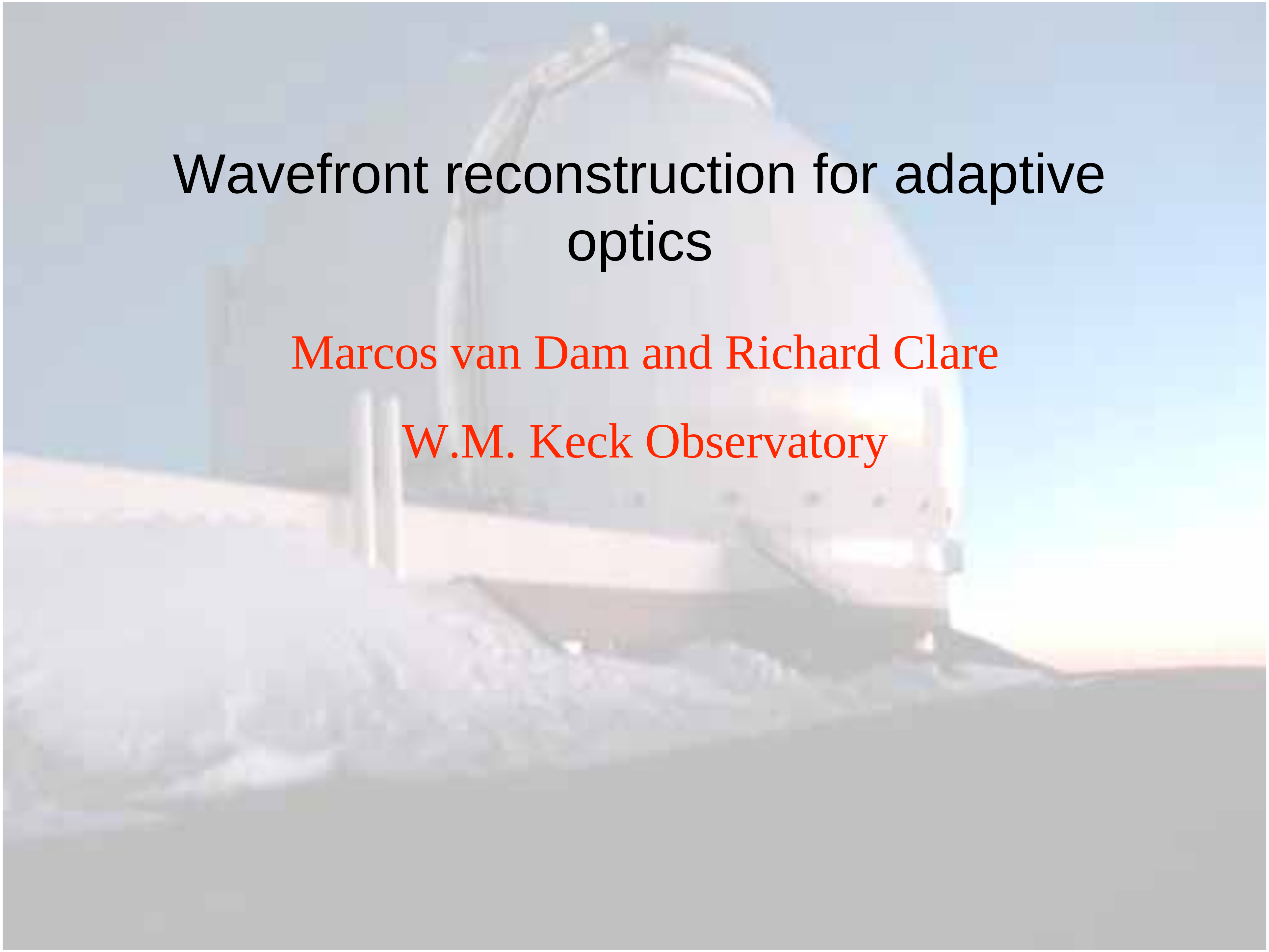


Wavefront reconstruction for adaptive optics

Marcos van Dam and Richard Clare

W.M. Keck Observatory

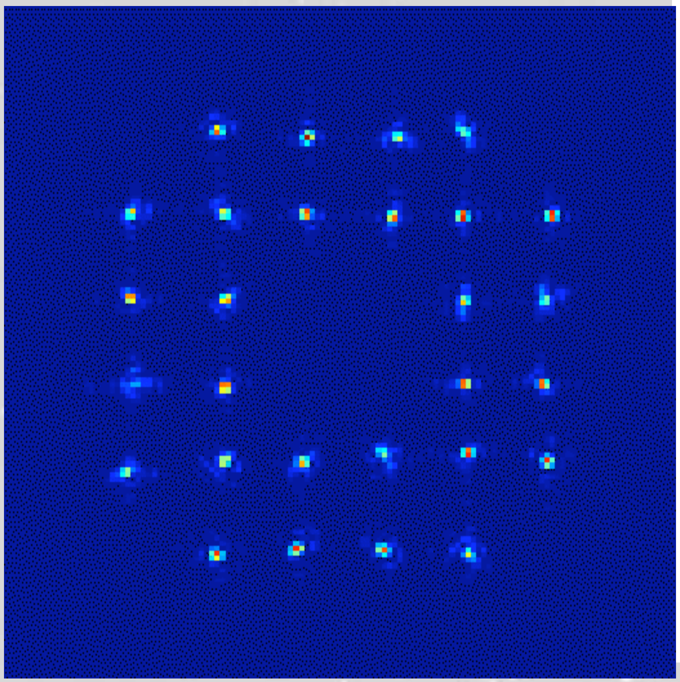
Friendly people

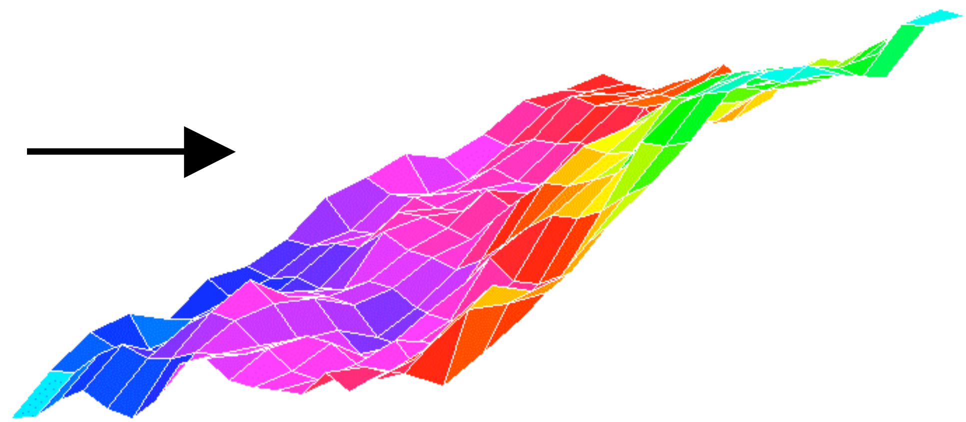
- We borrowed slides from the following people:
  - **Lisa Poyneer**
  - **Luc Gilles**
  - **Curt Vogel**
  - **Corinne Boyer**
- Mahalo!

Imke de Pater *et al*

Outline

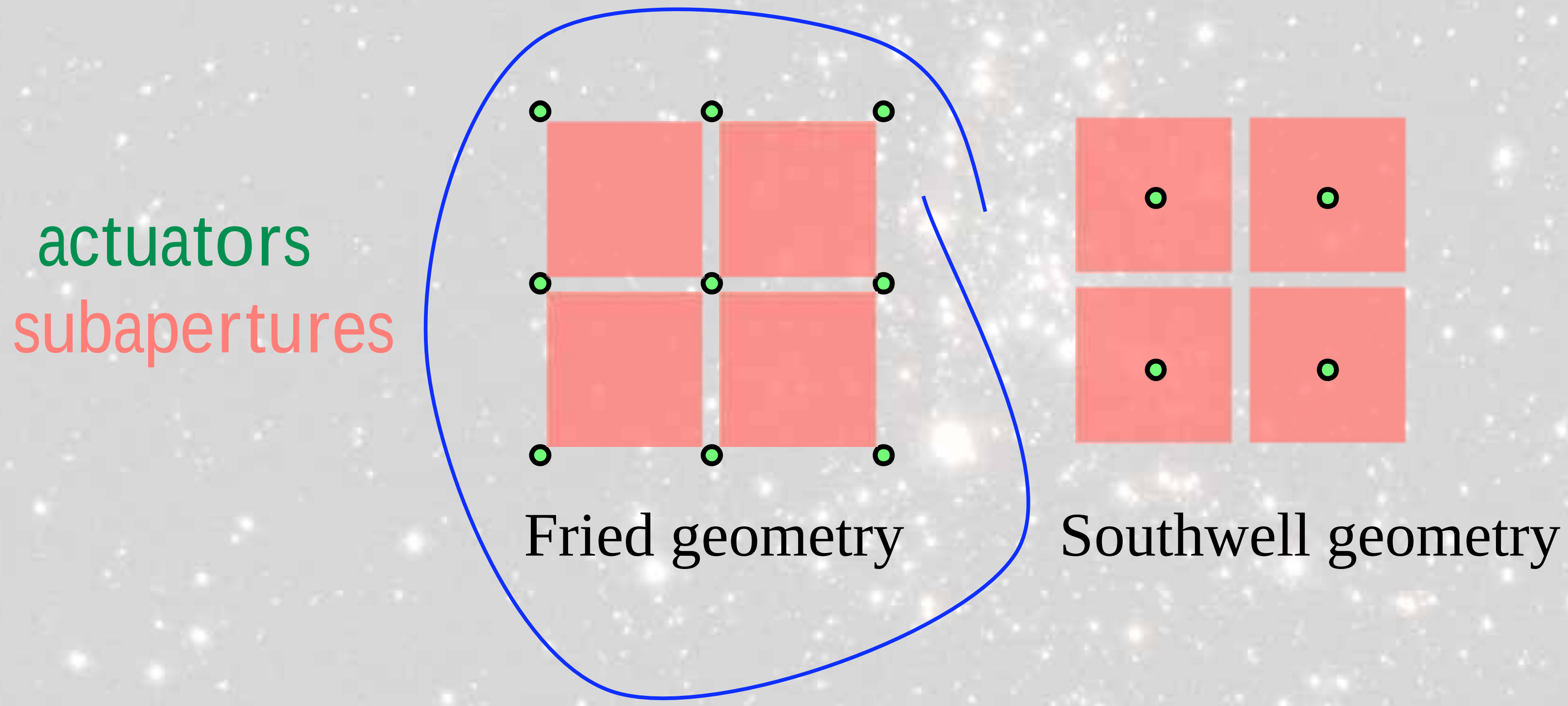
- System matrix, H: from actuators to centroids $s = Ha$
- Reconstructor, R: from centroids to actuators $a = Rs$
- Fast algorithms and hardware





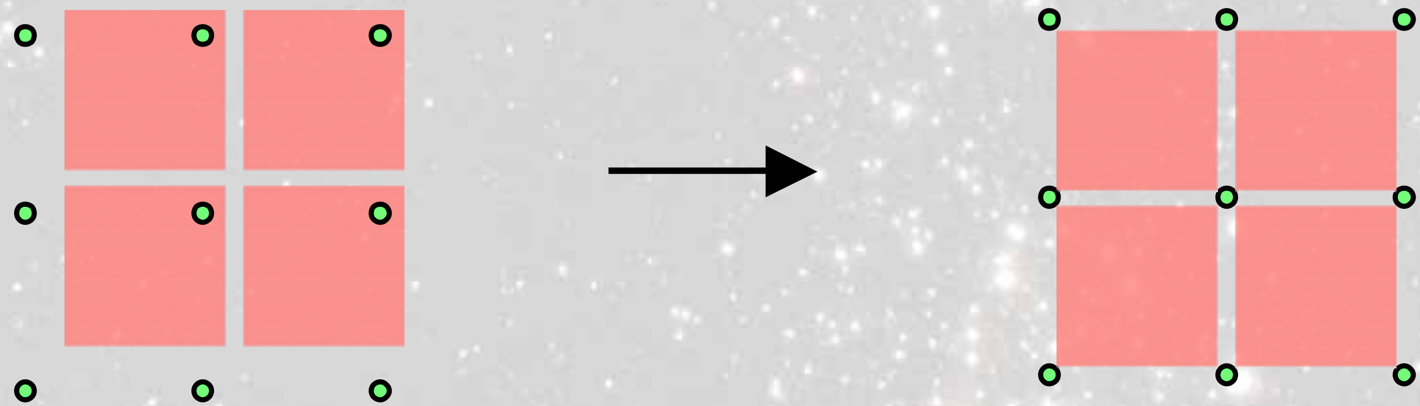
Case Study

- Consider a Shack-Hartmann wavefront sensor
- Actuators in a square array with a separation equal to the lenslet size
- Fried geometry between subapertures and actuators



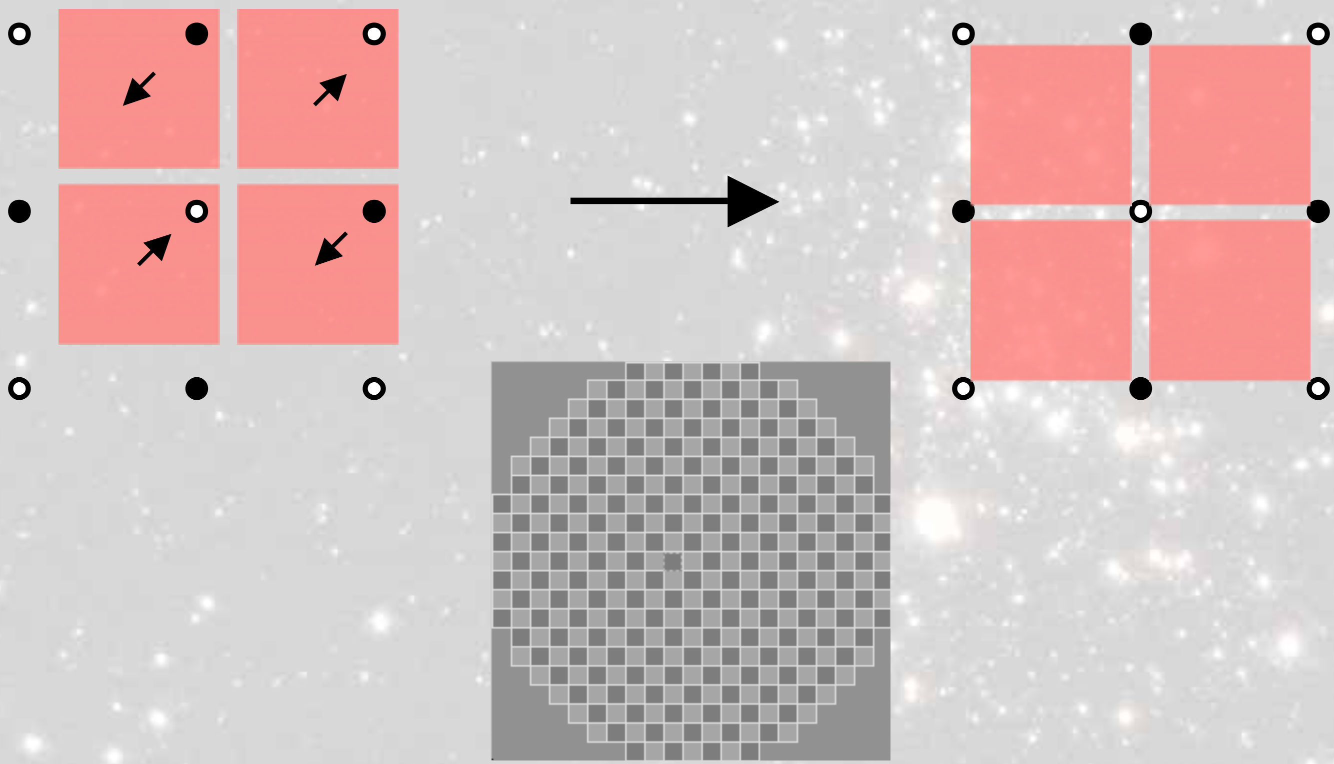
DM to lenslet registration

- How do we ensure that we have the right registration?



DM to lenslet registration

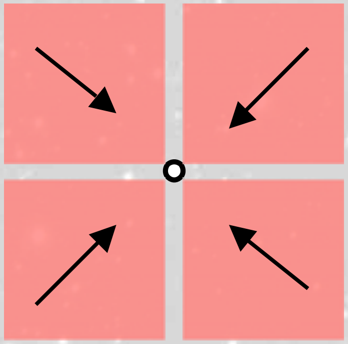
- Add waffle to the DM and adjust lenslet array or the beam until no centroids are measured



System matrix generation

- System matrix describes how a signal applied to the actuators, *a,* affects the centroids, *s.*

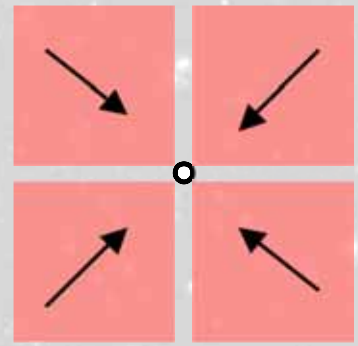
$$s = Ha$$



- Can be calculated theoretically or, preferably, experimentally

Theoretical system matrix

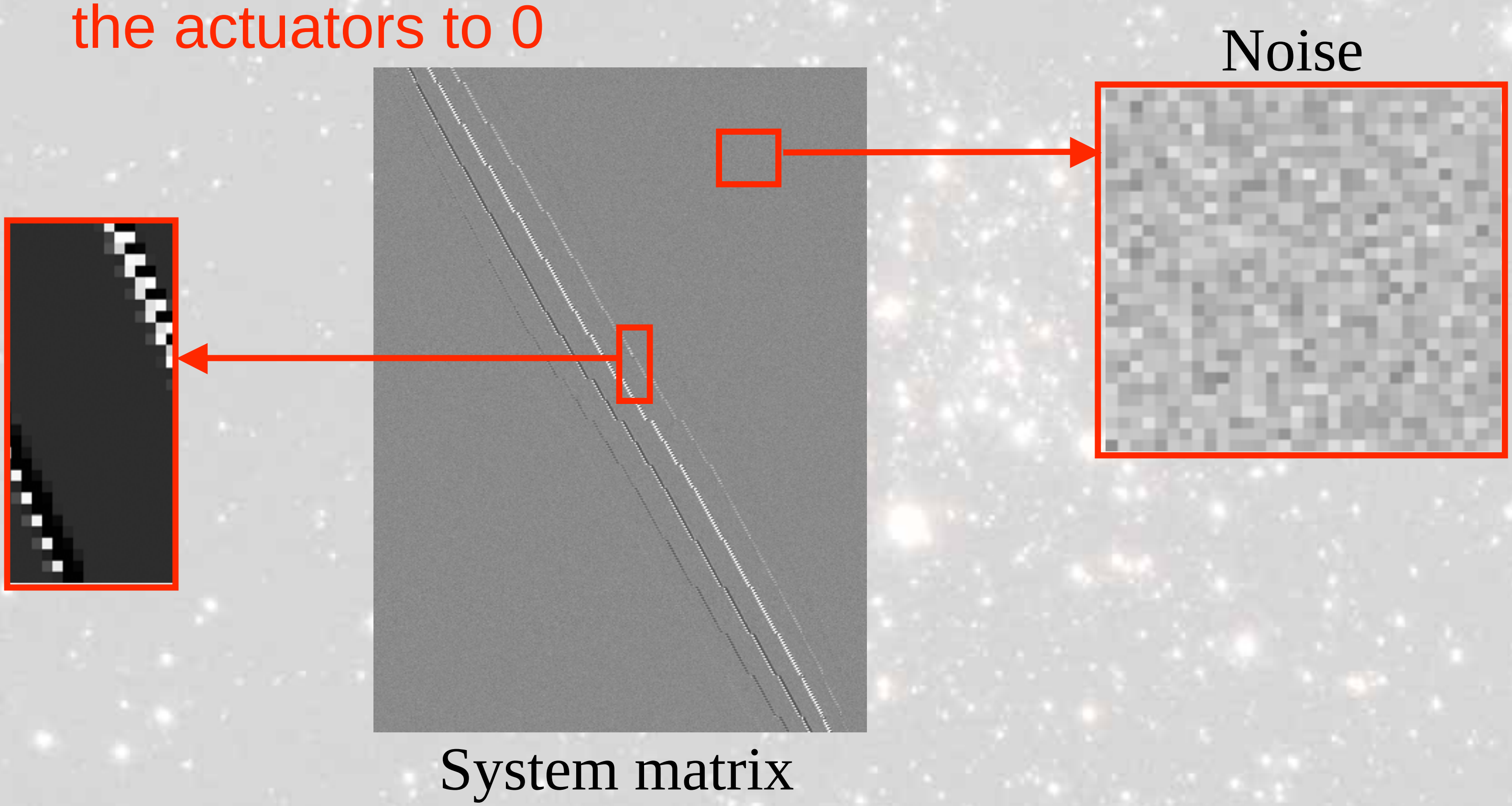
- For an actuator at (n+1/2,n+1/2),



$s_x = 1$ $s_y = -1$	$s_x = -1$ $s_y = -1$
$s_x = 1$ $s_y = 1$	$s_x = -1$ $s_y = 1$

Experimental system matrix

- Poke one actuator at a time in the positive and negative directions and record the centroids
- Set centroid values from subapertures far away from the actuators to 0



Inverting the system matrix

- We have the system matrix $s = Ha$
- We need a reconstructor matrix to convert from centroids to actuator voltages $a = Rs$

$$Ha = s$$

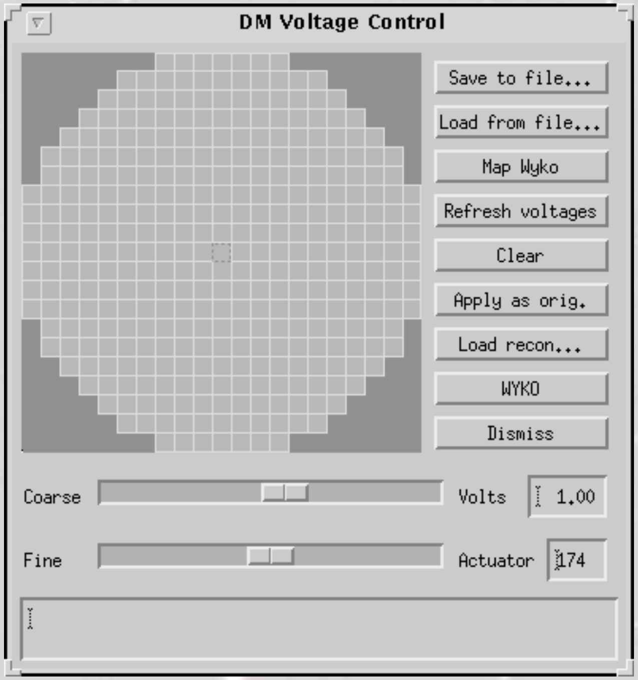
$$H^T H a = H^T s$$

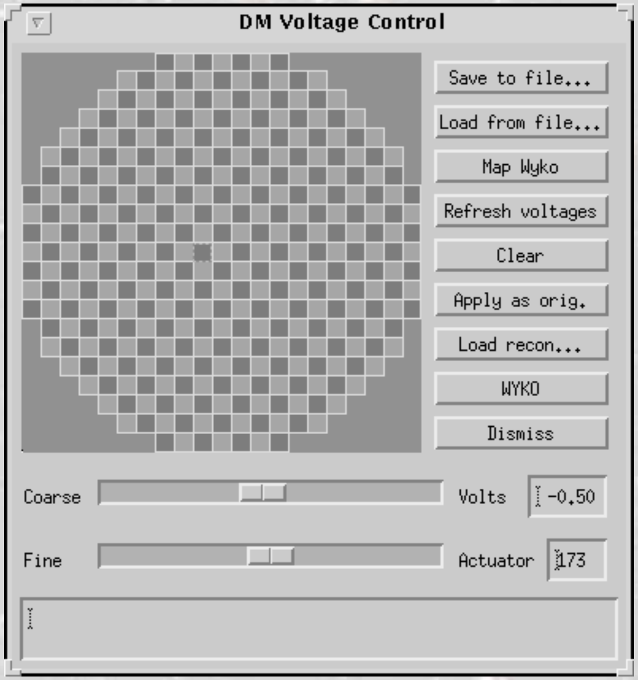
$$a = \underbrace{(H^T H)^{-1} H^T}_{R} s$$

Least-squares reconstructor R

Least-squares reconstructor

- Least squares reconstructor is $(H^T H)^{-1} H^T$
- Minimizes $(s - Ha)^2$
- But $H^T H$ is not invertible because some modes are invisible!
- Two invisible modes are piston and waffle





Singular value decomposition

- The SVD reconstructor is found by rejecting small singular values of *H*.
- Write $H = U\Lambda V^T$

$$\Lambda = \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \lambda_3 & \\ & & & 0 \end{bmatrix}$$

λ_i are the eigenvalues of $H^T H$

- The pseudo inverse is $H^+ = V\Lambda^{-1}U^T$

Singular value decomposition

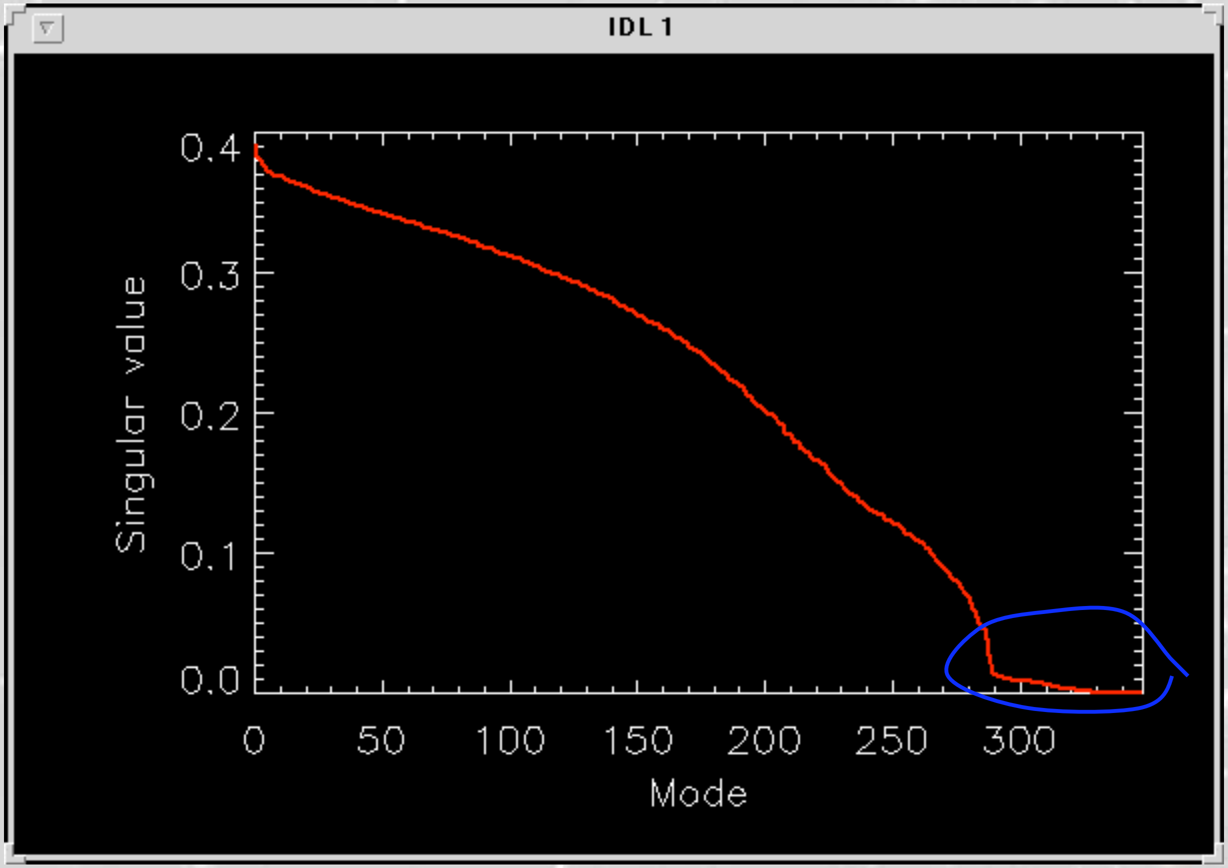
- The pseudo inverse is $H^{+} = V\Lambda^{-1}U^{T}$

$$\Lambda^{-1} = \begin{bmatrix} \lambda_1^{-1} & & & \\ & \lambda_2^{-1} & & \\ & & \lambda_3^{-1} & \\ & & & 0 \end{bmatrix}$$

- Replace all the λ_i^{-1} with 0 for small values of λ_i

Singular value decomposition

- Example: Keck Observatory



← Set to zero

Noise propagation

- Suppose we only have centroid noise in the system with variance σ^2
- Variance of actuator commands is:

$$\begin{aligned}
\mathrm{Var}(a) &= \mathrm{Var}(Rs) \\
&= \mathrm{E}[(Rs)^2] - \underbrace{(\mathrm{E}[(Rs)])^2}_{0} \\
&= \mathrm{E}[(Rs)^2] \\
&= |R|^2\,\mathrm{E}[s^2] \\
&= |R|^2\sigma^2
\end{aligned}$$

Noise propagation

- Recall $\mathrm{Var}(a) = |R|^2 \sigma^2$
- The total noise for all actuators is $|R|^2 = \sum R_{i,j}{}^2$
- For the SVD, this is equal to the sum of the singular modes of the reconstructor $|R|^2 = \sum \lambda_i{}^{-2}$

Throw away the noisiest modes!

- The average noise propagation is $|R|^2 = \sum R_{i,j}{}^2 / N$ where *N* is the number of actuators.

Least-squares reconstructor

- For well-conditioned *H* matrices, we can penalize piston, *p*, and waffle, *w*:

$$p = [1,1,1,1,1,1,\ldots]^T$$

Invertible

$$w = [1,-1,1,-1,1,\ldots]^T$$

$$R = (H^T H + pp^T + ww^T)^{-1} H^T$$

- Minimizes $\underbrace{(s - Ha)^2}_{\text{Choose the actuator voltages that best cancel the measured centroids}} + (p^T a)^2 + (w^T a)^2$

Choose the actuator voltages that
best cancel the measured centroids

Least-squares reconstructor

- For well-conditioned H matrices, just heavily penalize piston, *p*, and waffle, *w*:

$$p = [1,1,1,1,1,1,\ldots]^T$$

$$w = [1,-1,1,-1,1,\ldots]^T$$

$$R = (H^T H + pp^T + ww^T)^{-1} H^T$$

- Minimizes $(s - Ha)^2 + \underbrace{(p^T a)^2}_{\text{Choose the actuator voltages such that there is no piston}} + (w^T a)^2$

Least-squares reconstructor

- For well-conditioned H matrices, just heavily penalize piston, *p*, and waffle, *w*:

$$p = [1,1,1,1,1,1,\ldots]^T$$

$$w = [1,-1,1,-1,1,\ldots]^T$$

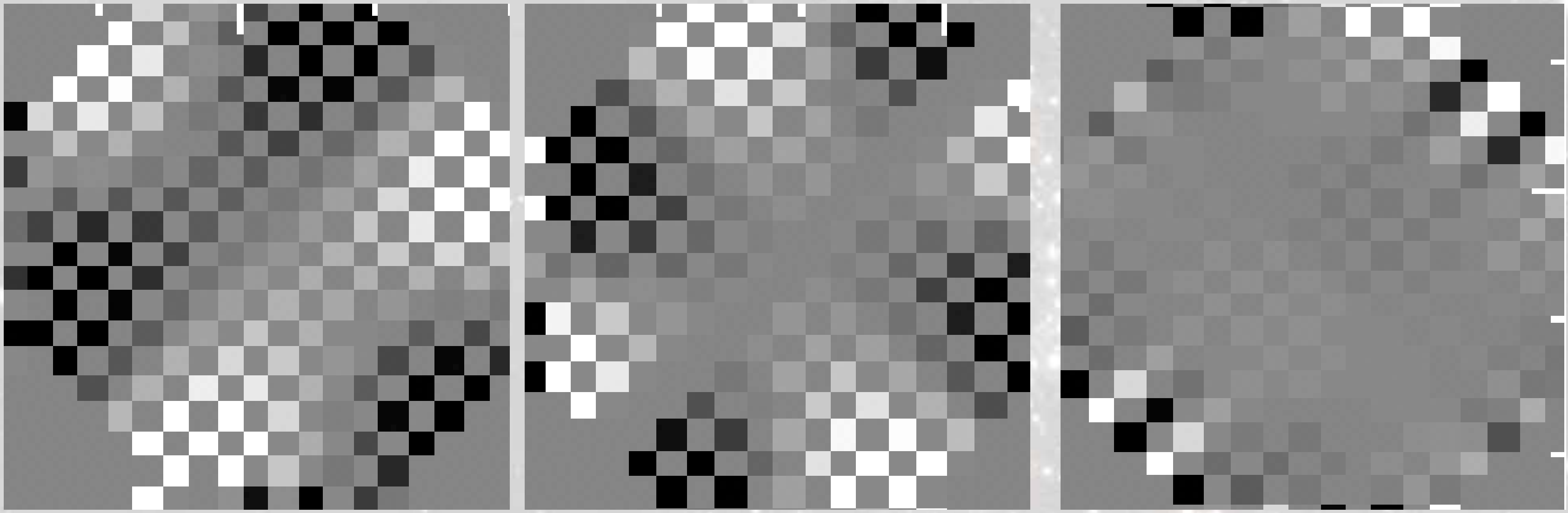
$$R = (H^T H + pp^T + ww^T)^{-1} H^T$$

- Minimizes $(s - Ha)^2 + (p^T a)^2 + \underbrace{(w^T a)^2}$

Choose the actuator voltages such that there is no waffle

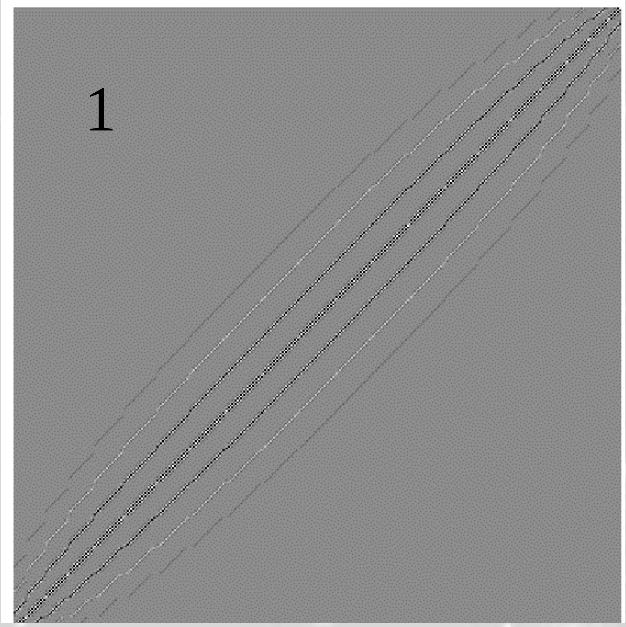
Least-squares reconstructor

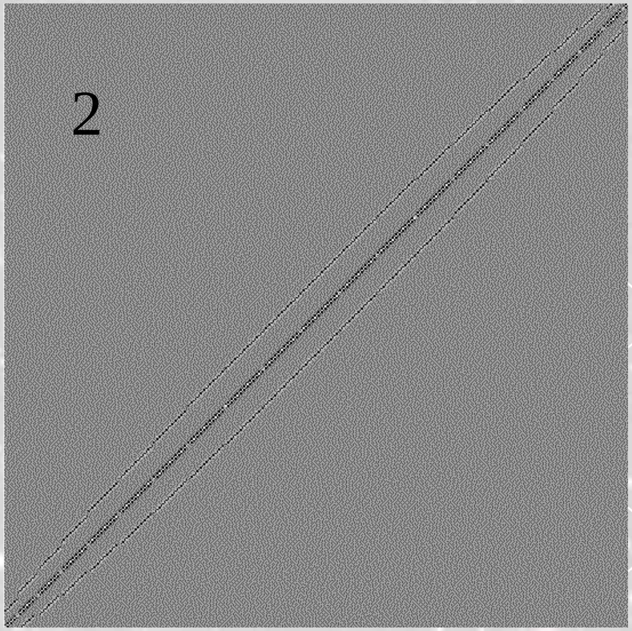
- Most modes have local waffle but no global waffle
- Must regularize before inverting



Least-squares reconstructor

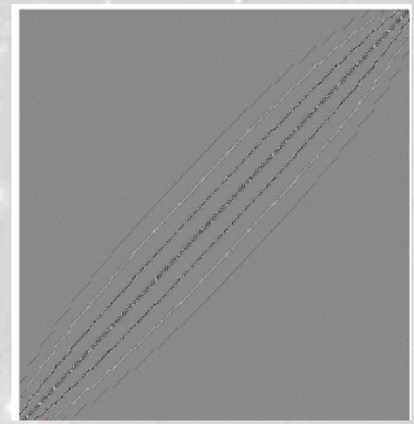
- Penalize waffle in the inversion:
  1. Inverse covariance matrix of Kolmogorov turbulence or
  2. Waffle penalization matrix



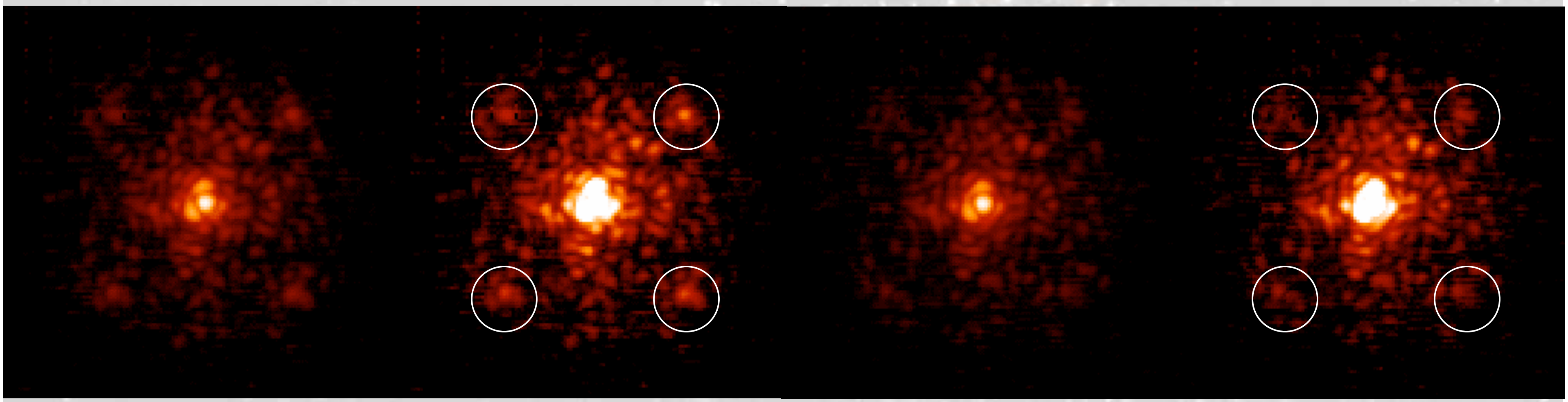


Least-squares reconstructor

$$R = (H^T H + \alpha C_\phi^{-1})^{-1} H^T$$



Inverse covariance matrix for Kolmogorov turbulence



SVD

Bayesian

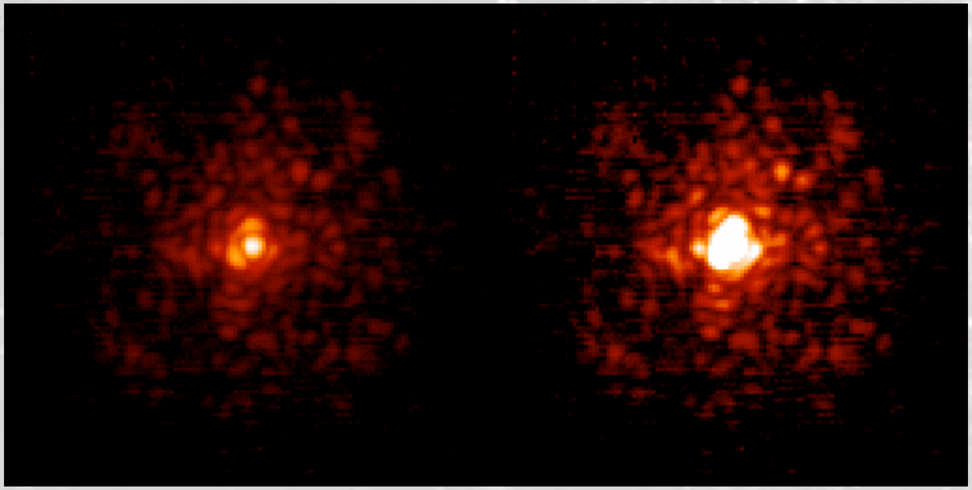
Least-squares reconstructor

$$R = (H^T H + \alpha C_\phi^{-1})^{-1} H^T$$

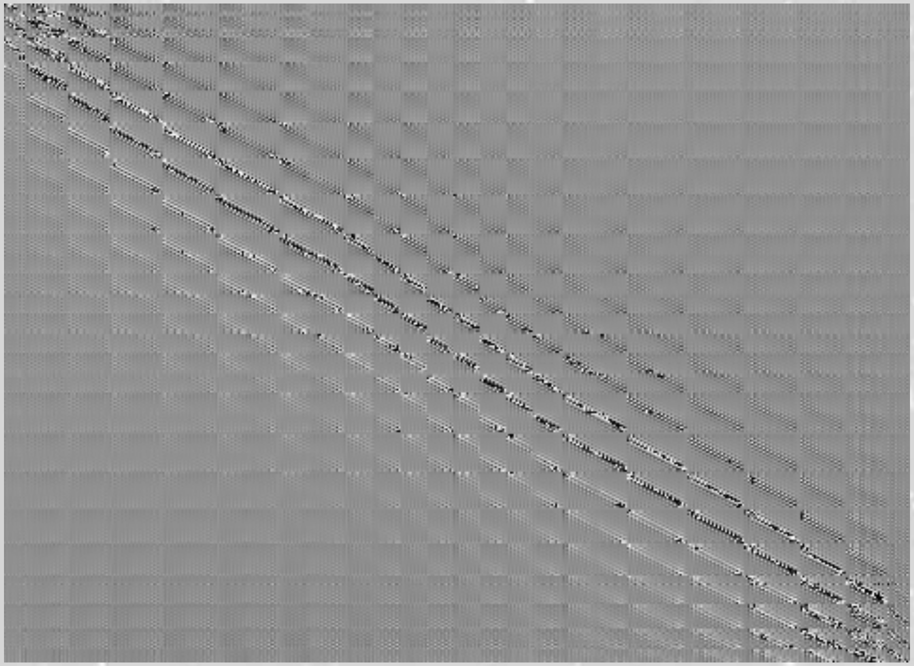
Noise-to-signal parameter

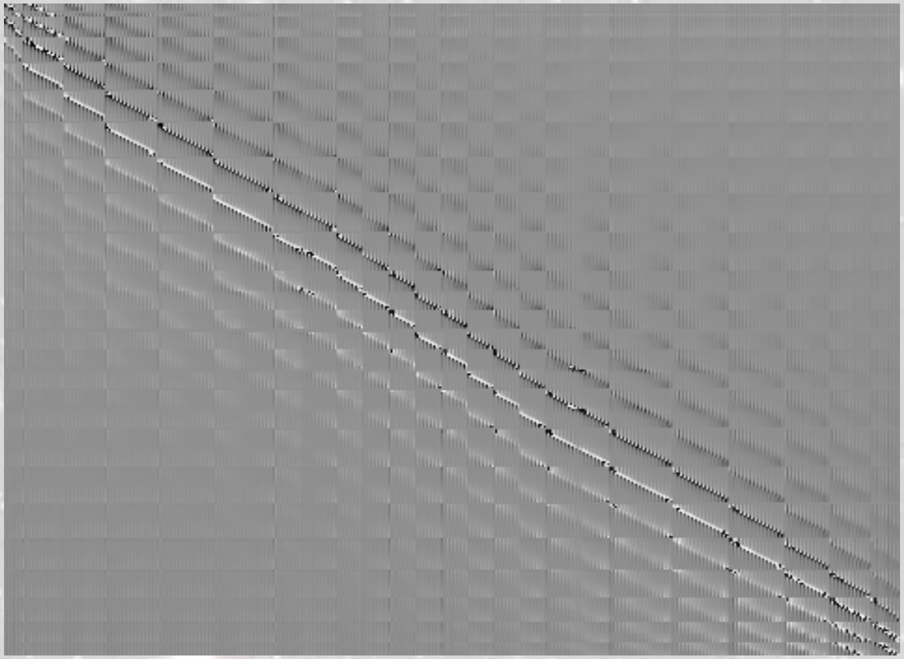
- Minimizes $(s - Ha)^2 + \alpha a^T C_\phi^{-1} a$

Bright star α=1 Faint star α=30



SVD vs Bayesian

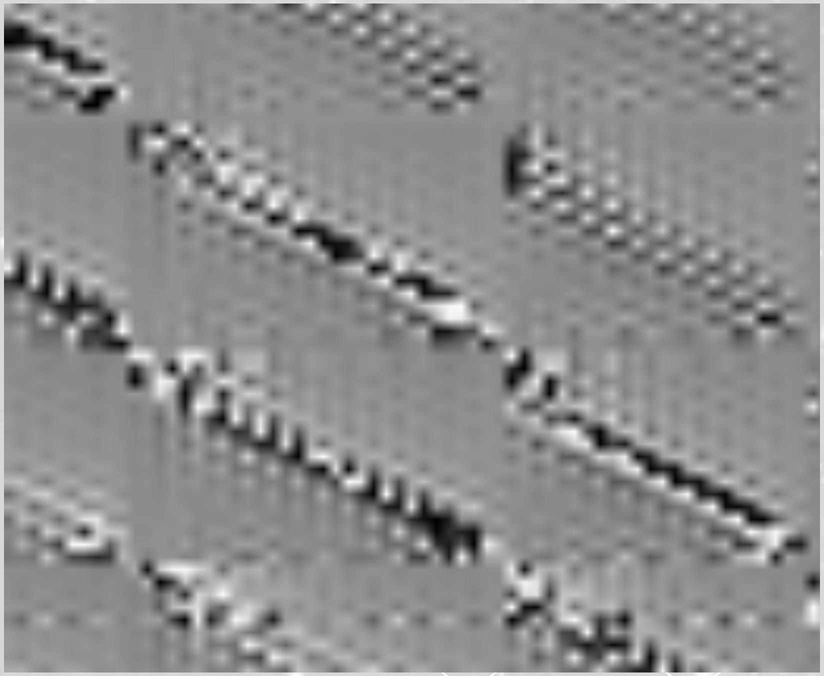




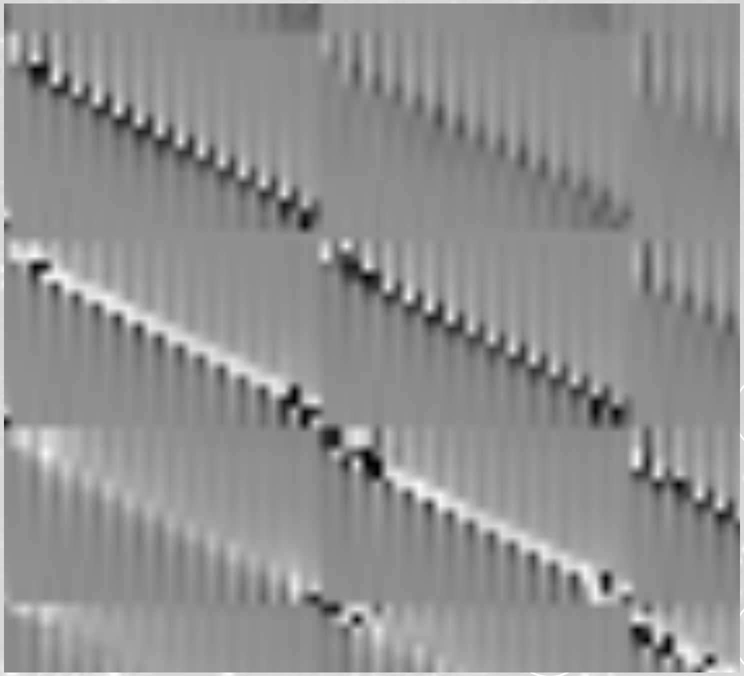
SVD Bayesian

10 Noise propagation 1

SVD vs Bayesian



SVD

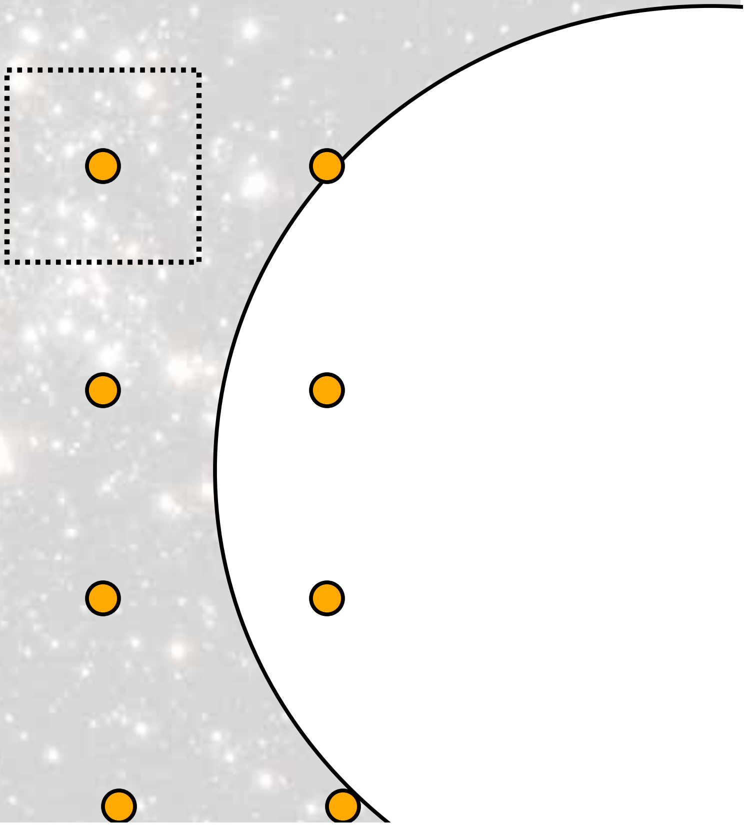


Bayesian

Slaved actuators

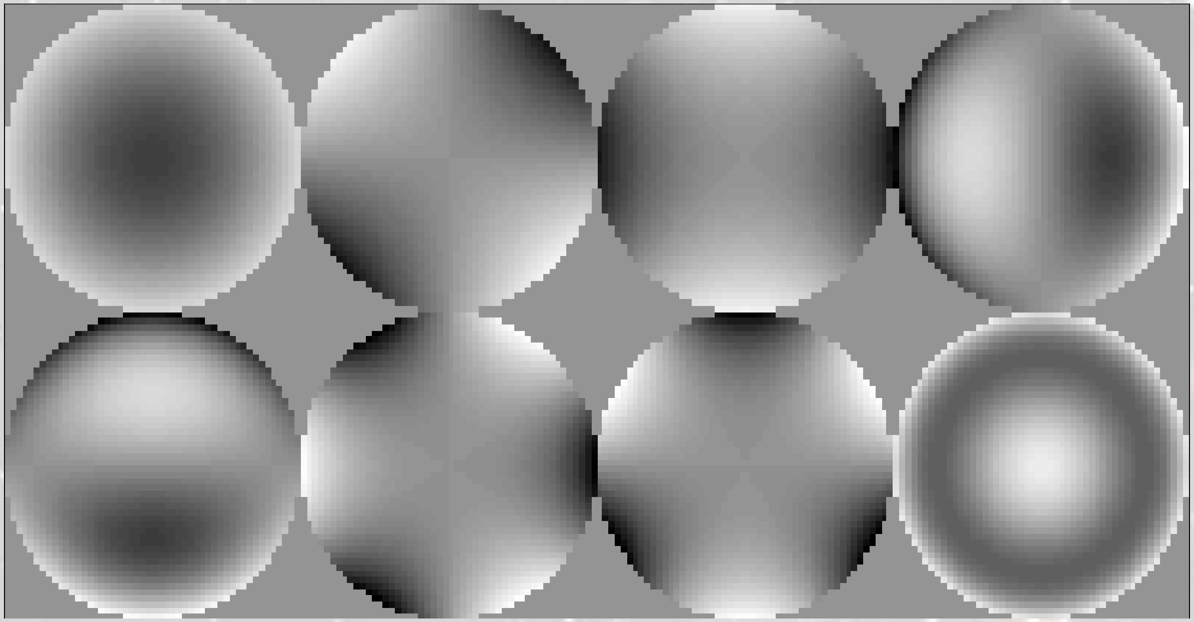
- Some actuators are located outside the pupil and do not directly affect the wavefront
- They are often “slaved” to the average value of its neighbors

Slaved to average value of its neighbors



Modal reconstructors

- Can choose to only reconstruct certain modes
- Avoids reconstructing unwanted modes (e.g., waffle)



Zernike modes

Modal reconstructors

$Z = [z_1, z_2, z_{3,} ...]$ Zernike modes

HZ Centroids measured by applying Zernike modes to the DM

$R = Z[(HZ)^T (HZ)]^{-1} (HZ)^T$ Zernike reconstructor

Fourier transform reconstructor

- The slope measurements are derivatives of the phase, φ

$$s_x[m,n] = \phi[m+1,n] - \phi[m,n] \qquad s_y[m,n] = \phi[m,n+1] - \phi[m,n]$$

- We can take the Fourier transform of both equations

Fourier transform reconstructor

$$s_x[m,n] = \phi[m+1,n] - \phi[m,n]$$

Fourier transform

$$S_x[k,l] = \Phi[k,l]\exp[j2\pi k/N] - \Phi[k,l]$$

Fourier transform reconstructor

$$s_y[m,n] = \phi[m,n+1] - \phi[m,n]$$

Fourier transform

$$S_y[k,l] = \Phi[k,l]\exp[j2\pi l/N] - \Phi[k,l]$$

Fourier transform reconstructor

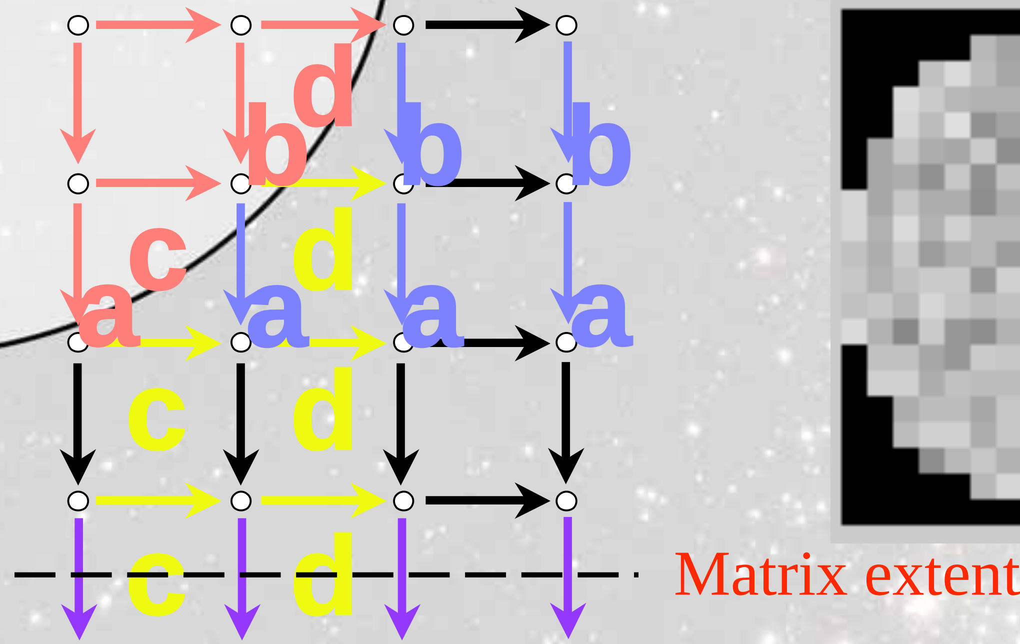
- Solve for $\Phi[k,l]$

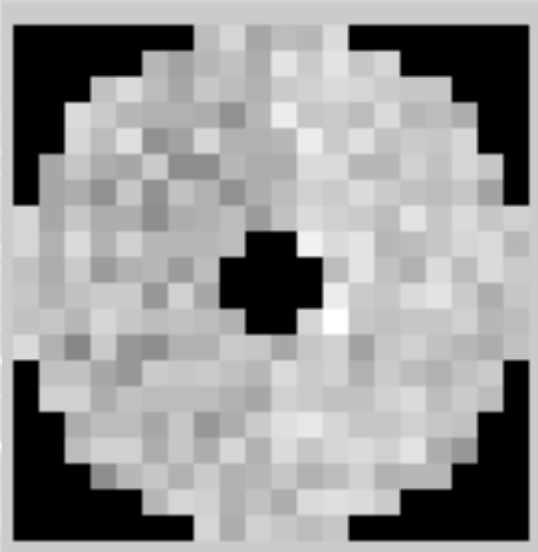
$$\hat{\Phi} = \frac{(\exp[-j2\pi k/N]-1)S_x[k,l] + (\exp[-j2\pi l/N]-1)S_y[k,l]}{4(\sin^2[\pi k/N] + \sin^2[\pi l/N])}$$

- We can now apply any filter in the Fourier domain: e.g., we can low pass filter the signal to remove high spatial frequencies.

- Take the inverse Fourier transform to get the phase

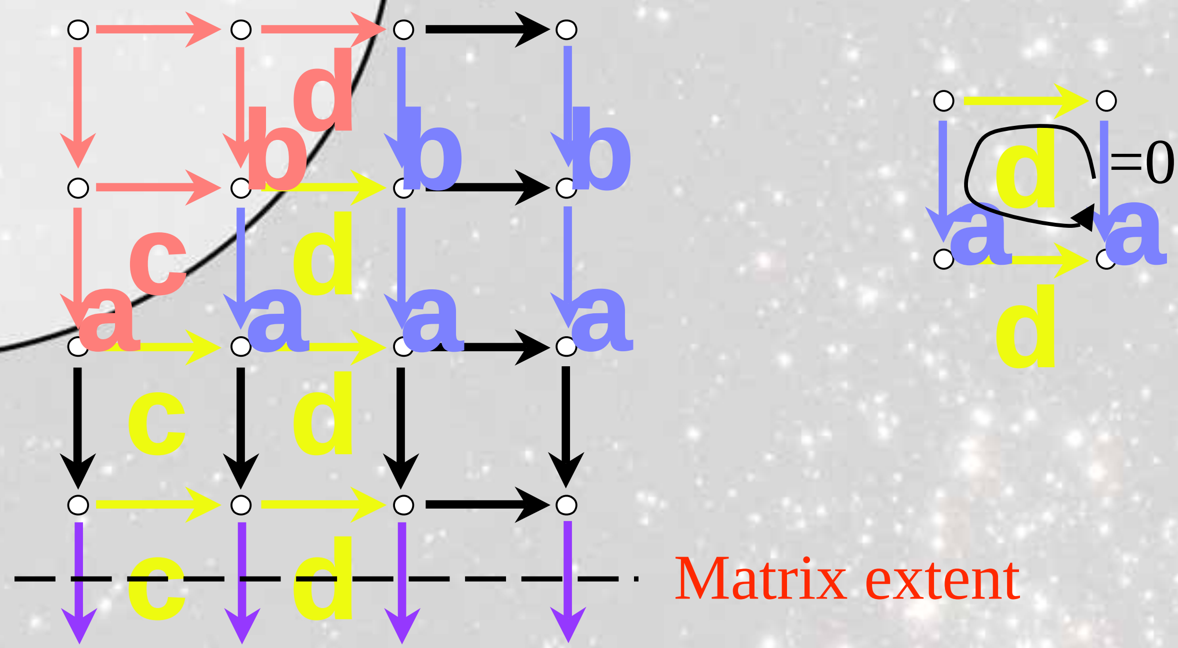
Boundary problem





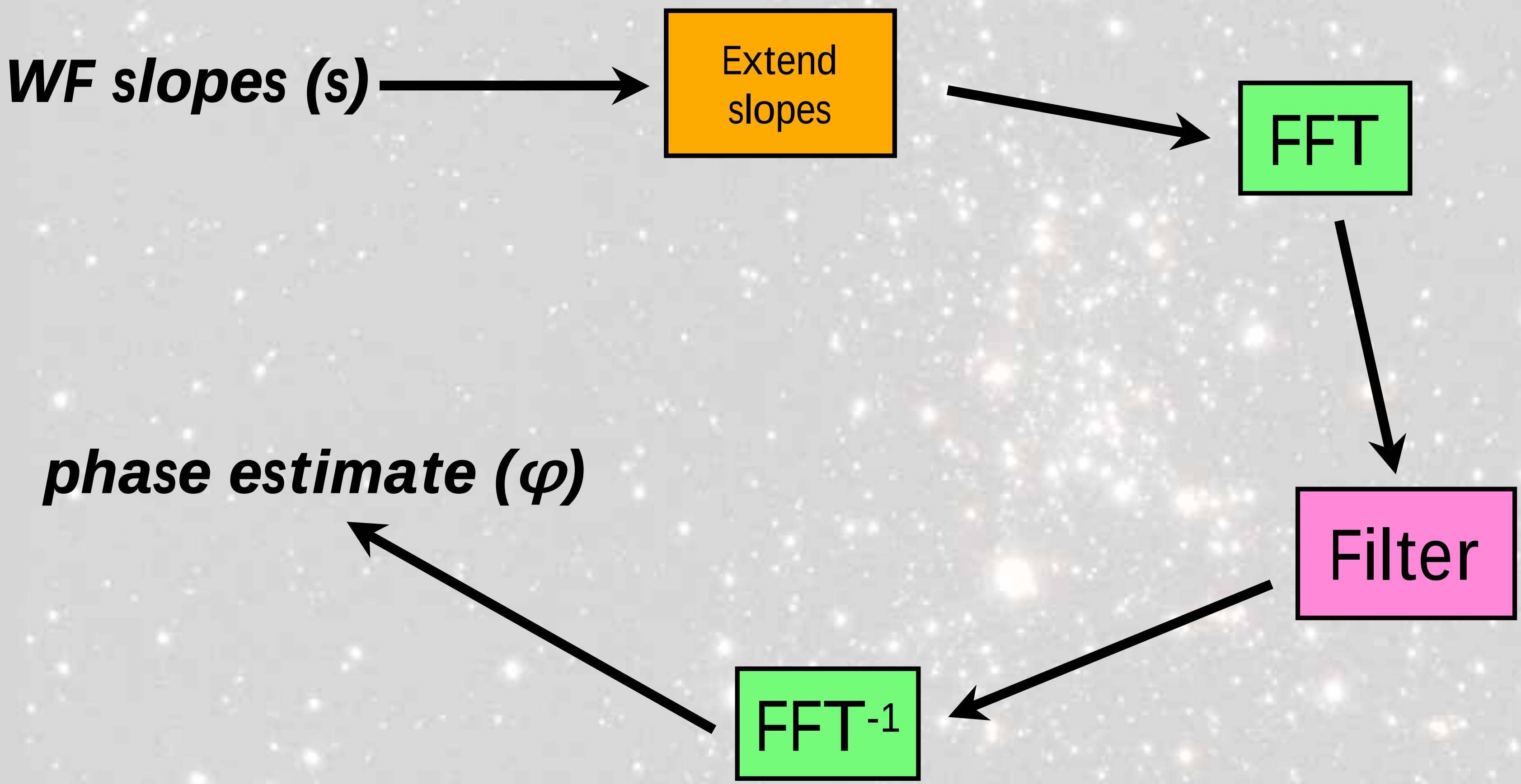
- Finite sized aperture creates a boundary problem leading to incorrect wavefront estimation

Boundary problem



- Solve by managing slopes outside the aperture
- Extend slopes in orthogonal directions outside the aperture
- Set wrap-around slopes to enforce periodicity

Summary of FFT Reconstructor



Control laws

- Now that we have the reconstructed wavefront, *a*, what do we do?

$u[n] = a$ Wavefront error at time n

$y[n]$ Mirror position at time n

- Simplest control law is integrator with variable loop gain, *k*

$$\underbrace{y[n]}_{\text{New mirror command}} = \underbrace{y[n-1]}_{\text{Current mirror command}} + k\underbrace{u[n]}_{\text{Reconstructed wavefront}}$$

New mirror command Current mirror command Reconstructed wavefront

Control laws

- Need to clip the voltages to the maximum voltage, Vmax

$$y[n] = \min(y[n], V\max)$$

- Actuator clipping and DM hysteresis can introduce invisible modes. These can be removed using a "leaky" integrator

$$y[n] = 0.99\,y[n-1] + ku[n]$$

$$y[n] = \min(y[n], V\max)$$

Control laws

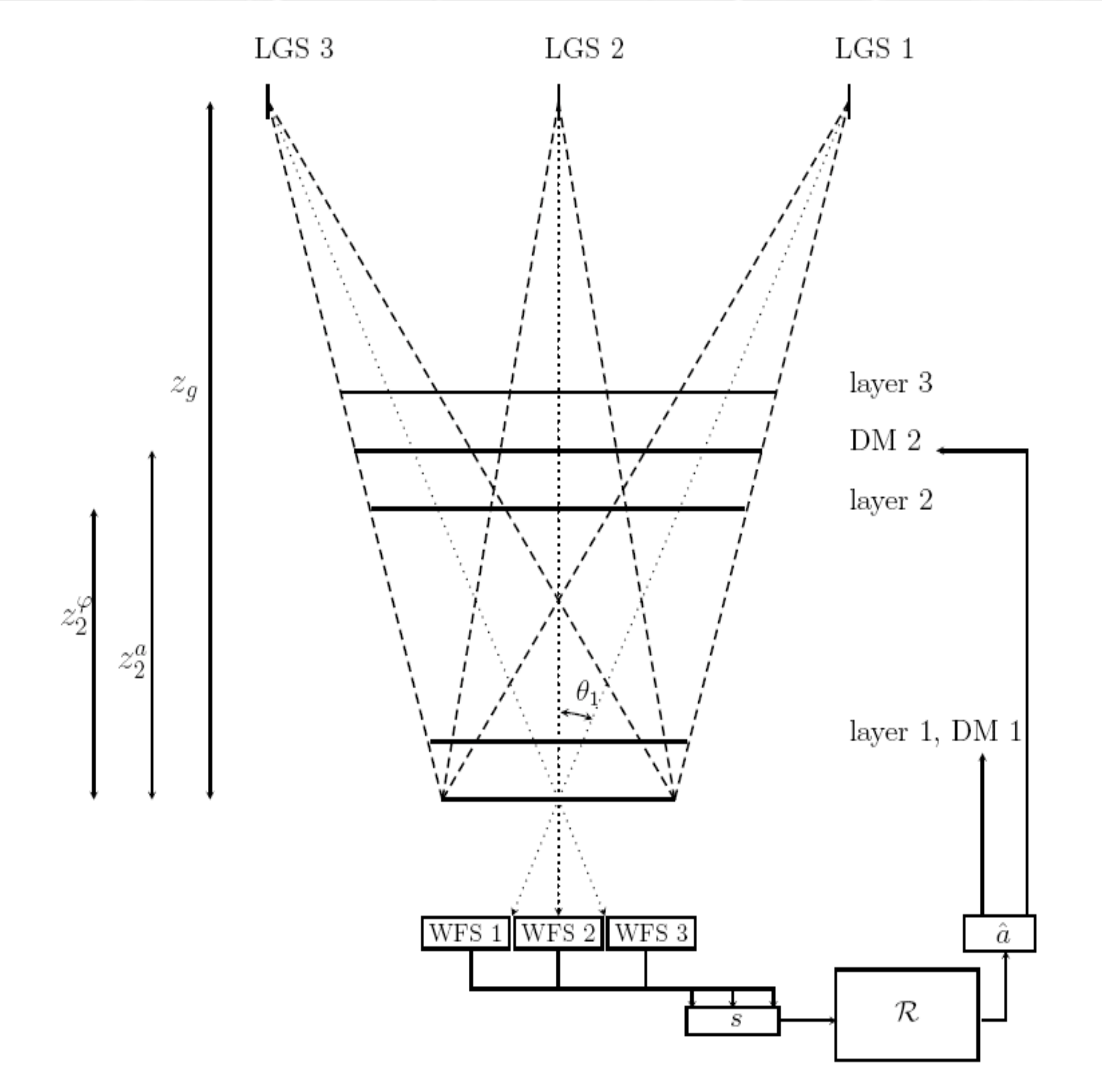
- Controllers can preempt the response

$$y[n] = y[n-1] + ku[n] + \underbrace{c(u[n] - u[n-1])}_{\text{Add a fraction of the way the error is changing}}$$

- Or to allow for computational delay (Smith compensator)

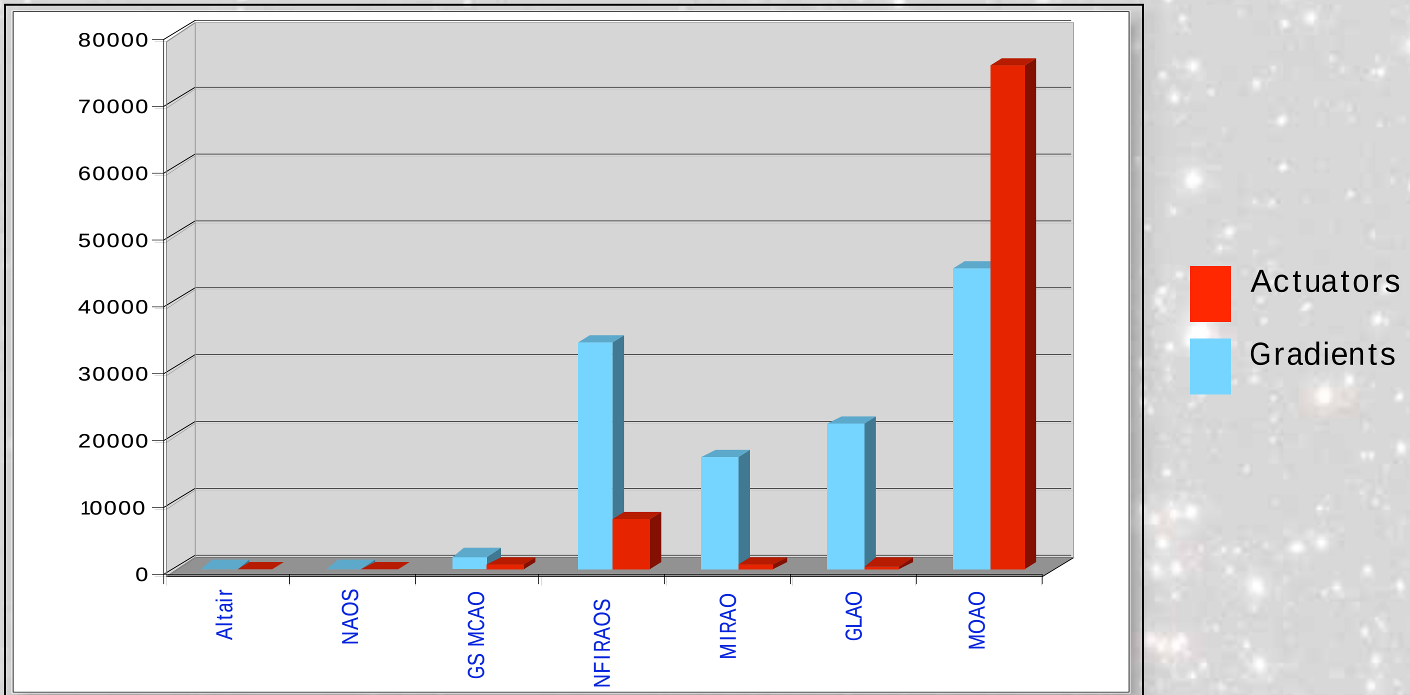
$$y[n] = y[n-1] - \underbrace{c(y[n-1] - y[n-2])}_{\text{Remove portion of the way the DM is moving}} + ku[n]$$

Multi-conjugate adaptive optics



System complexity increases

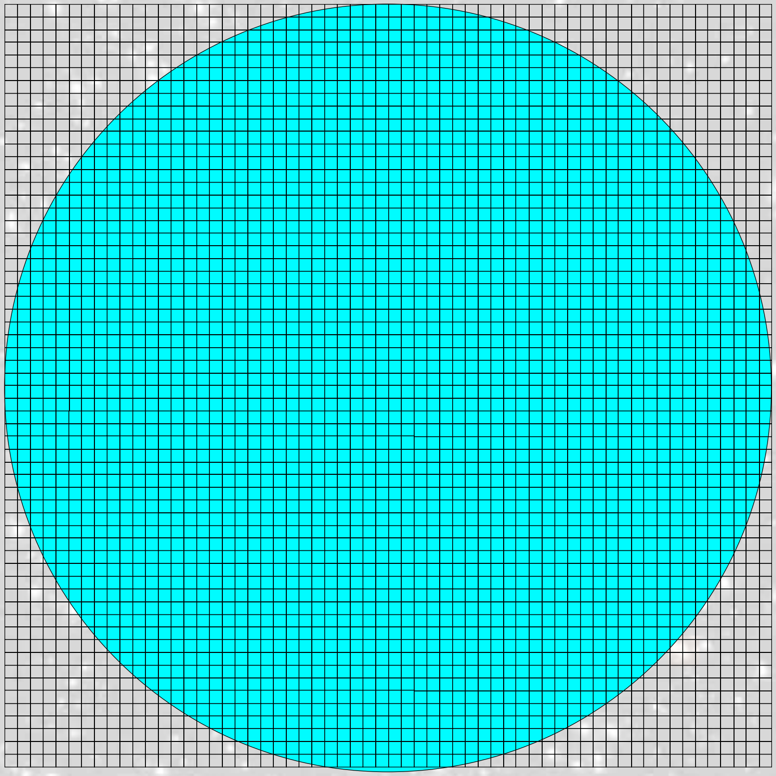
- Next generation telescopes and AO systems will have much greater computational demands than current systems



TMT systems

Computationally efficient reconstructors

- For example, for the Thirty Meter Telescope (TMT) AO system NFIRAOS
  - DM with 60x60 actuators
  - 6 LGS with 60x60 subapertures per WFS
  - 800 Hz sampling rate for LGS



TMT reconstructor example

- 3600 actuators and 43200 slope measurements
- *H* is 43200 x 3600 element matrix

- Offline calculation (not in real time but as pupil rotates, etc):

$$R = (H^T H + \alpha C_\phi^{-1})^{-1} H^T$$

- Inverse is O(a^3) ≈ 5x10^{10} floating point operations (flops)
- 1000 times more flops than Keck!

TMT reconstructor example

- Online calculation (at 800 Hz) is

$$a = Rs$$

- $\approx 2 \times 10^8$ flops at 800 Hz $\approx 1 \times 10^{11}$ flops/second
- 2000 times more than Keck

- Online and offline calculations for TMT are not feasible using current algorithms and hardware

Now for the good stuff!

Fast reconstructors

Bayesian reconstructor: $$R = (H^T H + \alpha C_\phi^{-1})^{-1} H^T$$

Slopes to actuators: $$a = (H^T H + \alpha C_\phi^{-1})^{-1} H^T s$$

Substitute and rearrange: $$(H^T H + \alpha C_\phi^{-1}) a = H^T s$$

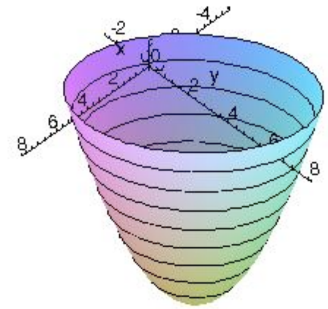
Problem formulation

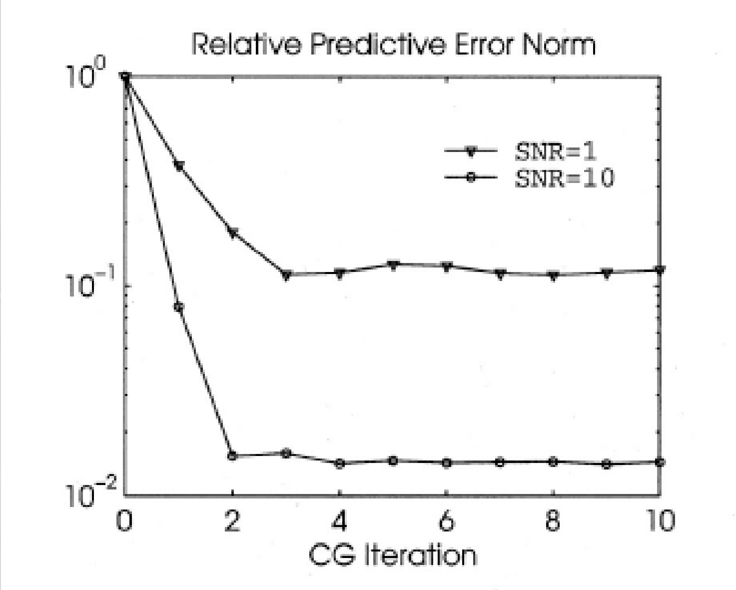
Problem formulation: $$\underbrace{(H^T H + \alpha C_\phi^{-1})}_{P} a = \underbrace{H^T s}_{y}$$

We want to solve the linear system: $$Pa = y$$

Conjugate gradient algorithm

- Solve $Pa=y$ by minimizing $|Pa-y|^2$.
- Conjugate gradient is an iterative method
- Get closer to solution with more iterations
- Trade-off speed vs accuracy
- Convergence depends on condition number of P





Preconditioning

- Preconditioning matrix M applied to transform problem to speed up convergence

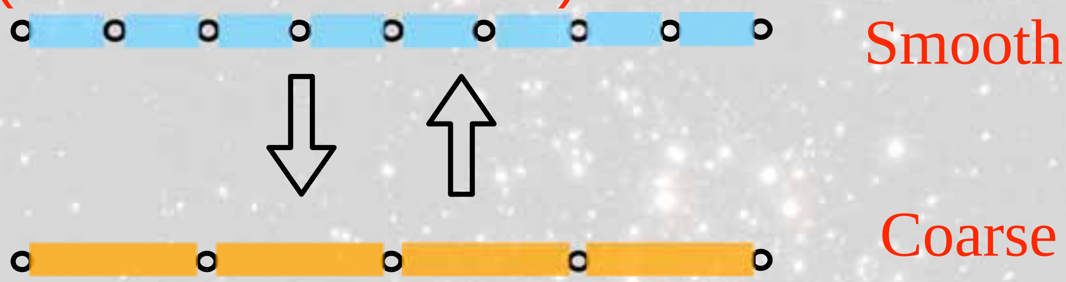
$$Pa = y$$

$$\underbrace{M^{-1}PM^{-1}}_{P'}\underbrace{Ma}_{a'} = \underbrace{M^{-1}y}_{y'}$$

- Apply conjugate gradient algorithm to transformed problem
- Choice of preconditioner M is critical to speed and accuracy
- Convergence now dependent on condition number of P' not P

Multi-grid preconditioned CG

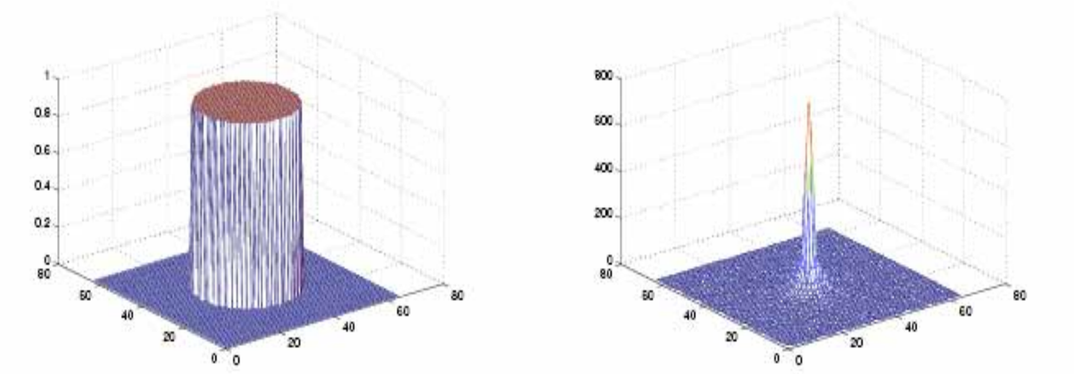
- This a multiple resolution solution
- Example: consider the problem Pa=y in 2 different grid sizes (coarse and smooth)



  - Perform CG method with fast convergence for high frequencies to the smooth grid
  - Project the low frequency error onto the coarser grid
  - Perform CG method on the coarse grid
  - Project the coarse grid solution back to the smooth grid
  - Perform CG method on the smooth grid using high frequencies from the smooth grid solution and low frequencies from the coarse grid solution

Fourier domain preconditioned CG

- Key idea is to transform P to Fourier domain where F is Fourier transform matrix $P' = F^{-1}PF$



Pupil Mask Fourier transform of pupil mask

- In spatial domain, pupil mask matrix is almost full
- In Fourier domain, truncate the FT of the pupil mask to only a few pixels
- This matrix is now sparse so far fewer matrix multiplies

Complexity of methods

- Approximate number of operations to solve: $a = Rs$ where there are n actuators

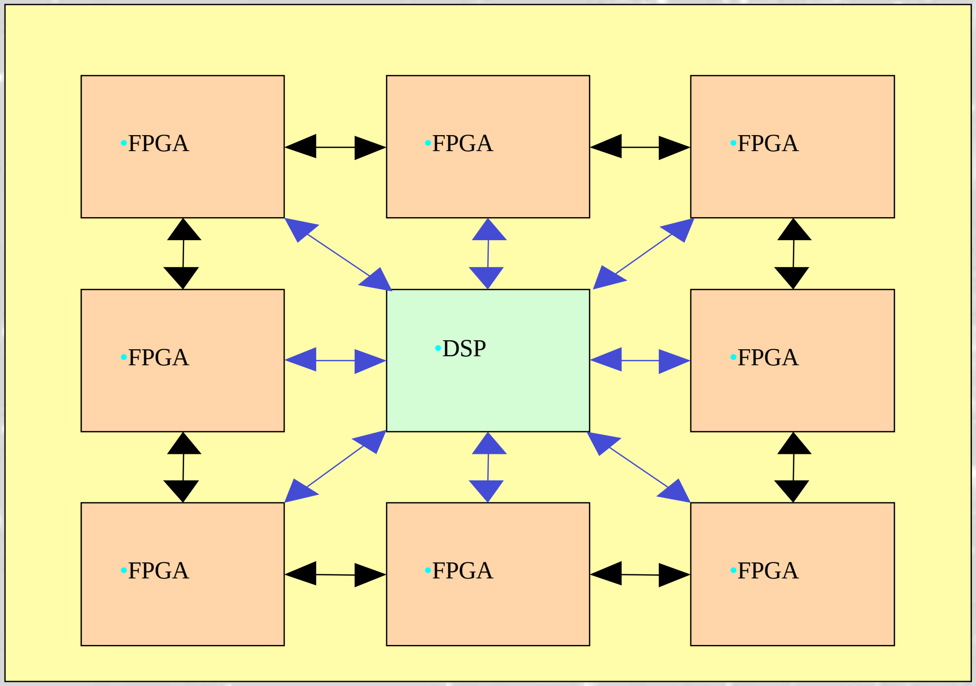
Method	Operations
Direct solve	$O(n^3)$
Fourier transform reconstructor	O(n log n)
Multi-grid preconditioned CG	O(n log n)
Fourier domain preconditioned CG	O(n log n)

Hardware approaches

- Lick uses 2 Pentium processors for online and offline computation
- Need to use more processors and be able to split the problem into parallel blocks
- DSP – Digital Signal Processor (a fast mathematical processor)
- FPGA – Field Programmable Gate Array (lots and lots of logic gates)

Hardware approach for TMT

- Proposed TMT hardware solution is to use combination of FPGAs and DSPs
- DSP does pixel processing (centroiding etc)
- FPGAs do tomography and fitting steps



Mahalo!