
Order of Convolution Operands

- Provable that $I*h = h*I$ (commutativity)
  - Equivalence is proven by change of variables $\underline{y} = \underline{z}-\underline{x}$ in $\int_{\underline{x}} I(\underline{z}-\underline{x})\, h(\underline{x})\, d\underline{x}$
    - Note that the kernel h is itself an image
  - Alternatively provable via fact (see later) that convolution in space is equivalent to multiplication of FTs (just the eigen-relation), and multiplication is commutative
- Provable that $h_1*(h_2*I) = (h_1*h_2)*I = h_2*(h_1*I)$ (associativity)
  - Also due to associativity of multiplication, as applied to the 3 FTs (of I, of h_1, of h_2)

Interpretation and Direct Computation of Convolution

- Each input position $\underline{x}$ with value $I(\underline{x})$ is replaced by $I(\underline{x})$ times kernel $h(\underline{x})$, then superposition
  - $\int_{\underline{x}} h(\underline{z}-\underline{x})\, I(\underline{x})\, d\underline{x}$ [see online movies found by Google (convolution), both Wiki and Wolfram]
- Output at each $\underline{z}$ from weighting function per pixel: $w(\underline{z}) = h(-\underline{z})$
  - $\int_{\underline{x}} I(\underline{z}-\underline{x})\, h(\underline{x})\, d\underline{x}$, then change variables $\underline{y} = -\underline{x}$, leading to $\int_{\underline{y}} I(\underline{z}+\underline{y})\, h(-\underline{y})\, d\underline{y}$
    - If h is symmetric, weighting function w = kernel h

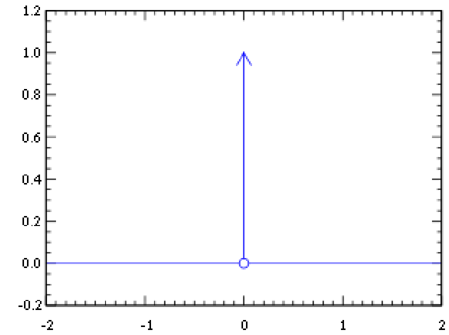
Examples of Continuous Convolution Kernels



- For blurring, $h(\underline{x})$ = isotropic Gaussian w/ RMS width σ:
  - $(1/(\sigma \sqrt{2\pi})^M) \exp(-½|\underline{x}|^2/\sigma^2) = \Pi_{i=1}^{M} (1/(\sigma \sqrt{2\pi})) \exp(-½|\underline{x}_i|^2/\sigma^2)$. It has unit volume
- For unweighted regional averaging, $h(\underline{x}) = \mathrm{rect}_{\Delta x}(\underline{x}) = (\Delta x)^{-M}$ inside the rectangle $[-\Delta x/2, \Delta x/2]$ in each of the M dimensions; and 0 elswhere. That h has unit volume
- For identity operator, it is a limit (next slide) $\delta(x)$
- For derivative taking, it is a limit (later slides)
- For imaging, h = point spread function ("psf") = image of a point (a few slides later)

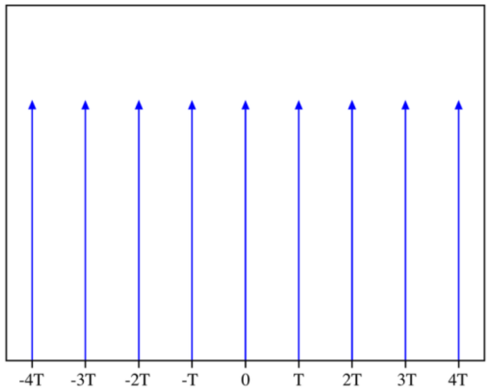
Convolution Kernels that are Limits

- Identity kernel $\delta(\underline{x})$, the "Dirac delta function"
  - Limit as width goes to zero of any positive function with width as a parameter and integral over $[-\infty, \infty] = 1$
    - Examples: Gaussian($\underline{x}$), $\text{rect}_{\Delta x}(x)$ [see movie found via Google(Dirac delta function)]
  - It is an infinitely high spike that is zero except at $\underline{x}=0$
  - $\int_\Omega \delta(\underline{x}_0-\underline{x})\, f(\underline{x})\, d\underline{x} = f(\underline{x}_0)$
  - Discrete counterpart is an image centered at (0,0) or (0,0,0) that is 1/voxel (pixel) volume (area) (typically 1) at the center voxel and zero in every other pixel or voxel



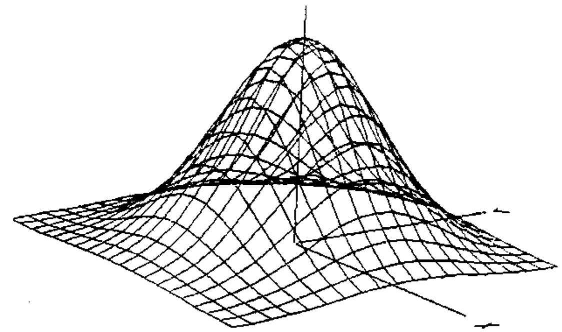
Convolution Kernels that are Limits

- The $\mathrm{Comb}_{\Delta x}(x)$ function has a δ function centered at every integer multiple of Δx; $= \Sigma_{j=-\infty}^{\infty} \delta(x-j\Delta x)$
- $\Delta x\ \mathrm{Comb}_{\Delta x}(x) \times I$ = sampled version of I

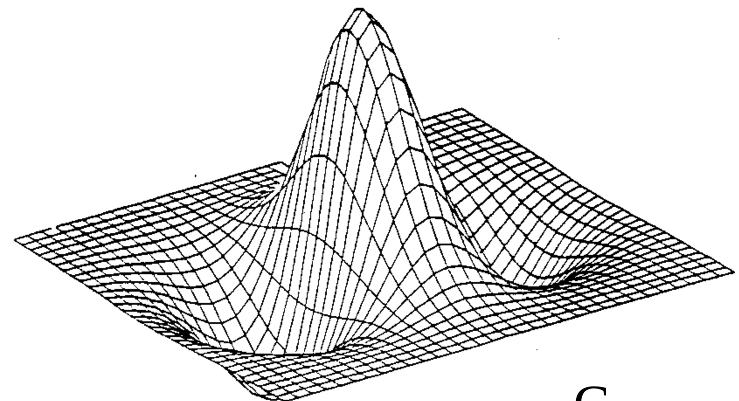


Convolution Kernels that are Limits

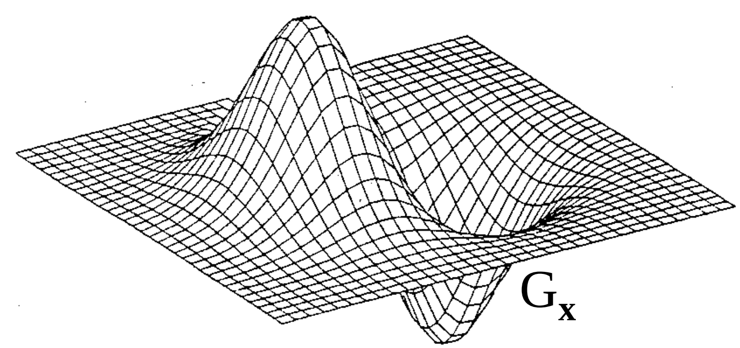
- Derivatives D of any order in any direction(s)
  - They are linear and shift-invariant:
    - Consider $D(\alpha I_1+\beta I_2)$
    - Consider $DI(\underline{x}-\underline{\Delta x})$
  - Due to linearity and shift-invariance, their operation commutes and is associative with other linear, shift-invariant operators
  - They need to be computationally applied using a unit-volume kernel h, e.g., Gaussian: $[Dh]*I(\underline{x}) = D(h*I(\underline{x})) = h*DI(\underline{x})$
  - Math derivative is limit of Dh as width goes to zero of that derivative of the unit-volume kernels that led to $\delta(\underline{x})$



Gaussian: G



$-G_{xx}$



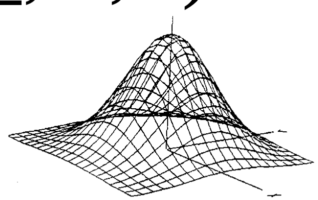
G_x

Convolution Kernels for Imaging

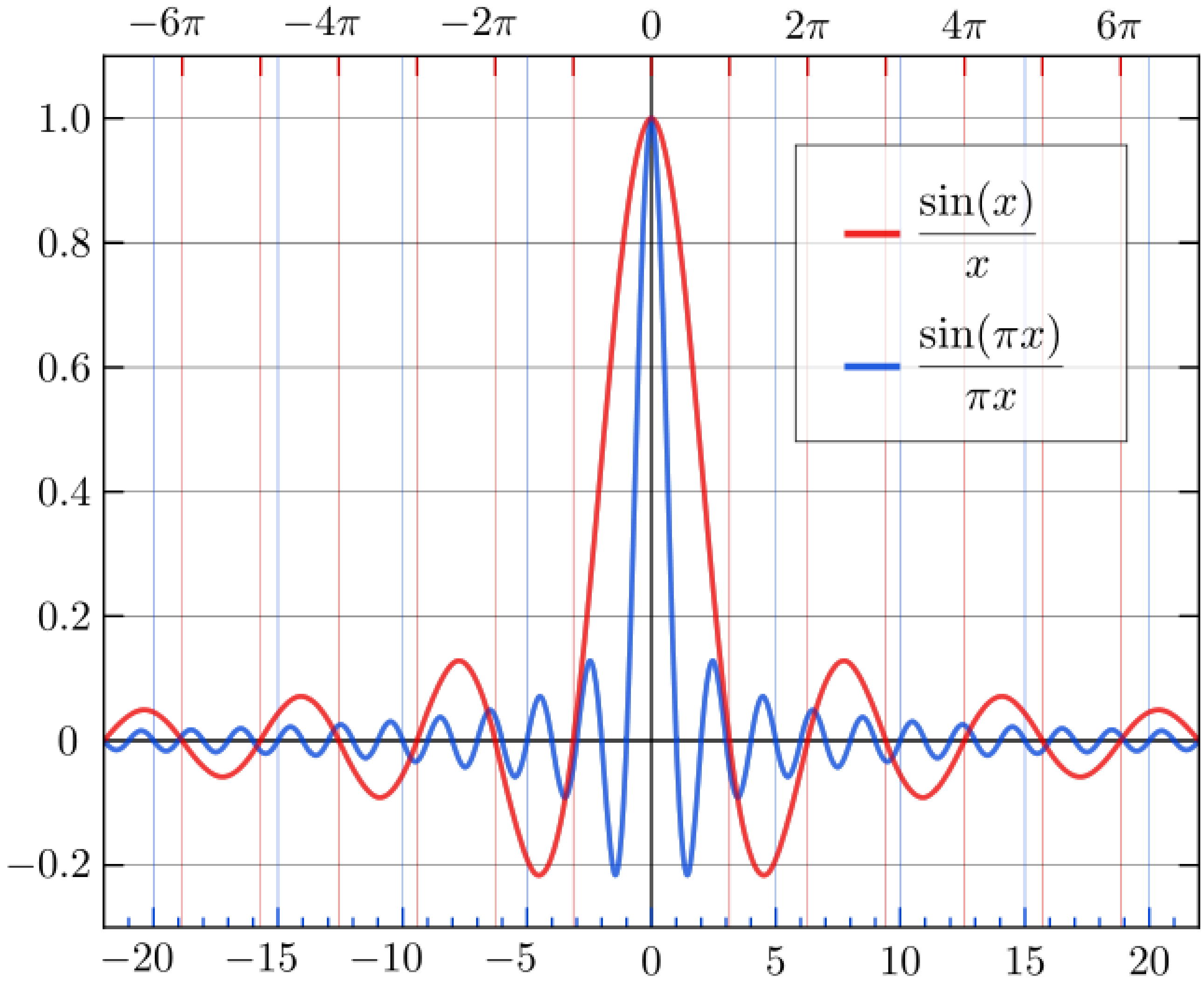
- $h(\underline{x})$ = point spread function ("psf") = image of a point $\delta(\underline{x})$: $\int_{\underline{x}} I(\underline{z}-\underline{x})\, h(\underline{x})\, d\underline{x} = \int_{\underline{x}} \delta(\underline{z}-\underline{x})\, h(\underline{x})\, d\underline{x} = h(\underline{z})$
  - RMS width of h is amount of blurring
  - Since $I(\underline{z}) = \int_{\underline{x}} I(\underline{x})\, \delta(\underline{z}-\underline{x})\, d\underline{x}$, a linear combination of Dirac δ functions, $T(I(\underline{z})) = \int_{\underline{x}} I(\underline{x})\, T(\delta(\underline{z}-\underline{x}))\, d\underline{x} = \int_{\underline{x}} I(\underline{x})\, h(\underline{z}-\underline{x})\, d\underline{x}$, the convolution of I with the psf.
  - Alternative: line spread function = image of a line $I(\underline{x}) = \delta(x)$ [the function is constant in all variables but x]

Common Fourier Transforms

Kernel	Fourier transform
• $\delta(x)$, $\delta(x-x_0)$	$1,\ e^{i2\pi\nu x_0}$
• Sampling function $\Delta x\ \mathrm{Comb}_{\Delta x}(x)$	$\mathrm{Comb}_{1/\Delta x}(\nu)$
• Averaging: $\mathrm{Rect}_{\Delta x}(x)$	$\sin(\pi\nu\Delta x)/(\pi\nu\Delta x)$
• Gaussian($\underline{x}$; $0,\sigma$)	$\exp(-\frac{1}{2}[2\pi\sigma]^2\lvert\underline{\nu}\rvert^2) = 2\pi\sigma^2$ Gaussian($\underline{\nu}$; $0,1/[2\pi\sigma]$)
• $\partial^n/\partial x^n$	$(2\pi i\nu_x)^n$
• Laplacian: $\Sigma_{i=1}^{M}\ \partial^2/\partial x_i^{\,2}$	$-(2\pi\lvert\underline{\nu}\rvert)^2$

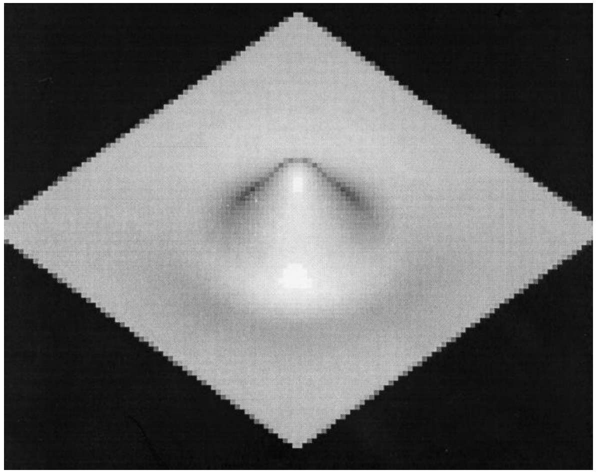


The sinc function: sin(αx)/(αx)



Laplacian of Gaussian kernel
Wider or larger aperture gives different scales

- In M-D: $\nabla^2 G = \Sigma_{i=1}^{M} \partial^2 G/\partial x_i^2$
- FT of $\nabla^2 = -(2\pi|\underline{\nu}|)^2$
  - Circularly symmetric in both freq and space, like isotr'c. G
    - For all kernels, circularly symmetric in space $\Leftrightarrow$ circularly symmetric in freq
    - Thus independent of coordinate dirs.
- As kernel, selects out circular blobs
  - Response high at center of blob
- When applied as kernel, zero crossings select edges

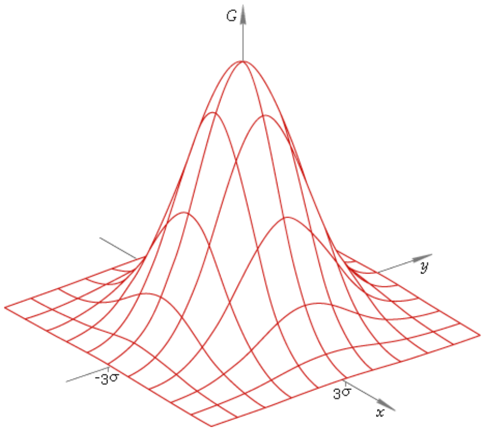


-Laplacian of Gaussian

The Convolution Theorem

- Think of convolution under Fourier representation
  - Thm: If $\mathcal{F}$(I) is Fourier rep of I and $\mathcal{H}$ is Fourier rep of kernel h, then $\mathcal{F}$(I) × $\mathcal{H}$ is Fourier rep of I*h
  - This is how you prove commutativity and associativity of shift-invariant linear operators
- What this buys you
  - Yields fast convolution
  - Yields good understanding of convolution and thus the ability to design convolution kernels
    - Think of *filters* $\mathcal{H}(\underline{v})$ vs $\underline{v}$ rather than kernels $h(\underline{x})$ vs $\underline{x}$

Examples of Filters



- Gaussian($\underline{v}$): low pass, blurring
  - Separable and isotropic
- Rect($\underline{v}$): low pass, blurring
  - Not isotropic if applied separably

- 1/ $\mathcal{H}(\underline{v})$: high pass, sharpening
- 1/ $\mathcal{H}(\underline{v})$ up to some $|\underline{v}|$, then falloff: sharpening, then smoothing of small detail

Fast Convolution and Fast Computation of Fourier Image Representation

- Fast convolution results from basis functions that are 1) eigenfunctions of convolution; 2) orthonormal; 3) separable, and
  - Multiplicative decomposition of levels of detail: $\psi^{j}(\mathbf{x}) = \psi^{j_1}(\mathbf{x})\ \psi^{j_2}(\mathbf{x})$
- This is true of the sinusoids
- This allows a divide-and-conquer algorithm FFT for computing the Fourier image representation or its inverse that is $\Theta(N \log N)$

Speedy Convolution via the Frequency Domain

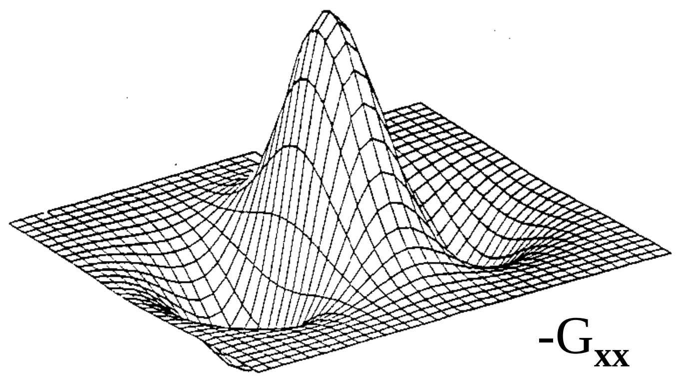
- Convolution of I with impulse response function h is filtering (when convolution wraps around)
- Method
  - Compute coefficients of sinusoidal decompositions $\mathcal{F}$(I), $\mathcal{F}$(h) in exponential form ("Fourier transform")
      - By FFT; speed O(N log N); N= # of pixels /voxels in image
  - Compute product, level of detail by level of detail, $\mathcal{F}(I) \times \mathcal{F}(h)$; speed Θ(N) vs. Θ(Nm) for direct convolution
    - Where m is number of pixels or voxels in the kernel h
  - Reconstruct result from sinusoidal basis functions ("inverse Fourier transform)"
      - By FFT; speed Θ(N log N)

The Inverse DFT

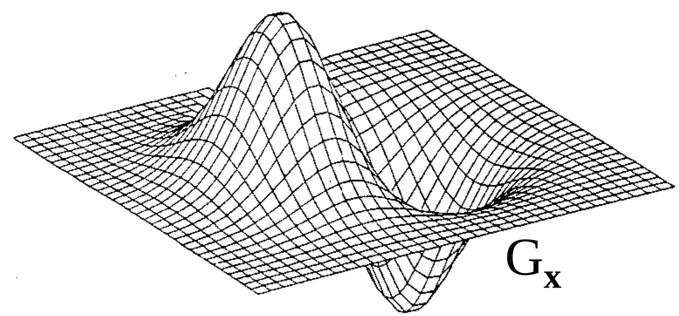
- Reconstructing J($\underline{x}$) from the coefficients of the Fourier basis functions
- It turns out that it is exactly the same as the DFT, except with exp(iθ) replacing exp(-iθ)
- Thus FFT^{-1} is done by FFT algorithm with one sign changed
  - So speed of FFT^{-1} is Θ(N log N)
- Convolution theorem works re inverse
  - Not only $\mathcal{F}^{-1}[\mathcal{F}(I)\ \mathcal{F}(h)] = I*h$
    but also $\mathcal{F}^{-1}[\mathcal{F}(I)*\mathcal{F}(h)] = I\times h$

Convolution Kernels that are Limits

- Fourier transform of the derivative operators
  - If $DI = \partial^{n_1}/\partial x_1^{\,n_1}\ \partial^{n_2}/\partial x^{n_2} \ldots \partial^{n_M}/\partial x^{n_M}\, I$,
    $\mathcal{F}\,(DI(\underline{x})) = \Pi_j (2\pi i \nu_{x_j})^{n_j}\, \mathcal{F}(I)$
  - $\mathcal{F}([D{*}h]\ I(\underline{x})) = [\Pi_j (2\pi i \nu_{x_j})^{n_j}\, \mathcal{H}(\underline{\nu})]\, \mathcal{F}(I)$
  - As $h \rightarrow \delta(\underline{x})$,
    $\mathcal{H}(\underline{\nu}) \rightarrow 1$, e.g., for Gaussian
  - Discrete counterpart used is typically sampled deriv. of Gaussian



$-G_{xx}$



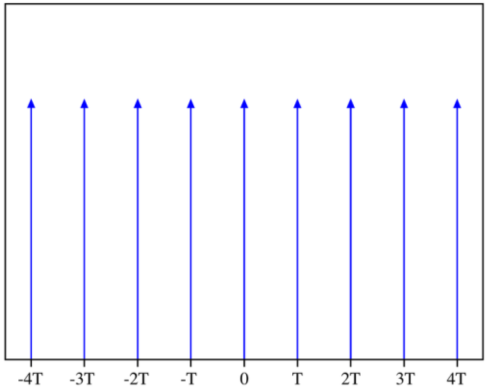
G_x

Sampling and integration (digital images)

- Model
  - Within-pixel integration at all points
    - Has its own kernel, typically rect over pixel
  - Then sampling
- Sampling = multiplication by pixel area × brush function
  - Brush function is sum of impulses (δ functions) at pixel centers

Sampling and integration (digital images)

- Model
  - Within-pixel integration at all points
    - Has its own kernel, typically rect over pixel
  - Then sampling



- Sampling = multiplication by pixel area × brush function
  - Brush function is sum of impulses

- $\Delta x^M \, \mathrm{comb}_{\Delta x}(\underline{x}) =$
  $\Delta x^M \, \Sigma_{j,k=-\infty}^{\infty} \, \delta(x-j\Delta x, y-k\Delta y)$; $M=2$

Creating digital images

- $I_{sampled} =$
  $[I(\underline{x}) * \text{rect}_{\Delta x}(\underline{x})] \times \Delta x^M \text{comb}_{\Delta x}(\underline{x})$
  - Within-pixel integration at all points
    - For unweighted averaging within pixel: rect
    - For weighted averaging, extending beyond pixel: Gaussian
  - Then sampling (then cutoff to finite image; see later)
- Consider in frequency domain (with rect)
  - $\mathcal{F}[I_{sampled}] = \mathcal{F}([I(\underline{x}) * \text{rect}_{\Delta x}(\underline{x})] \times \Delta x^M \text{comb}_{\Delta x}(\underline{x})) =$
    $\mathcal{F}(I(\underline{x}) * \text{rect}_{\Delta x}(\underline{x})) * \mathcal{F}(\Delta x^M \text{comb}_{\Delta x}(\underline{x})) =$
    $[\mathcal{F}(I(\underline{x})) \times \Pi_{i=1}^{M} [\sin(\pi \nu_i \Delta x)/(\pi \nu_i \Delta x)]] * \text{comb}_{1/\Delta x}(\underline{\nu})$
    - Define $G(\underline{\nu}) = [\mathcal{F}(I(\underline{x})) \times \Pi_{i=1}^{M} [\sin(\pi \nu_i \Delta x)/(\pi \nu_i \Delta x)]]$

Sampling = Aliasing [Shannon]

In diag.: “h” = Δx

$G(\nu) = \mathcal{F}(I(\underline{x})) \times [\sin(\pi\nu\Delta x)/(\pi\nu\Delta x)]$

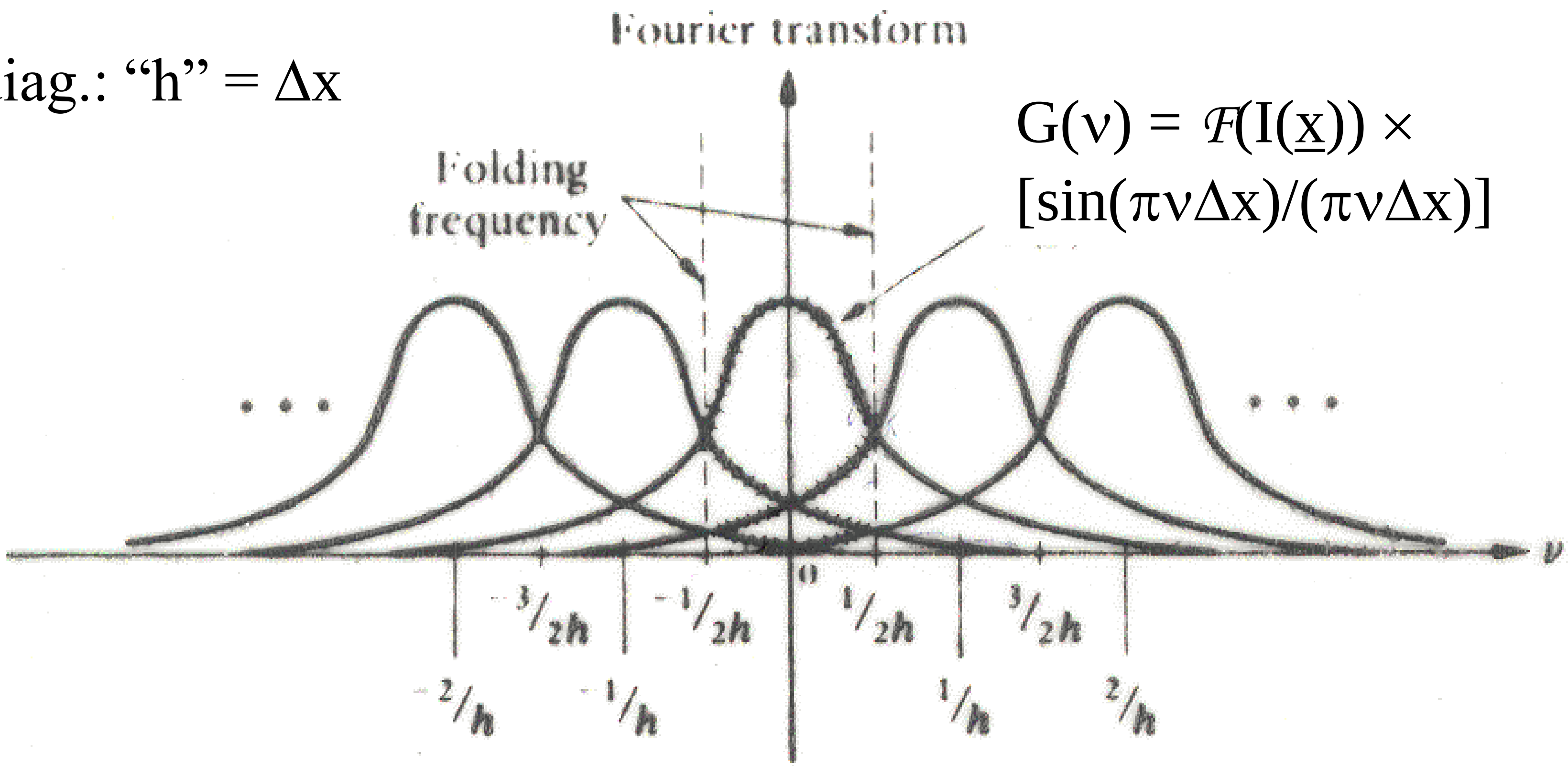


FIG. 4-22 Fourier transform of a sampled function

Frequencies $\nu \pm j(1/\Delta x)$ for $j \neq 0$ aliases as freq. ν

The effect of aliasing: folding

- Folding frequency = Nyquist frequency $\nu_N = \pm\frac{1}{2}(1/\Delta x)$
  - There is folding at $\nu = \nu_N$
  - With each fold, there is complex conjugation of $G(\underline{\nu})$
    - $\mathcal{F}[I_{sampled}](\underline{\nu})$ in $[0, +\frac{1}{2}(1/\Delta x)] = \Sigma_{k=-\infty}^{\infty} G^{*k}(\underline{\nu} - k(1/\Delta x))$
  - where $G(\underline{\nu}) = \mathcal{F}(I(\underline{x}))(\underline{\nu}) \times \Pi_{i=1}^{M} [\sin(\pi\nu_i\Delta x)/(\pi\nu_i\Delta x)]$
  - and $*^k$ means k applications of complex conjugation

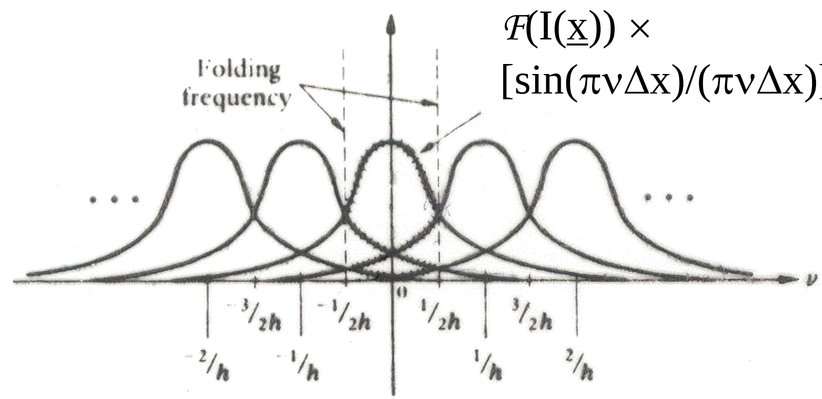


FIG. 4-22 Fourier transform of a sampled function

Non-aliasing under band limitation

- Folding frequency = Nyquist frequency ν_N = $\pm\frac{1}{2}(1/\Delta x)$
- If G is band-limited to frequency ν_N,
  $\mathcal{F}(I(\underline{x}))(\underline{\nu}) \times$
  $\Pi_{i=1}^{M} [\sin(\pi\nu_i\Delta x)/(\pi\nu_i\Delta x)]$
  = G in interval
  $[-\frac{1}{2}(1/\Delta x), +\frac{1}{2}(1/\Delta x)]$,
  0 elsewhere
  $= G(\underline{\nu}) \times \mathrm{rect}_{1/(2\Delta x)}(\underline{\nu})$

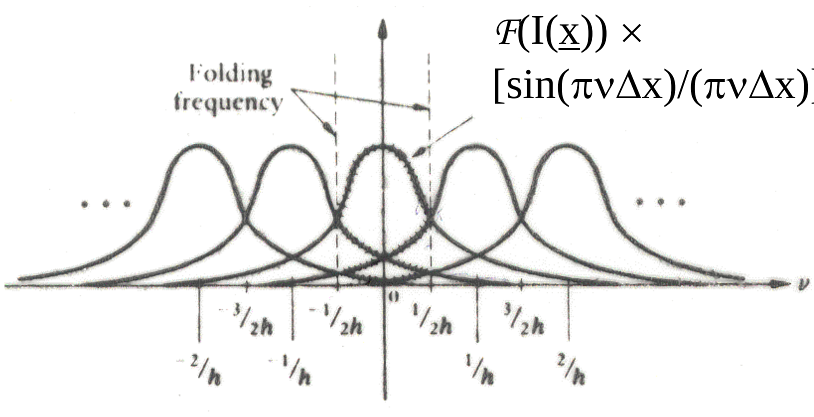


FIG. 4-22 Fourier transform of a sampled function

Recovering continuous image from the discrete image, when no aliasing

- Divide by $\Pi_{i=1}^{M}$ [$\sin(\pi\nu_i\Delta x)/(\pi\nu_i\Delta x)$]] in [$-½(1/\Delta x), +½(1/\Delta x)$]
  – Or by Gaussian if that was used in forming pixels
- Then multiply by $\mathrm{rect}_{1/(2\Delta x)}(\underline{\nu})$ to yield $\mathcal{F}(I(\underline{x}))(\underline{\nu})$
- Apply $\mathcal{F}^{-1}$
  – Multiplication by rect in freq. is equivalent to convolution with $\Pi_k \mathrm{sinc}(\pi\nu_k\Delta x)$ in space

Preventing aliasing by adequate sampling

- Band limit to $\nu = \pm \nu_N = \pm ½(1/\Delta x)$
  - After rect blurring (sinc multiplication) is accounted for
    - Treat as $1/(\pi \nu_N \Delta x) = 2/\pi$ at $\nu = \nu_N$
- But most spectra get small but not zero for large frequencies
  - If imaging convolution kernel has std dev σ (so its FT has std dev (in ν) $1/(2\pi\sigma)$)
  - And your objective is $\mathscr{F}(I)(\nu_N) / \mathscr{F}(I)(\nu=0) = \varepsilon$ in order to get under ε fractional pollution from the first round of folding
  - And you assume scene FT (spectrum) is flat in ampl.
  - Then $\Delta x = \pi\sigma / \sqrt{[2 \ln(2/\pi\varepsilon)]}$
- Sampling can coarsen as you increase scale

Anti-aliasing

- For fixed Δx: band limit by Gaussian blurring so that $\Delta x = \pi\sigma / \sqrt{[2 \log(2/\pi\varepsilon)]}$, i.e., by convolution with Gaussian that when combined with the other sources of blurring produces $\sigma = (\Delta x/\pi) \sqrt{[2 \log(2/\pi\varepsilon)]}$
  – Do it before sampling!
- Typically it is too expensive to blur with Gaussian
  – Blur with cheaper function with same rms value as the aforementioned Gaussian, e.g., rect
    - Exercise: what is RMS width of rect as a function of Δx?

Good things about the Fourier representation

- Eigenfunctions of all shift-invariant operators
- Separable, rotational invariance, multiplicative decomposition, orthonormality
  - Thus fast calculation of coefficients of basis functions
  - Thus fast calculation of convolution
- Need relatively few eigenfunctions (and thus coefficients) to get a rather accurate approximation to an image, if the image is pretty smooth
  - Thus, most common compression methods store the coefficients, not the image pixel values

Linear Non-Shift-Invariant Operators

- For shift-invariant (linear) operators, the weighting function (equivalently the kernel) is not a function of position, only of position relative to the application pixel
  - $h(\underline{x}-\underline{y})$
- For non-shift-invariant operators, the weighting function varies with position of application
  - $h(\underline{y}, \underline{x}-\underline{y})$

A bad thing about the Fourier representation

- Need more basis functions for any level of accuracy than is necessary
- What basis requires the fewest basis functions on average?
  - – Ones specialized to your particular family of images
    - See next few slides

The Big! Difficulty with the Fourier Representation

- It expresses locality very badly
  - For example, phase info is the only reflection of position, but only global shifts show up clearly
  - Somehow you need Fourier analysis through a Gaussian window centered at the location of interest
    - Multiplication by a Gaussian in space, so convolution with a Gaussian in DFT (“frequency space”)
    - But you need this for many window locations
- For better locality than with Fourier basis, see the “scale and locality” section, coming up

Orthogonal Decomposition with Basis Images Specialized to Your Data

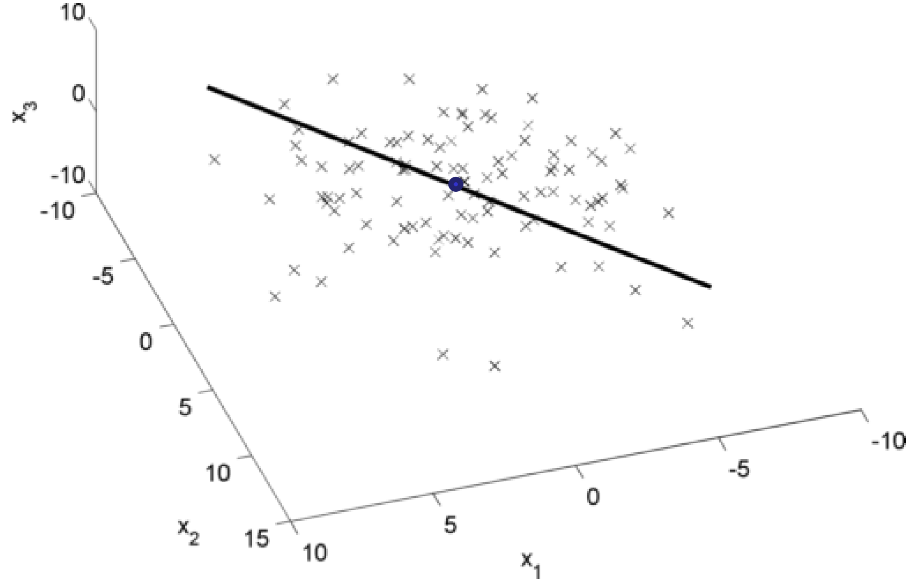
- Objective: a set of N basis vectors $\mathbf{u}^j$ for an N-tuple $\mathbf{a}$ (for us, the discrete image $\mathbf{I}$ in row major order) in order, $j = 1, 2, \ldots, N$, such that for each $J = 1, 2, \ldots, N$, decomposition into the first J basis vectors produces a vector decomposition of $\mathbf{a}$ in the basis functions that on the average, over the *training population* of $\mathbf{A}$, is closest to $\mathbf{A}$
  - Depends on the population
  - Is computed from a training sample of the population: $\underline{\mathbf{I}}^k$, $k = 1, 2, \ldots, n$

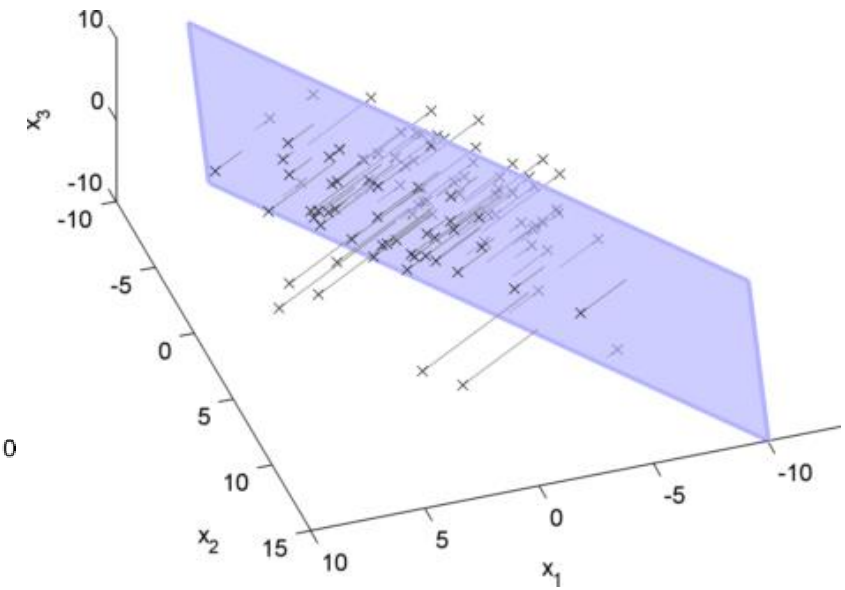
Orthogonal Decomposition Specialized to Your Data: Strategy

- Data list A with N features (pixels) and n instances
  - $n \times N$ array A; each of the n rows $\mathbf{I}^k$ is a training sample
  - Typically $n << N$
- Consider an orthonormal row-major-image basis $\{\mathbf{u}^j \mid j=1,2,\ldots,N\}$, to be discovered
  - Let $\mathbf{b}^k$ list the N coefficients of the $\mathbf{u}^j$ in $\mathbf{I}^k$
  - If the coefficient of $\mathbf{u}^1$ in $\mathbf{b}^k$ (i.e., $b^k{}_1 = \mathbf{I}^k \bullet \mathbf{u}^1$) provides a maximally accurate fit, on the average, to the training samples
    - i.e., $(1/N)\Sigma_k|\mathbf{I}^k - (\mathbf{I}^k \bullet \mathbf{u}^1)\, \mathbf{u}^1|^2$ is minimum
  - then we will prove that on the average, the square of the part of $\mathbf{b}^k$ that is the coefficient of $\mathbf{u}^1$ (i.e., $(b^k{}_1)^2$), summed over the training images, is the largest proportion of $\Sigma_k|\mathbf{b}^k|^2$ (i.e., of $\Sigma_{k,j}(b^k{}_j)^2$)
    - That is, $\Sigma_k(b^k{}_1)^2 / \Sigma_{k,j}(b^k{}_j)^2 = \Sigma_k(\mathbf{I}^k \bullet \mathbf{u}^1)^2 / \Sigma_{k,j}(\mathbf{I}^k \bullet \mathbf{u}^j)^2$ is maximum
- To see this, we need the general Parseval's theorem

Visualizing training images, $\mathbf{u}^1$, and $\mathbf{u}^2$

- With N pixels a row-major-image $\mathbf{I}^k$ is a point in N-space
- A (unit) basis image is a unit vector $\mathbf{u}^j$ for each j
- There is a line in N-space along each $\mathbf{u}^j$
- $\mathbf{u}^1$ and $\mathbf{u}^2$ span a plane; the first J $\mathbf{u}^j$ span a flat J-dimensional hyperplane
- You want best fitting (in distance^2) to the $\mathbf{I}^k$, of the line, of the plane, …
  - So $\mathbf{u}^1$ is along best fitting line, $\mathbf{u}^2$ $\perp$ and with $\mathbf{u}^1$ defines best fitting plane, …





Parseval's Theorem

- Consider a full orthonormal basis $\{\mathbf{u}^i, i = 1, 2, \ldots, N\}$ for N-entry vectors
  - The coefficients b_i for $\mathbf{I}$ in this basis are $b_i = \mathbf{I} \bullet \mathbf{u}^i$
- Consider another full orthonormal basis $\{\mathbf{v}^i, i = 1, 2, \ldots, N\}$ for N-entry vectors
  - The coefficients c_i for any $\mathbf{I}$ in this basis are $c_i = \mathbf{I} \bullet \mathbf{v}^i$
- Then $\Sigma_{i=1}^{N} (b_i)^2 = \Sigma_{i=1}^{N} (c_i)^2$
  - Proved by looking at $\mathbf{I} \bullet \mathbf{I} = (\Sigma_{i=1}^{N} b_i \mathbf{u}^i) \bullet (\Sigma_{j=1}^{N} b_j \mathbf{u}^j) = (\Sigma_{i=1}^{N} c_i \mathbf{v}^i) \bullet (\Sigma_{j=1}^{N} c_j \mathbf{v}^j)$
  - One possible orthonormal set is the discrete δ-function images $\boldsymbol{\delta}^i$
    - $\boldsymbol{\delta}^i$ is zero except that in pixel i it is 1
    - the coefficient of $\boldsymbol{\delta}^i$ is $\mathbf{I}$'s entry in the i^{th} pixel
    - Thus $\Sigma_{i=1}^{N} (b_i)^2 = \Sigma_{j,k=1}^{N} (\mathbf{I}(x_j, y_k))^2$ for any orthonormal basis

Orthogonal Decomposition: Mathematical setup

- Consider an orthonormal basis $\{\mathbf{u}^i\}$, to be discovered from the training images $\mathbf{I}^k$
  - We think of a set of basis functions (images) $\{\mathbf{u}^i, i=1,2,...,N\}$. Let the matrix U have columns $\mathbf{u}^i$, i=1,2,...,N. U is square.
  - The coefficient of $\mathbf{u}^i$ for expressing $\mathbf{I}^k$ is $b_i^k = \mathbf{I}^k \bullet \mathbf{u}^i = \mathbf{I}^{kT}\, \mathbf{u}^i$, so $\mathbf{b}^k = (\mathbf{I}^{kT}\, U)^T = U^T\mathbf{I}^k$
    - Each of those basis vectors $\mathbf{u}^i$, as well as the vectors $\mathbf{I}^k$, is an image tuple produced by expressing the image array in row-major order
  - Write the set of coefficients $\{b_i^k \mid i=1,2,...,N\}$ for expressing $\mathbf{I}^k$ as the tuple (vector) $\mathbf{b}^k$. Then $\mathbf{I}^k = \Sigma_{i=1}^N\, b_i^k\, \mathbf{u}^i = U\, \mathbf{b}^k$ $= U\,(U^T\mathbf{I}^k) = (UU^T)\,\mathbf{I}^k$
    - Having orthonormal columns means $U^T = U^{-1}$, so UU^T = the identity matrix **Id**
    - Also, orthonormality means $U^TU =$ **Id** and mult'n by U is a rotation.
  - Let the approximation of $\mathbf{I}^k$ up to the J^{th} basis function be $\mathbf{I}^{k,J} = \Sigma_{i=1}^J\, b_i^k\, \mathbf{u}^i = U\, \mathbf{c}^k$, where $c_i^k = b_i^k$ = if $i \le J$ and zero otherwise

Orthogonal Decomposition Specialized to Your Data: what is to be proved

- Consider an orthonormal basis $\{\mathbf{u}^i\}$, to be discovered from the training images $\mathbf{I}^k$
  - From the previous slide,
    - $\mathbf{I}^k = U\,\mathbf{b}^k$
    - $\mathbf{I}^k = UU^T\mathbf{I}^k$
    - $\mathbf{I}^{k,J} = VV^T\mathbf{I}^k$, where V is the same as U in its first J columns and 0 in the remaining columns
    - Thus $\mathbf{I}^{k,1} = \mathbf{u}^1\mathbf{u}^{1T}\mathbf{I}^k$
  - We will prove that if $(1/n)\Sigma_k|\mathbf{I}^k - \mathbf{I}^{k,1}|^2$ is at a minimum over all choices of basis function $\mathbf{u}^1$,

    $\Sigma_k(b^k{}_1)^2/\ \Sigma_{k,j}(b^k{}_j)^2 = \Sigma_k(\mathbf{I}^k \bullet \mathbf{u}^1)^2\ /\ \Sigma_{k,j}{}^N\,(\mathbf{I}^k \bullet \mathbf{u}^j)^2$ is at a maximum
    - Re the denominators in the 2nd and 3rd expressions: Parseval's theorem shows that $\Sigma_{j=1}{}^N(b^k{}_j)^2 = \Sigma_{j=1}{}^N\,(\mathbf{I}^k \bullet \mathbf{u}^j)^2$ is fixed for that k, independent of the basis chosen, so the sum of those over k is independent of that basis choice
    - It follows from Parseval's theorem that the part of the $\mathbf{I}^k$ to be expanded in the $\mathbf{u}^j$ for j>1 is *minimum*
      - Also for j>2, j>3, …, j>N-1

Orthogonal Decomposition Specialized to Your Data: what is to be proved

- To prove that if $(1/n)\Sigma_k|\mathbf{I}^k - \mathbf{I}^{k,1}|^2$ is minimum over all choices of basis function $\mathbf{u}^1$,
  $\Sigma_k(\mathbf{b}^k{}_1)^2/ \Sigma_{k,j}(\mathbf{b}^k{}_j)^2 = \Sigma_k(\mathbf{I}^k \bullet \mathbf{u}^1)^2 / \Sigma_{k,j} (\mathbf{I}^k \bullet \mathbf{u}^j)^2$ is maximum
  - $\Sigma_k|\mathbf{I}^k - \mathbf{I}^{k,1}|^2 = \Sigma_k(\mathbf{I}^k - \mathbf{I}^{k,1})^T(\mathbf{I}^k - \mathbf{I}^{k,1}) =$
    $\Sigma_k(\mathbf{I}^k - \mathbf{u}^1\mathbf{u}^{1^T}\mathbf{I}^k)^T(\mathbf{I}^k - \mathbf{u}^1\mathbf{u}^{1^T}\mathbf{I}^k) = \Sigma_k\mathbf{I}^{k^T}\mathbf{I}^k$
    $- (\Sigma_k(\mathbf{I}^{k^T}\mathbf{u}^1\mathbf{u}^{1^T})\mathbf{I}^k - \Sigma_k\mathbf{I}^{k^T}(\mathbf{u}^1\mathbf{u}^{1^T}\mathbf{I}^k))$
    $+ \Sigma_k\mathbf{I}^{k^T}\mathbf{u}^1\mathbf{u}^{1^T}\mathbf{u}^1\mathbf{u}^{1^T}\mathbf{I}^k$
    - The first term in the final sum does not depend on the basis
    - Both components in the second term pair are the same, so the second term is -2 $\Sigma_k\mathbf{I}^{k^T}\mathbf{u}^1\mathbf{u}^{1^T}\mathbf{I}^k$
    - Due to normality $\mathbf{u}^{1^T}\mathbf{u}^1=1$, so the third term collapses to $\Sigma_k\mathbf{I}^{k^T}\mathbf{u}^1\mathbf{u}^{1^T}\mathbf{I}^k$, so combining the 2nd and 3rd terms gives - $\Sigma_k\mathbf{I}^{k^T}\mathbf{u}^1\mathbf{u}^{1^T}\mathbf{I}^k$
    - So the term to be minimized is $-\Sigma_k\mathbf{I}^{k^T}\mathbf{u}^1\mathbf{u}^{1^T}\mathbf{I}^k$
    - That is, maximize $\Sigma_k\mathbf{I}^{k^T}\mathbf{u}^1\mathbf{u}^{1^T}\mathbf{I}^k$, which can be rewritten as $\Sigma_k(b^k{}_1)^2$, thus also maximizing $\Sigma_k(b^k{}_1)^2/ \Sigma_{k,j}(b^k{}_j)^2$

Orthogonal Decomposition Specialized to Your Data: Rewriting in terms of the data matrix

- We wish maximize $\Sigma_k \mathbf{I}^{k^T} \mathbf{u}^1 \mathbf{u}^{1^T} \mathbf{I}^k$
- $\Sigma_k \mathbf{I}^{k^T} \mathbf{u}^1 \mathbf{u}^{1^T} \mathbf{I}^k = \Sigma_k \mathbf{u}^{1^T} \mathbf{I}^k \, \mathbf{I}^{k^T} \mathbf{u}^1 = \mathbf{u}^{1^T} (\Sigma_k \mathbf{I}^k \, \mathbf{I}^{k^T}) \, \mathbf{u}^1$
- The $\mathbf{I}^{k^T}$ are the rows of the data matrix A, so $\Sigma_k \mathbf{I}^k \, \mathbf{I}^{k^T} = A^T A$, an N×N (square), symmetric matrix that does not depend on $\mathbf{u}^1$
- Maximizing $\mathbf{u}^{1^T} A^T A \, \mathbf{u}^1$ is accomplished by choosing $\mathbf{u}^1$ to be the eigenvector of $A^T A$ with the maximum eigenvalue, as proven in the next 3 slides
  - Also proven there is that $A^T A$ has only non-negative eigenvalues
  - This is called *singular value decomposition* of A

The Major Points So Far
Re Avg Case Best MS Fitting

- An important linear algebra theorem:
  For all $n \times N$ matrices M, $M = R\Gamma S^T = R_1^{n \times n} \, \Gamma^{n \times N} \, R_2^{N \times N}$
  - Each R_i is a rotation matrix
  - $\Gamma^{n \times N}$ is "diagonal" (zeroes everywhere except on $n \times n$ diagonal if $n<N$)

- For all $\mathbf{I}$ and all $U^{N \times N}$ = columns of orthonormal basis vectors, if $\mathbf{I} = \Sigma_{i=1}^{N} b_i \, \mathbf{u}^i$, $\Sigma_{i=1}^{N} (b_i)^2$ is independent of U

- With $\mathbf{I}^{k,1} = c^k{}_1 \, \mathbf{u}^1$, to minimize the average (over k and pixels) MS error between $\mathbf{I}^{k,1}$ and $\mathbf{I}^k$, set $c^k{}_1 = b^k{}_1$ and maximize
  $\mathbf{u}^{1^T} (\Sigma_k \mathbf{I}^k \, \mathbf{I}^{k^T}) \, \mathbf{u}^1 = \mathbf{u}^{1^T} \, A^T A \, \mathbf{u}^1$

Towards Analysis of A^TA
This is basic linear algebra

- Data list A with N features and n instances
  - $n \times N$ array $A = R\Gamma S^{-1} = R\Gamma S^T$ (typically $n << N$)
    - S is $N \times N$, R is $n \times n$, Γ is $n \times N$
    - R and S are orthonormal, rotation matrices; Γ is $n \times N$ with $\Gamma_{ij} = 0$ if $i \neq j$ ("diagonal", but not necessarily square)
    - $R^{-1} = R^T$; $S^{-1} = S^T$
    - $R^TA = \Gamma S^T$; $AS = R\Gamma$
      - The columns of R (n-vectors) and of S (N-vectors) are called respectively the left and right eigenvectors of A
        - » Both are orthonormal bases of their respective (n- and N-) spaces
      - Transposing the first expression gives $A^TR = S\Lambda$, so the left eigenvectors of A are the right eigenvectors of A^T and vice-versa
- Consider A^TA ($N \times N$, big, symmetric) = $S\Gamma^TR^TR\Gamma S^T = S(\Gamma^T\Gamma)S^T = S(\Gamma^T\Gamma)S^{-1}$
  - $N \times N$ $\Gamma^T\Gamma$ is diagonal with eigenvalues of A^TA

Analysis of A^TA

This is basic linear algebra

- Data list A with N features and n instances
- Consider A^TA ($N \times N$, big, symmetric) = $S\Gamma^T R^T S \Gamma R^T = S(\Gamma^T\Gamma)S^T$
  - $\Gamma^T\Gamma$ is an $N \times N$ diagonal matrix
  - Thus S is a matrix of (orthonormal) eigenvectors of A^TA, and the diagonal elements of $\Gamma^T\Gamma$ are eigenvalues of A^TA
  - The diagonal elements of $\Gamma^T\Gamma$ are zeros or squares of the diagonal elements of the $n \times N$ matrix Γ
  - Thus A^TA is "non-negative definite"
- Arrange the columns and rows of $\Gamma^T\Gamma$ in decreasing order of its diagonal elements
  - The columns of S must be reordered accordingly

Over all unit $\mathbf{u}$, the one that maximizes $\mathbf{u}^T A^T A\, \mathbf{u}$

- Data list A with N features and n instances
- Consider $A^T A = S(\Gamma^T\Gamma)S^T$
  - $\mathbf{u}^T A^T A\, \mathbf{u} = \mathbf{u}^T S(\Gamma^T\Gamma)S^T\mathbf{u} = \Sigma_{i=1}^{N}(\Gamma^T\Gamma)_{ii}\, w_i^2$, where $\mathbf{w}=S^T\mathbf{u}$

    Thus $\mathbf{u}=S\mathbf{w}$
  - To maximize $\Sigma_{i=1}^{N}(\Gamma^T\Gamma)_{ii}\, w_i^2$ with $|\mathbf{w}|=1$, $w_1=1$ and $w_i=0$ for $i>1$
  - Thus $\mathbf{u}$ should be the first column of S, i.e., the eigenvector of $A^T A$ with the largest magnitude eigenvalue
- Similar considerations show that the next (with J=2) major part of the error in approximation of **I** is given by choosing the 2nd eigenvector (in decreasing order of eigenvalues) of $A^T A$ as $\mathbf{u}^2$
  - Etc. for successive $\mathbf{u}^i$ choices; i.e., U=S

Singular Value Decomposition via eigenanalysis of AA^T

- Data list A with N features and n instances
  - $N \times n$ array $A = R\Gamma S^T$ (typically $n << N$)
    - S is $N \times N$, R is $n \times n$
    - R and S are orthonormal, rotation matrices Γ is $n \times N$ with $\Gamma_{ij} = 0$ if $i \neq j$ ("diagonal")
- Consider AA^T ($n \times n$, small, symmetric)
  - $AA^T = R\Gamma S^T S \Gamma^T V^T = R(\Gamma\Gamma^T)R^T$; $\Gamma\Gamma^T$ is $n \times n$ diag.
  - So columns of R are (orthonormal) eigenvectors of AA^T and diagonal elements of $\Gamma\Gamma^T$ are (non-negative) eigenvalues of AA^T
  - Diagonal elements of AA^T (after reordering in decreasing order) are the first n diagonal entries in $\Gamma^T\Gamma$, the eigenvalues of A^TA
    - The rest of the diagonal elements of $\Gamma^T\Gamma$, the final N-n eigenvalues of A^TA, are zero
  - Considering $AA^T\mathbf{v}=\lambda\mathbf{v}$, $A^TAA^T\mathbf{v} = \lambda A^T\mathbf{v}$, i.e., $A^T\mathbf{v}$ is an eigenvector of A^TA with eigenvalue λ, so $\mathbf{u}^i$ = normalized $A^T\mathbf{v}^i$
    - Thus solve eigenproblem on AA^T, then convert to the $\mathbf{u}^i$ we need

Principal Component Analysis

- A standard approach in statistics for lowering the number of features used
- It is SVD after modifying the training cases by subtracting out the mean of the training cases from each case
  - For images, the mean is also an image
    - So eigendirections are through mean rather than through origin
  - $A^TA/(n-1)$ is the estimated covariance matrix of the features, i.e., of the pixel values

Summary of Specialized Basis Functions

- As with Fourier basis functions they are eigenimages, but of $A^T A$, where A is the training matrix
- They are the most efficient set, according to the least squares measure
- Their compression effectiveness rises strongly when the images are well aligned and the intensity values are well normalized
  - But they are specialized to the training images and thus do not compress well images not like the training set
- They do not provide fast implementation of shift-invariant operators