

---

# Numerical Linear Algebra and Image Restoration

---

Maui High Performance Computing Center  
Friday, October 10, 2003

James G. Nagy  
Emory University  
Atlanta, GA

## Review from Wednesday

- **Problem:**  $\mathbf{b} = A\mathbf{x} + \mathbf{e}$
- **Solution:** Compute a filtered solution:

$$\mathbf{x}_{\text{filt}} = \sum_{i=1}^n \phi \frac{\mathbf{u}_i^T \mathbf{b}}{\sigma_i} \mathbf{v}_i$$

- **Decomposition:** Depending on matrix structure, use
  - (a) Spectral decomposition:  $A = V^* \Lambda V$
  - (b) Singular value decomposition:  $A = U \Sigma V^T$
  - (c) Or, an approximation of one of these.

## Review from Thursday

### Matrix Structures

BC	PSF Not Separable	PSF Separable
zero	BTTB	Toep $\otimes$ Toep
periodic	BCCB	Circ $\otimes$ Circ
reflexive	BTTB+BTHB +BHTB+BHHB	(Toep+Hank) $\otimes$ (Toep+Hank)
reflexive with symm. PSF	same as reflexive with symmetry	same as reflexive with symmetry

## Review from Thursday

### Matrix Structures

BC	PSF Not Separable	PSF Separable
zero	try SVD approximation	use SVD
periodic	use FFT	use FFT
reflexive	try SVD approximation	use SVD
reflexive with symm. PSF	use DCT	use DCT

Easy cases use FFT, DCT or Kronecker product SVD.

## Review from Thursday

### SVD Approximations:

- Construct the approximation:

$$A = C_1 \otimes D_1 + C_2 \otimes D_2 + \cdots + C_k \otimes D_k$$

- Compute an SVD of the dominant term:

$$C_1 \otimes D_1 = (U_c \otimes U_d)(\Sigma_c \otimes \Sigma_d)(V_c \otimes V_d)^T$$

- Determine which of the following gives best approximation of a diagonal matrix:

$$\mathcal{F}A\mathcal{F}^* = \mathcal{F} \left( \sum C_i \otimes D_i \right) \mathcal{F}^*$$

$$\mathcal{C}A\mathcal{C}^T = \mathcal{C} \left( \sum C_i \otimes D_i \right) \mathcal{C}^T$$

$$(U_c \otimes U_d)^T A (V_c \otimes V_d) = (U_c \otimes U_d)^T \left( \sum C_i \otimes D_i \right) (V_c \otimes V_d)$$

## Review from Thursday

### SVD Approximations:

Use  $A \approx U\Sigma V^T$ , (or  $A \approx U\Sigma V^*$ ) where

- If FFT is best,

$$U = V = \mathcal{F}^*, \quad \Sigma = \text{diag} \left( \mathcal{F} \left( \sum C_i \otimes D_i \right) \mathcal{F}^* \right)$$

- If DCT is best,

$$U = V = \mathcal{C}^T, \quad \Sigma = \text{diag} \left( \mathcal{C} \left( \sum C_i \otimes D_i \right) \mathcal{C}^T \right)$$

- If separable SVD is best,

$$U = U_c \otimes U_d, \quad V = V_c \otimes V_d,$$

$$\Sigma = \text{diag} \left( (U_c \otimes U_d)^T \left( \sum C_i \otimes D_i \right) (V_c \otimes V_d) \right)$$

## Friday Topics

- Spatially variant blurs
  - Model for  $A$  using multiple PSFs.
  - Kronecker product approximations from the PSFs.
  - SVD approximations.
- Accelerating iterative methods.
- Wrap up.

## Space Variant Model for $A$

Recall, using linear algebra notation:

- The  $i$ -th column of  $A$  can be written as:

$$A\mathbf{e}_i = \begin{bmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_i & \cdots & \mathbf{a}_n \end{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} = \mathbf{a}_i$$

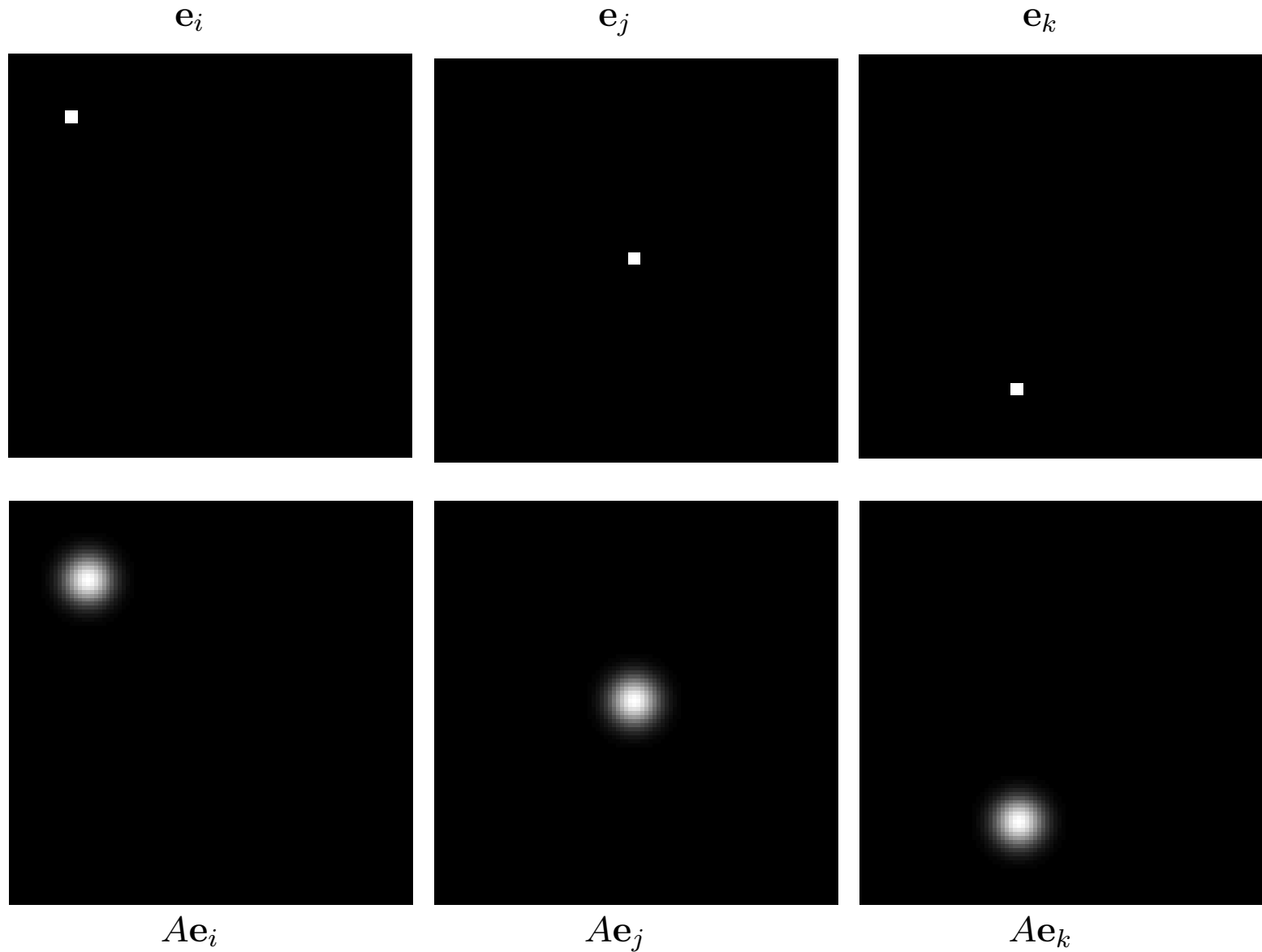
- In an imaging system,

$$\mathbf{e}_i = \text{point source}$$

$$A\mathbf{e}_i = \text{point spread function (PSF)}$$

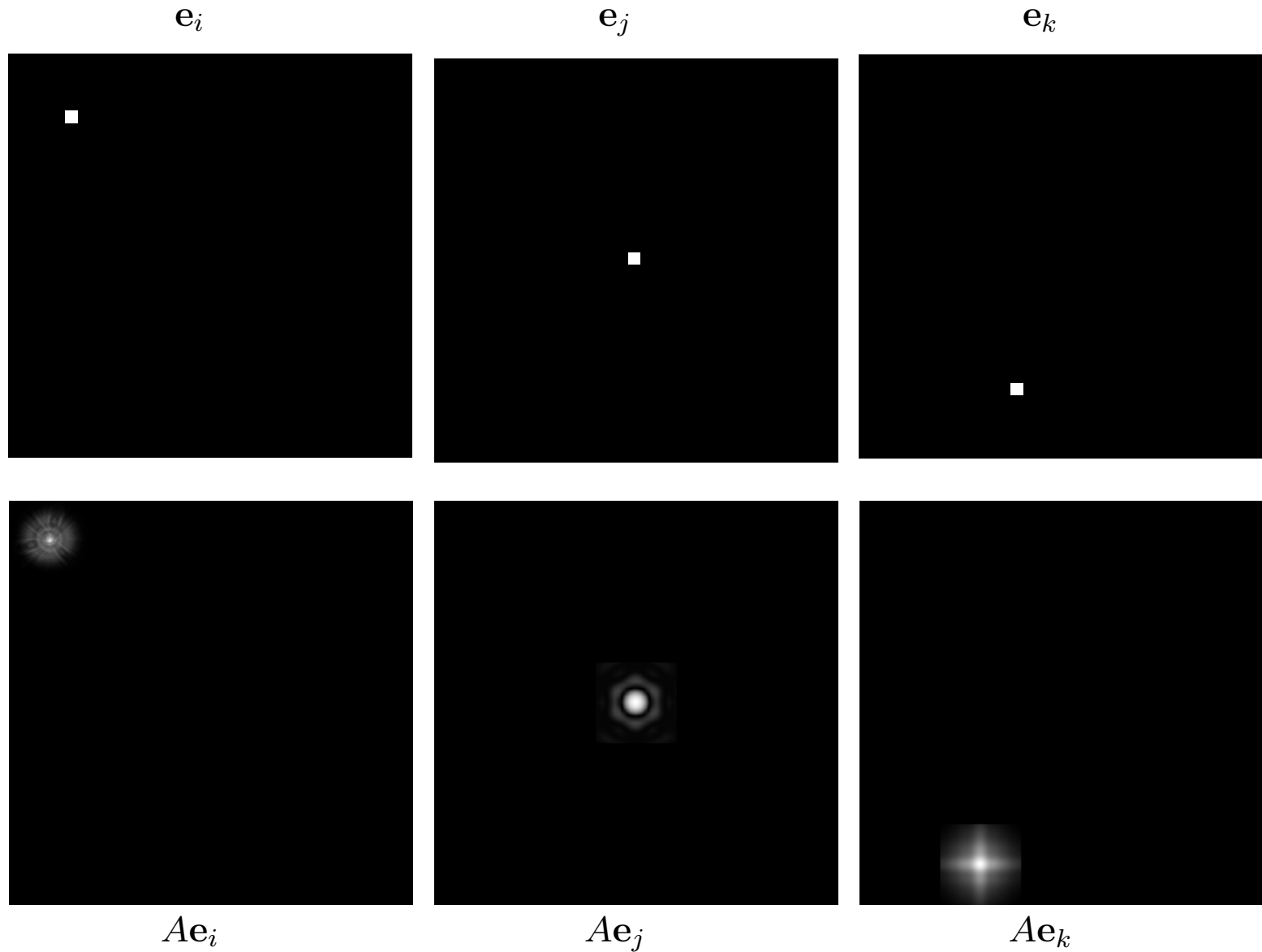
# Space Variant Model for $A$

Spatially invariant PSF



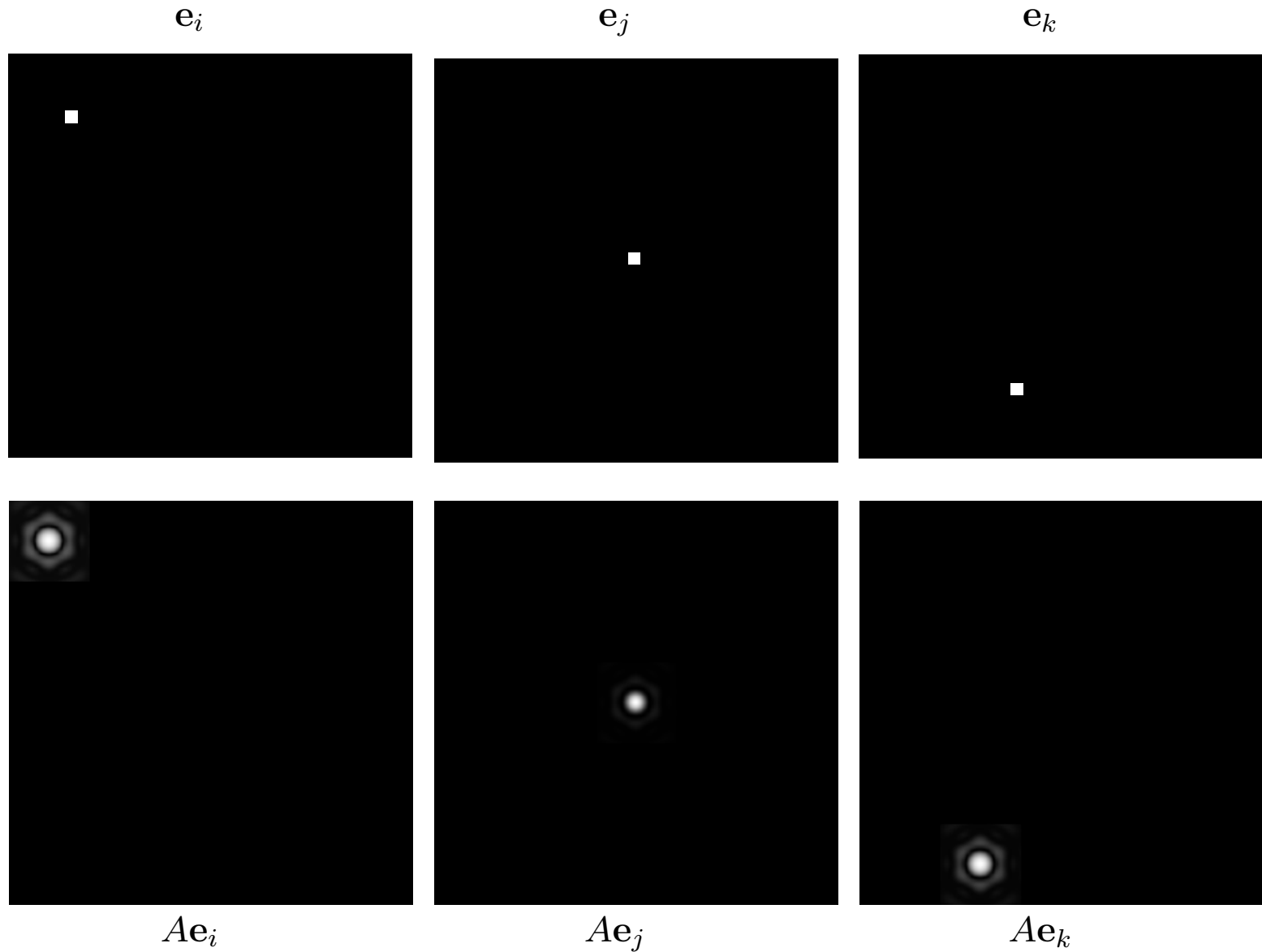
# Space Variant Model for $A$

Spatially variant PSF



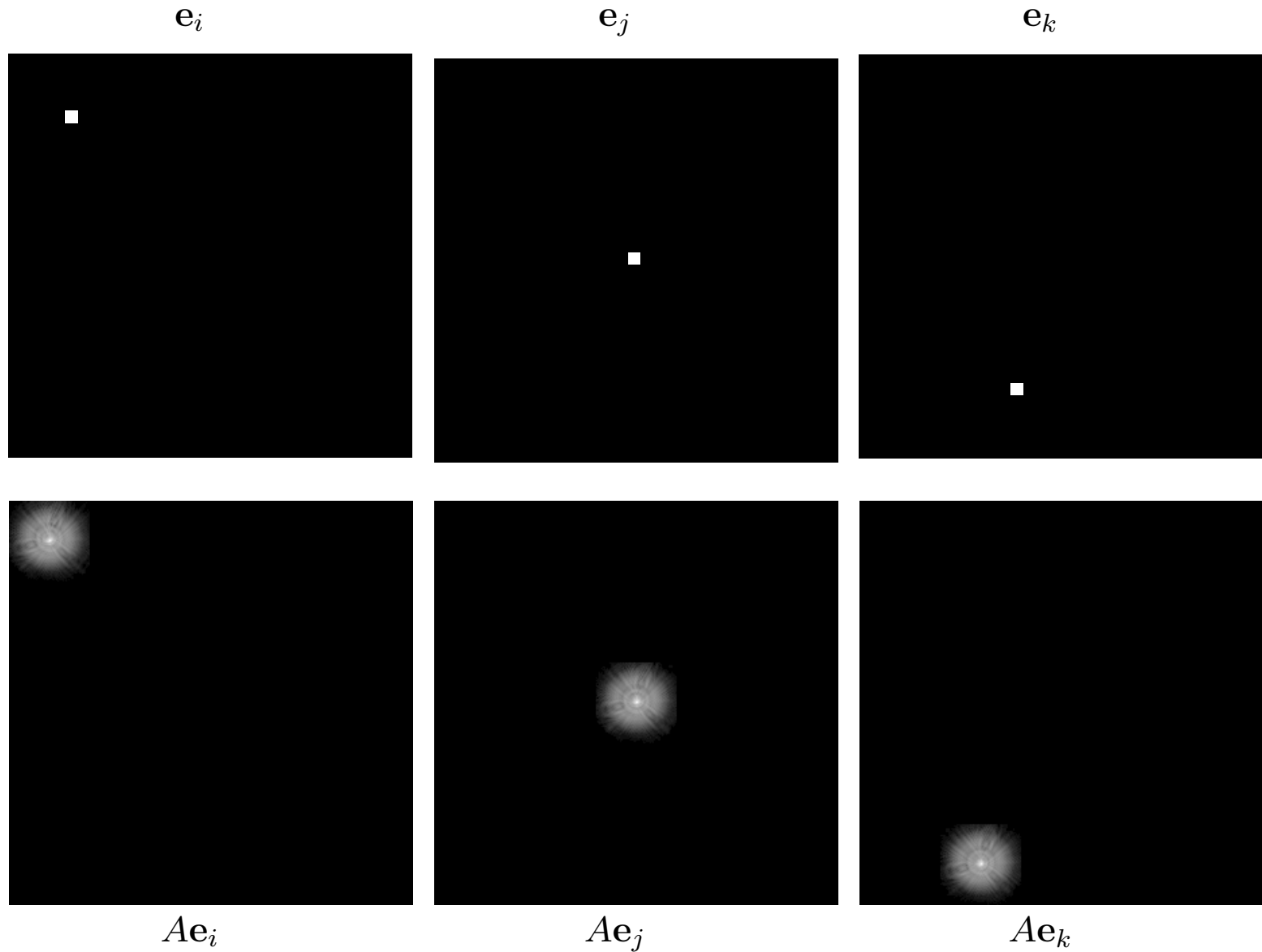
# Space Variant Model for $A$

Spatially variant PSF



# Space Variant Model for $A$

Spatially variant PSF



## Space Variant Model for $A$

That is, spatially variant implies

- Each column of  $A$  is different.  
(But maybe only slightly different.)
- $A$  is **not** Toeplitz (**not** circulant).
- Need a point source centered at each pixel to completely construct  $A$ .

$\Rightarrow$  need  $n^2$  PSFs

## Space Variant Model for $A$

Suppose the blur is separable:  $A = C \otimes D$

For example, consider  $3 \times 3$  images:

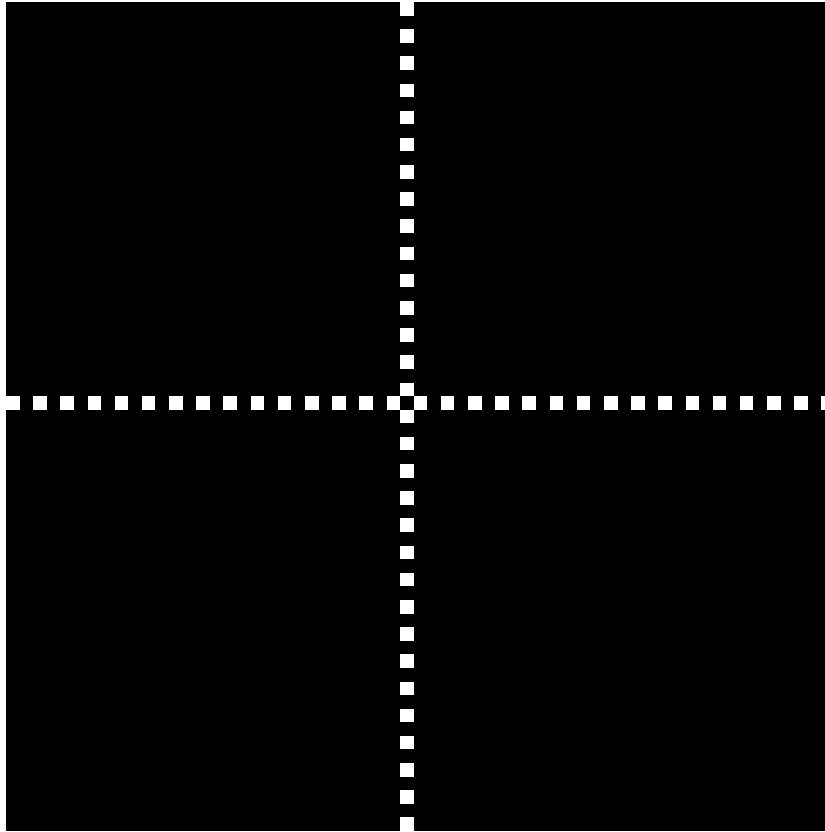
$$\begin{aligned} A &= \overbrace{\left[ \mathbf{c}_1 \mid \mathbf{c}_2 \mid \mathbf{c}_3 \right]}^{3 \text{ columns}} \otimes \overbrace{\left[ \mathbf{d}_1 \mid \mathbf{d}_2 \mid \mathbf{d}_3 \right]}^{3 \text{ columns}} \\ &= \underbrace{\left[ \mathbf{c}_1 \otimes \mathbf{d}_1 \mid \mathbf{c}_1 \otimes \mathbf{d}_2 \mid \mathbf{c}_1 \otimes \mathbf{d}_3 \mid \mathbf{c}_2 \otimes \mathbf{d}_1 \mid \mathbf{c}_2 \otimes \mathbf{d}_2 \mid \cdots \right]}_{9 \text{ columns}} \end{aligned}$$

**Notice:** For an image with  $n \times n$  pixels:

- $A$  has  $n^2$  columns of length  $n^2$ .
- If blur is separable, need only  $2n$  columns of length  $n$ .
- That is, for separable blur, need only  $2n$  PSFs.

## Space Variant Model for $A$

For separable blur, if the  $2n$  point sources are placed as:



## Space Variant Model for $A$

Then we get the  $2n$  PSFs:

$$\begin{bmatrix} & & & \mathbf{c}_1 \mathbf{d}_j^T & & \\ & & & \mathbf{c}_2 \mathbf{d}_j^T & & \\ & & & \vdots & & \\ \mathbf{c}_i \mathbf{d}_1^T & \mathbf{c}_i \mathbf{d}_2^T & \dots & \dots & \dots & \mathbf{c}_i \mathbf{d}_n^T \\ & & & \vdots & & \\ & & & \mathbf{c}_n \mathbf{d}_j^T & & \end{bmatrix}$$

## Space Variant Model for $A$

- If blur is not separable, we can use the  $2n$  PSFs to get a separable approximation.
- If it is possible to get some PSFs, but not all  $2n$ , then what do we do?
  - Gather PSFs,  $P_{ij}$ , in various regions.
  - Assume blur is locally spatially invariant.
  - Use interpolation to construct  $A$ :

$$A = \sum_{i=1}^p \sum_{j=1}^p I_{ij} A_{ij}$$

where  $\sum I_{ij} = I$ , and  $A_{ij}$  is defined by PSF  $P_{ij}$ .

## Space Variant Model for $A$

### Remarks:

- Space variant model is:

$$A = \sum_{i=1}^p \sum_{j=1}^p I_{ij} A_{ij}$$

- Each  $A_{ij}$  is defined by a spatially invariant PSF, so iterative image restoration methods can be implemented efficiently.
- It is possible to form Kronecker product, and SVD approximations.

## Space Variant SVD Approximations

Using the model: 
$$A = \sum_{i=1}^p \sum_{j=1}^p I_{ij} A_{ij}$$

- Each  $A_{ij}$  can be decomposed as:

$$A_{ij} = \sum_{k=1}^r C_k^{(i)} \otimes D_k^{(j)}$$

- Therefore, our model for  $A$  is:

$$A = \sum_{i=1}^p \sum_{j=1}^p I_{ij} \left( \sum_{k=1}^r C_k^{(i)} \otimes D_k^{(j)} \right)$$

- It can be shown that  $I_{ij} = \hat{I}_i \otimes \tilde{I}_j$

Now for a little algebra ...

## Space Variant SVD Approximations

... by exploiting properties of Kronecker products, we get:

$$\begin{aligned} A &= \sum_{i=1}^p \sum_{j=1}^p \left( \hat{I}_i \otimes \tilde{I}_j \right) \left( \sum_{k=1}^r C_k^{(i)} \otimes D_k^{(j)} \right) \\ &= \sum_{k=1}^r \sum_{i=1}^p \sum_{j=1}^p \left( \hat{I}_i C_k^{(i)} \otimes \tilde{I}_j D_k^{(j)} \right) \\ &= \sum_{k=1}^r \left( \left( \sum_{i=1}^p \hat{I}_i C_k^{(i)} \right) \otimes \left( \sum_{j=1}^p \tilde{I}_j D_k^{(j)} \right) \right) \\ &= C_k \otimes D_k \end{aligned}$$

where

$$C_k = \sum_{i=1}^p \hat{I}_i C_k^{(i)} \quad \text{and} \quad D_k = \sum_{j=1}^p \tilde{I}_j D_k^{(j)}$$

## Space Variant SVD Approximations

### Summary:

- Gather PSFs,  $P_{ij}$ , in various regions.

- For each  $P_{ij}$ ,  $i, j = 1, \dots, p$

- decompose  $P_{ij} = \sum_{k=1}^r \mathbf{c}_k \mathbf{d}_k^T$

- use  $\mathbf{c}_k, \mathbf{d}_k$  to build part of  $C_k, D_k$

- Use  $A = \sum_{k=1}^r C_k \otimes D_k$  to construct SVD approximation (as discussed in Thursday's lecture).

- Computational cost for  $n \times n$  images (i.e.,  $n^2 \times n^2$   $A$ ) is at most  $O(pn^3)$ .
-

## Accelerating Iterative Methods

Consider algorithms (e.g., EMLS, CG, ...) of the form

$\mathbf{x}_0$  = initial estimate of  $\mathbf{x}$

for  $j = 0, 1, 2, \dots$

- $\mathbf{x}_{j+1}$  = computations involving  $\mathbf{x}_j$ ,  $A$ ,  
and preconditioner,  $M$
- determine if stopping criteria are satisfied

end

Need to efficiently:

- multiply with  $A$ ,  $A^T$   
(easy, use FFTs)
- application of preconditioner

## Accelerating Iterative Methods

Typical approach to preconditioning  $A\mathbf{x} = \mathbf{b}$

Find matrix  $M$  satisfying the following properties:

- Inexpensive to “construct”  $M$ .
- Inexpensive to solve  $M\mathbf{z} = \mathbf{w}$ .
- $M^{-1}A \approx I$

“Ideal” preconditioner:  $M = A = U\Sigma V^T$

- Solving  $M\mathbf{z} = \mathbf{w} \Rightarrow \mathbf{z} = V\Sigma^{-1}U^T\mathbf{w}$ .
- For ill-posed problems, noise/errors in  $\mathbf{w}$  are amplified.
- Remedy: Regularize using **modified** truncated SVD

$$\mathbf{z} = V\Sigma^\dagger U^T\mathbf{w}$$

where  $\Sigma^\dagger = \text{diag}(1/\sigma_1, \dots, 1/\sigma_k, 1, \dots, 1)$

## Accelerating Iterative Methods

Notice that the preconditioned system is:

$$\begin{aligned}M_{\tau}^{-1}A &= (U\Sigma_{\tau}V^T)^{-1}(U\Sigma V^T) \\ &= V\Sigma_{\tau}^{-1}\Sigma V^T \\ &= V\Delta V^T\end{aligned}$$

where  $\Delta = \text{diag}(1, \dots, 1, \sigma_{k+1}, \dots, \sigma_n)$

That is,

- Large (good) singular values clustered at 1.
- Small (bad) singular values not clustered.

In cases when SVD of  $A$  is not known, we use an approximation to construct  $M_{\tau}$ .

---

## Wrap up.

- It is possible to efficiently implement filtering methods for a variety of PSFs (including space variant problems) and boundary conditions.
- Sometimes an SVD approximation must be used. In this case, a good understanding of linear algebra issues involving Kronecker products leads to optimal approximations.
- The SVD approximations can be very useful in accelerating iterative image restoration algorithms.