

# High-Contrast Imaging Via Fast-Fourier Optimization

Robert J. Vanderbei

July 9, 2012

SIAM Annual Meeting 2012  
Minneapolis MN

<http://www.princeton.edu/~rvdb>

# The Plan...

Brief Synopsis of Pupil-Mask Coronagraphy

Fourier Transforms in 2D — Brute Force vs. Smart Approach

Applying it in an Optimization Context

Some New Masks for High-Contrast Imaging

Relation to the FFT

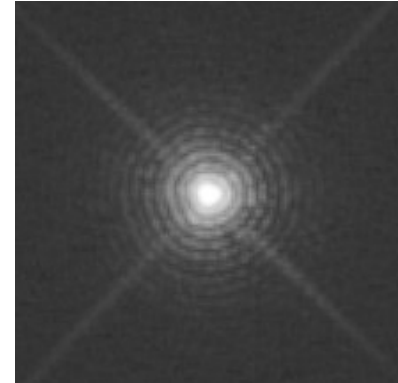
# Motivating Problem

## High-Contrast Imaging for Planet-Finding

Build a telescope capable of finding Earth-like planets around nearby Sun-like stars.

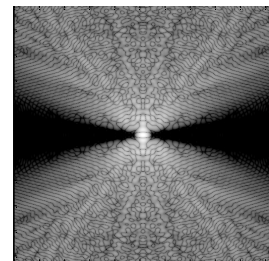
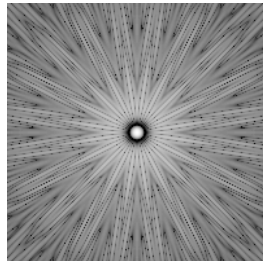
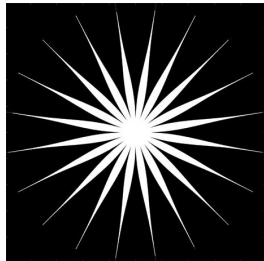
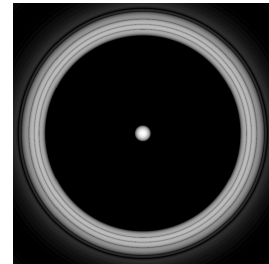
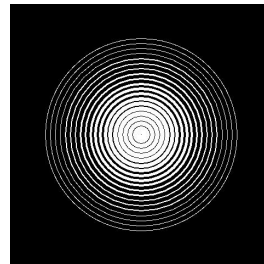
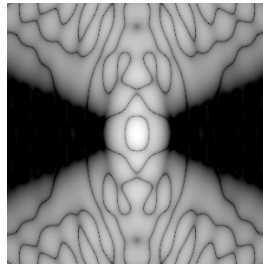
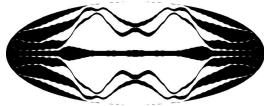
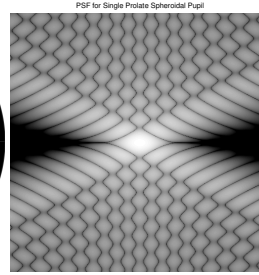
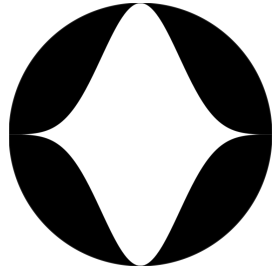
Problem is hard:

- Star is  $10^{10}$  times brighter than the planet.
- Angular separation is small  $\approx 0.01$  arcseconds.
- Light is a wave: the star is not a pinpoint of light—it has a *diffraction pattern*.
- Light is photons  $\Rightarrow$  Poisson statistics.



The diffraction pattern is the magnitude-squared of the *Fourier transform* of the telescope's *entrance pupil*.

# Potpourri of Pupil Masks



# Pupil Apodization

Let  $f(x, y)$  denote the transmissivity (i.e., *apodization*) at location  $(x, y)$  on the surface of a filter placed in the pupil of a telescope.

The *electromagnetic field* in the image plane of such a telescope associated with an on-axis point source (i.e., a star) is proportional to the Fourier transform of the apodization  $f$ .

Assuming that the telescope's opening has a diameter of one, the Fourier transform can be written as

$$\hat{f}(\xi, \eta) = \int_{-1/2}^{1/2} \int_{-1/2}^{1/2} e^{2\pi i(x\xi + y\eta)} f(x, y) dx dy.$$

The *intensity* of the light in the image is proportional to the *magnitude squared* of  $\hat{f}$ .

Assuming that the underlying telescope has a circular opening of diameter one, we impose the following constraint on  $f$ :

$$f(x, y) = 0 \quad \text{for} \quad x^2 + y^2 > (1/2)^2.$$

# Optimized Apodizations

Maximize *light throughput* subject to constraint that almost no light reaches a given *dark zone*  $\mathcal{D}$  and other *structural* constraints:

$$\begin{aligned} \text{maximize} \quad & \iint f(x, y) dx dy \quad \left( = \widehat{f}(0, 0) \right) \\ \text{subject to} \quad & \left| \widehat{f}(\xi, \eta) \right| \leq \varepsilon \widehat{f}(0, 0), \quad (\xi, \eta) \in \mathcal{D}, \\ & f(x, y) = 0, \quad x^2 + y^2 > (1/2)^2, \\ & 0 \leq f(x, y) \leq 1, \quad \text{for all } x, y. \end{aligned}$$

Here,  $\varepsilon$  is a small positive constant.

In general, the Fourier transform  $\widehat{f}$  is complex valued.

This optimization problem has a linear objective function and both linear constraints and second-order cone constraints.

Hence, a discretized version can be solved (to a global optimum).

# Exploiting Symmetry

Assuming that the filter can be symmetric with respect to reflection about both axes (note: sometimes not possible), the Fourier transform can be written as

$$\hat{f}(\xi, \eta) = 4 \int_0^{1/2} \int_0^{1/2} \cos(2\pi x\xi) \cos(2\pi y\eta) f(x, y) dx dy.$$

In this case, the Fourier transform is real and so the second-order cone constraints can be replaced with a pair of inequalities,

$$-\varepsilon \hat{f}(0, 0) \leq \hat{f}(\xi, \eta) \leq \varepsilon \hat{f}(0, 0),$$

making the problem an *infinite dimensional linear programming problem*.

# Discretization

Consider a two-dimensional Fourier transform

$$\widehat{f}(\xi, \eta) = 4 \int_0^{1/2} \int_0^{1/2} \cos(2\pi x\xi) \cos(2\pi y\eta) f(x, y) dx dy.$$

Its discrete approximation can be computed as

$$\widehat{f}_{j_1, j_2} = 4 \sum_{k_2=1}^n \sum_{k_1=1}^n \cos(2\pi x_{k_1} \xi_{j_1}) \cos(2\pi y_{k_2} \eta_{j_2}) f_{k_1, k_2} \Delta x \Delta y, \quad 1 \leq j_1, j_2 \leq m,$$

where

$$\begin{aligned} x_k &= (k - 1/2)\Delta x, & 1 \leq k \leq n, \\ y_k &= (k - 1/2)\Delta y, & 1 \leq k \leq n, \\ \xi_j &= (j - 1/2)\Delta \xi, & 1 \leq j \leq m, \\ \eta_j &= (j - 1/2)\Delta \eta, & 1 \leq j \leq m, \\ f_{k_1, k_2} &= f(x_{k_1}, y_{k_2}), & 1 \leq k_1, k_2 \leq n \\ \widehat{f}_{j_1, j_2} &\approx \widehat{f}(\xi_{j_1}, \eta_{j_2}), & 1 \leq j_1, j_2 \leq m. \end{aligned}$$

Complexity:  $m^2 n^2$ .

## A Clever (and Trivial!) Idea

The obvious brute force calculation requires  $m^2n^2$  operations.

However, we can “factor” the double sum into a nested pair of sums.

Introducing new variables that represent the inner sum, we get:

$$g_{j_1, k_2} = 2 \sum_{k_1=1}^n \cos(2\pi x_{k_1} \xi_{j_1}) f_{k_1, k_2} \Delta x, \quad 1 \leq j_1 \leq m, \quad 1 \leq k_2 \leq n,$$

$$\hat{f}_{j_1, j_2} = 2 \sum_{k_2=1}^n \cos(2\pi y_{k_2} \eta_{j_2}) g_{j_1, k_2} \Delta y, \quad 1 \leq j_1, j_2 \leq m,$$

Formulated this way, the calculation requires only  $mn^2 + m^2n$  operations.

# Brute Force vs Clever Approach

On the following page we show two AMPL model formulations of this problem.

On the left is the version expressed in the straightforward one-step manner.

On the right is the AMPL model for the same problem but with the Fourier transform expressed as a pair of transforms—the so-called *two-step process*.

The dark zone  $\mathcal{D}$  is a pair of sectors of an annulus with inner radius 4 and outer radius 20.

Except for the resolution, the two models produce the same result.

# Two AMPL Models

```
param rho0 := 4;          param rho1 := 20;
param m := 35;          # discretization parameter
param n := 150;        # discretization parameter
param dx := 1/(2*n);   param dy := dx;

set Xs := setof {j in 0.5..n-0.5 by 1} j/(2*n);
set Ys := Xs;
set Pupil :=
    setof {x in Xs, y in Ys: x^2+y^2<0.25} (x,y);
set Xis := setof {j in 0..m} j*rho1/m;
set Etas := Xis;
set DarkHole := setof {xi in Xis, eta in Etas:
    xi^2+eta^2>=rho0^2 &&
    xi^2+eta^2<=rho1^2 &&
    eta <= xi } (xi,eta);

var f {(x,y) in Pupil} >= 0, <= 1;

var fhat {xi in Xis, eta in Etas};

maximize area: sum {(x,y) in Pupil} f[x,y]*dx*dy;

subject to fhat_def {xi in Xis, eta in Etas}:
    fhat[xi,eta] = 4*sum {(x,y) in Pupil}
        f[x,y]*cos(2*pi*x*xi)
            *cos(2*pi*y*eta)*dx*dy;
subject to sidelobe_pos {(xi,eta) in DarkHole}:
    fhat[xi,eta] <= 10^(-5)*fhat[0,0];
subject to sidelobe_neg {(xi,eta) in DarkHole}:
    -10^(-5)*fhat[0,0] <= fhat[xi,eta];

solve;
```

```
param rho0 := 4;          param rho1 := 20;
param m := 35;          # discretization parameter
param n := 1000;       # discretization parameter
param dx := 1/(2*n);   param dy := dx;

set Xs := setof {j in 0.5..n-0.5 by 1} j/(2*n);
set Ys := Xs;
set Pupil :=
    setof {x in Xs, y in Ys: x^2+y^2 < 0.25} (x,y);
set Xis := setof {j in 0..m} j*rho1/m;
set Etas := Xis;
set DarkHole := setof {xi in Xis, eta in Etas:
    xi^2+eta^2>=rho0^2 &&
    xi^2+eta^2<=rho1^2 &&
    eta <= xi } (xi,eta);

var f {(x,y) in Pupil} >= 0, <= 1;
var g {xi in Xis, y in Ys};
var fhat {xi in Xis, eta in Etas};

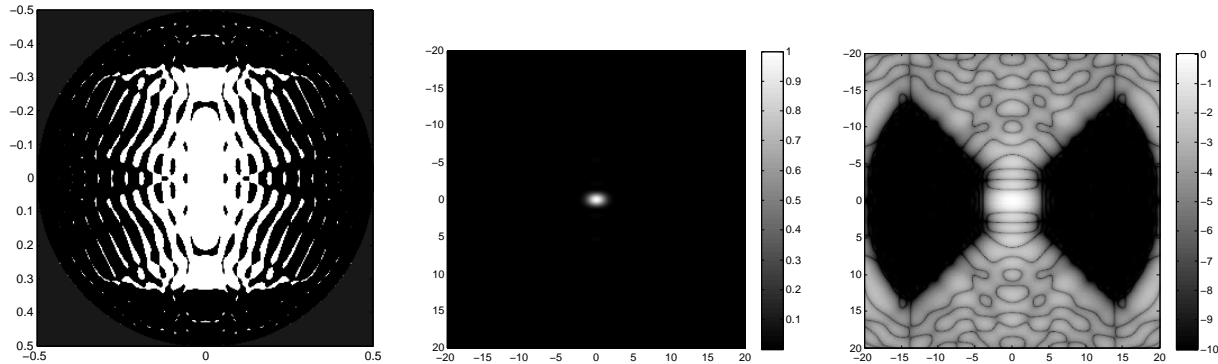
maximize area: sum {(x,y) in Pupil} f[x,y]*dx*dy;

subject to g_def {xi in Xis, y in Ys}:
    g[xi,y] = 2*sum {x in Xs: (x,y) in Pupil}
        f[x,y]*cos(2*pi*x*xi)*dx;
subject to fhat_def {xi in Xis, eta in Etas}:
    fhat[xi,eta] = 2*sum {y in Ys}
        g[xi,y]*cos(2*pi*y*eta)*dy;

subject to sidelobe_pos {(xi,eta) in DarkHole}:
    fhat[xi,eta] <= 10^(-5)*fhat[0,0];
subject to sidelobe_neg {(xi,eta) in DarkHole}:
    -10^(-5)*fhat[0,0] <= fhat[xi,eta];

solve;
```

# Optimal Solution



*Left.* The optimal apodization found by either of the models shown on previous slide.

*Center.* Plot of the star's image (using a linear stretch).

*Right.* Logarithmic plot of the star's image (black =  $10^{-10}$ ).

Notes:

- The “apodization” turns out to be purely opaque and transparent (i.e., a mask).
- The mask has “islands” and therefore must be laid on glass.

# Close Up

Brute force with  $n = 150$



Two-step with  $n = 1000$



# Summary Problem Stats

Comparison between a few sizes of the one-step and two-step models.

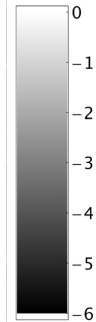
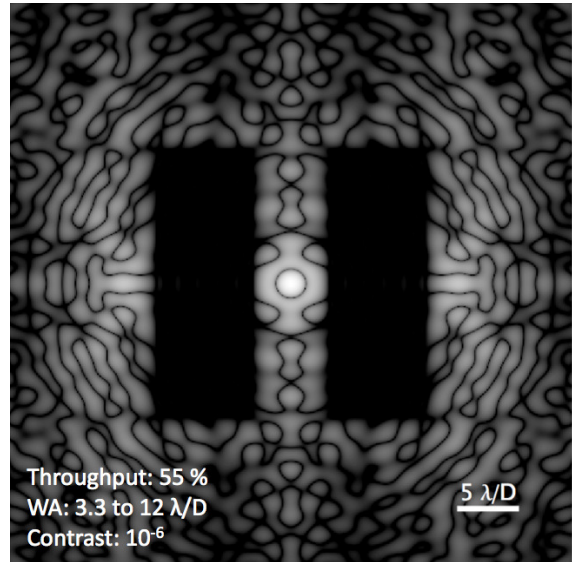
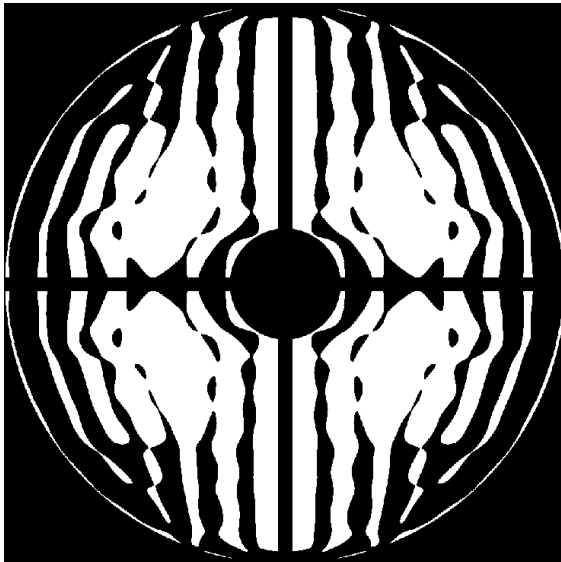
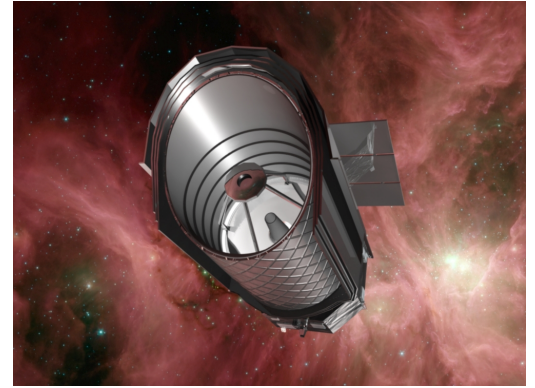
Problem-specific stats.

Model	$n$	$m$	constraints	variables	nonzeros	arith. ops.
One step	150	35	976	17,672	17,247,872	17,196,541,336
One step	250	35	*	*	*	*
Two step	150	35	7,672	24,368	839,240	3,972,909,664
Two step	500	35	20,272	215,660	7,738,352	11,854,305,444
Two step	1000	35	38,272	822,715	29,610,332	23,532,807,719

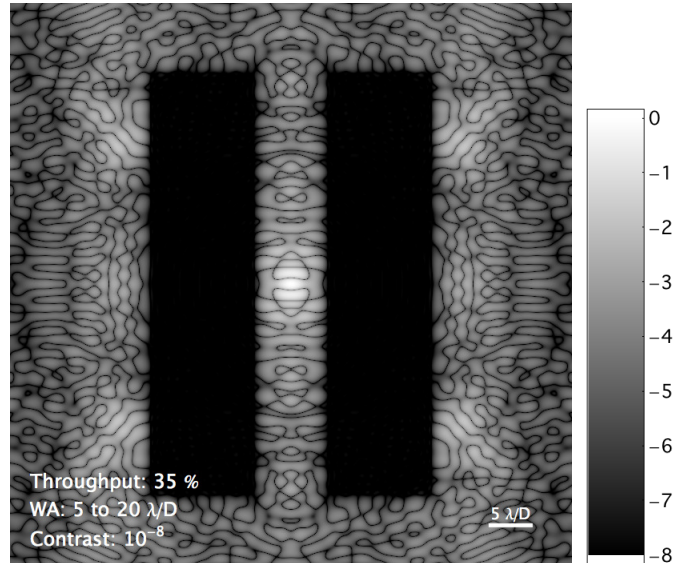
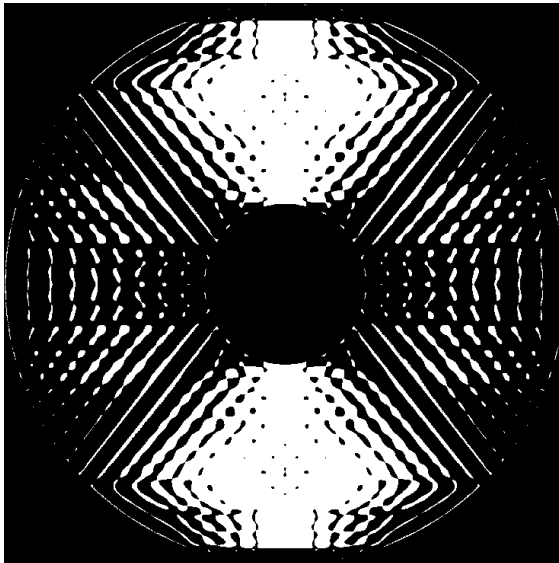
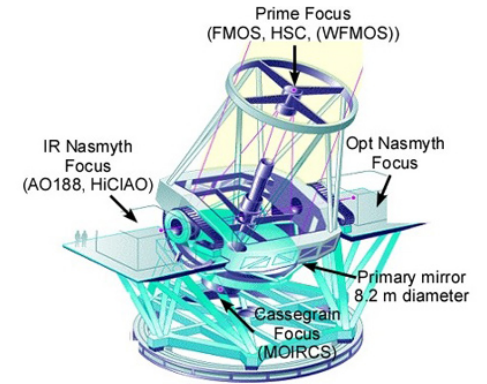
Hardware/Solution-specific performance comparison data.

Model	$n$	$m$	iterations	primal objective	dual objective	cpu time (sec)
One step	150	35	54	0.05374227247	0.05374228041	1380
One step	250	35	*	*	*	*
Two step	150	35	185	0.05374233071	0.05374236091	1064
Two step	500	35	187	0.05395622255	0.05395623990	4922
Two step	1000	35	444	0.05394366337	0.05394369256	26060

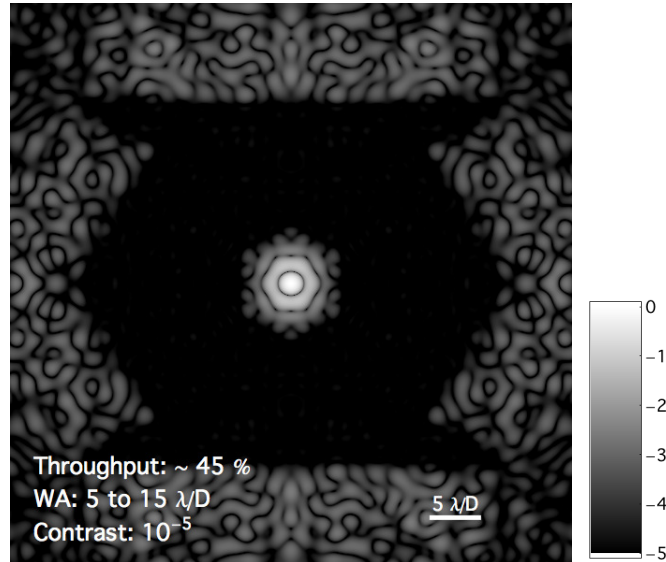
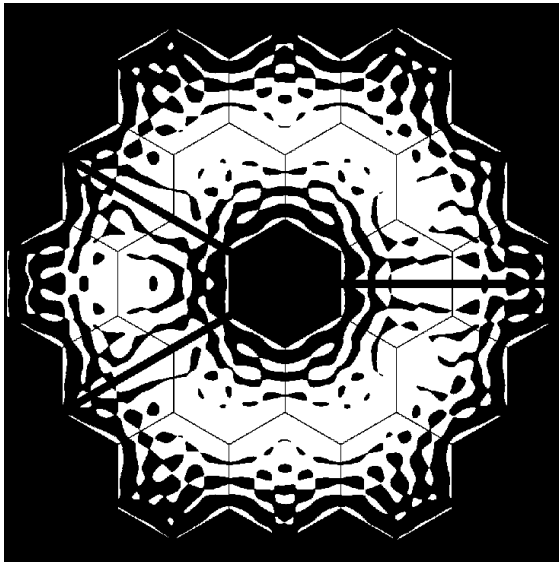
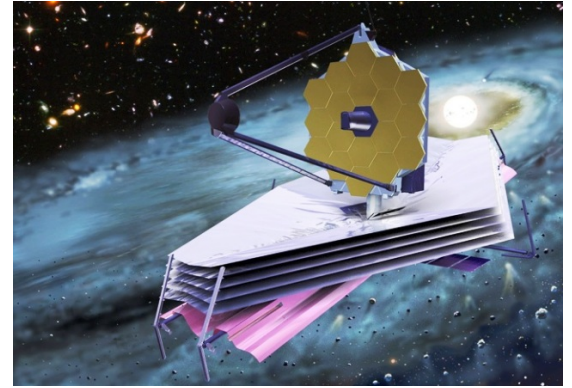
# SPICA



# Subaru



# JWST



# Matrix Notation

Let

$$F := [f_{k_1, k_2}], \quad G := [g_{j_1, k_2}], \quad \widehat{F} := [\widehat{f}_{j_1, j_2}], \quad \text{and} \quad K := [\kappa_{j_1, k_2}],$$

where  $K$  denotes the  $m \times n$  Fourier kernel matrix whose elements are

$$\kappa_{j_1, k_2} = 2 \cos(2\pi x_{k_1} \xi_{j_1}) \Delta x.$$

The two-dimensional Fourier transform  $\widehat{F}$  can be written simply as

$$\widehat{F} = KF K^T$$

and the computation of the transform in two steps is just the statement that the two matrix multiplications can, and should, be done separately:

$$G = KF$$

$$\widehat{F} = GK^T.$$

# MATLAB

Brute force method:

```
for j1 in 1:m
    for j2 in 1:m
        fhat(j1,j2) = 0;
        for k1 in 1:n
            for k2 in 1:n
                fhat(j1,j2) = fhat(j1,j2) + ...
                    4 * cos(2*pi*x(k1)*xi(j1)) * ...
                    cos(2*pi*y(k2)*eta(j2)) * ...
                    f(k1,k2)*dx*dy;
            end
        end
    end
end
```

Two-step method:

```
K = 2 * cos(2*pi*xi*x')*dx;
fhat = K*f*K';
```

# Clever Idea $\Rightarrow$ Matrix Sparsification

## Linear Programming Paradigm

Linear programming software solves problems in this form:

$$\begin{array}{ll} \text{maximize} & c^T x \\ \text{subject to} & Ax = b, \\ & x \geq 0, \end{array}$$

where  $b$  and  $c$  are given vectors and  $A$  is a given matrix.

Of course,  $x$  is a vector.

AMPL converts a problem from its “natural” formulation to this paradigm and then hands it off to a solver.

## Converting to the LP Paradigm

Let  $f_j$ ,  $g_j$ , and  $\hat{f}_j$  denote the column vectors of matrices  $F$ ,  $G$ , and  $\hat{F}$ :

$$F = [f_1 \cdots f_n], \quad G = [g_1 \cdots g_n], \quad \hat{F} = [\hat{f}_1 \cdots \hat{f}_m].$$

We can list the elements of  $F$ ,  $G$  and  $\hat{F}$  in column vectors:

$$\text{vec}(F) = \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix}, \quad \text{vec}(G) = \begin{bmatrix} g_1 \\ \vdots \\ g_n \end{bmatrix}, \quad \text{vec}(\hat{F}) = \begin{bmatrix} \hat{f}_1 \\ \vdots \\ \hat{f}_m \end{bmatrix}.$$

It is straightforward to check that

$$\text{vec}(G) = \begin{bmatrix} K & & \\ & \cdots & \\ & & K \end{bmatrix} \text{vec}(F)$$

and that

$$\text{vec}(\hat{F}) = \begin{bmatrix} \kappa_{1,1}I & \cdots & \kappa_{1,n}I \\ \vdots & & \vdots \\ \kappa_{m,1}I & \cdots & \kappa_{m,n}I \end{bmatrix} \text{vec}(G).$$

# The Constraint Matrix

## One-Step Method

$$Ax = b$$



$$\left[ \begin{array}{ccc|c} \kappa_{1,1}K & \cdots & \kappa_{1,n}K & -I \\ \vdots & & \vdots & \cdots \\ \kappa_{m,1}K & \cdots & \kappa_{m,n}K & -I \\ \hline & \vdots & & \vdots \end{array} \right] \begin{bmatrix} f_1 \\ \vdots \\ f_n \\ \widehat{f_1} \\ \vdots \\ \widehat{f_m} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ \vdots \end{bmatrix}$$

The big left block is a dense  $m^2 \times n^2$  matrix.



# Fast Fourier Transform (FFT)

One-dimensional Fourier transform:

$$\hat{f}(\xi) = \int_{-1/2}^{1/2} e^{2\pi i x \xi} f(x) dx.$$

Discrete approximation:

$$\hat{f}_j = \sum_{k=-n+1}^n e^{2\pi i k \Delta x j \Delta \xi} f_k \Delta x, \quad -m < j \leq m.$$

Let

$$N = 2n \quad \text{and} \quad M = 2m.$$

Suppose that  $N$  and  $M$  can be factored:

$$N = N_0 N_1 \quad \text{and} \quad M = M_0 M_1.$$

Suppose further that the factors are even too:

$$N_0 = 2n_0, \quad N_1 = 2n_1, \quad M_0 = 2m_0, \quad \text{and} \quad M_1 = 2m_1.$$

# It's the same idea!

If we now decompose our sequencing indices  $k$  and  $j$  into

$$k = N_0 k_1 + k_0 \quad \text{and} \quad j = M_0 j_1 + j_0,$$

we get

$$\widehat{f}_{j_0, j_1} = \sum_{k_0=-n_0+1}^{n_0} \sum_{k_1=-n_1+1}^{n_1} e^{2\pi i N_0 k_1 \Delta x M_0 j_1 \Delta \xi} e^{2\pi i N_0 k_1 \Delta x j_0 \Delta \xi} e^{2\pi i k_0 \Delta x (M_0 j_1 + j_0) \Delta \xi} f_{k_0, k_1} \Delta x.$$

We want the first exponential factor to evaluate to one. To make that happen, we assume that  $N_0 M_0 \Delta x \Delta \xi$  is an integer. With that first exponential factor out of the way, we can again write down a two-step algorithm

$$\begin{aligned} g_{j_0, k_0} &= \sum_{k_1=-n_1+1}^{n_1} e^{2\pi i N_0 k_1 \Delta x j_0 \Delta \xi} f_{k_0, k_1} \Delta x, & -m_0 < j_0 \leq m_0, \\ & & -n_0 < k_0 \leq n_0, \\ \widehat{f}_{j_0, j_1} &= \sum_{k_0=-n_0+1}^{n_0} e^{2\pi i k_0 \Delta x (M_0 j_1 + j_0) \Delta \xi} g_{j_0, k_0}, & -m_0 < j_0 \leq m_0 \\ & & -m_1 < j_1 \leq m_1. \end{aligned}$$

# Complexity

The number of multiply/adds required for this two-step algorithm is

$$NM_0 + MN_0 = MN \left( \frac{1}{M_1} + \frac{1}{N_1} \right).$$

If  $M \approx N$  and  $M_1 \approx N_1 \approx \sqrt{N}$ , the complexity simplifies to

$$2N\sqrt{N}.$$

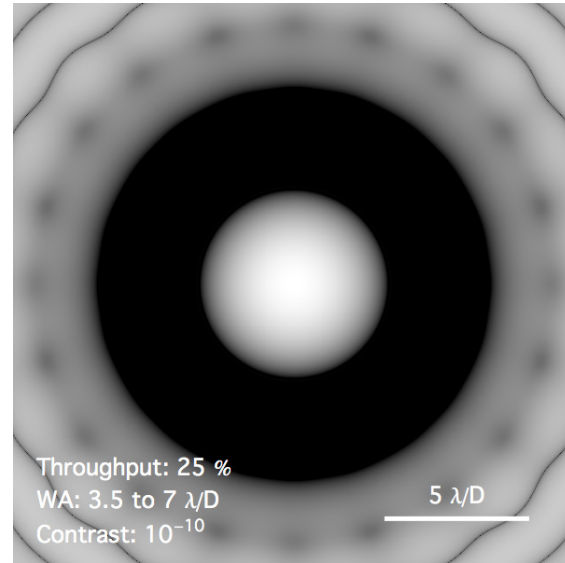
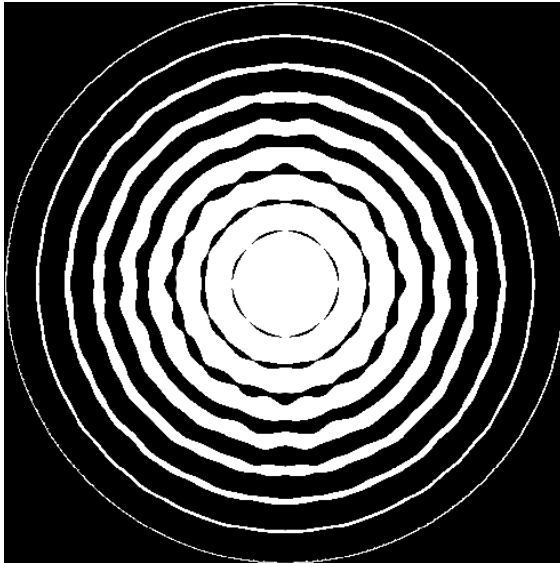
Compared to the one-step algorithm, which takes  $N^2$  multiply/adds, this two-step algorithm gives an improvement of a factor of  $\sqrt{N}/2$ . Also, if  $M$  is much smaller than  $N$ , we get further improvement over the full  $N \times N$  case.

Of course, if  $M_0, M_1, N_0$ , and  $N_1$  can be further factored, then this two-step algorithm can be extended recursively.

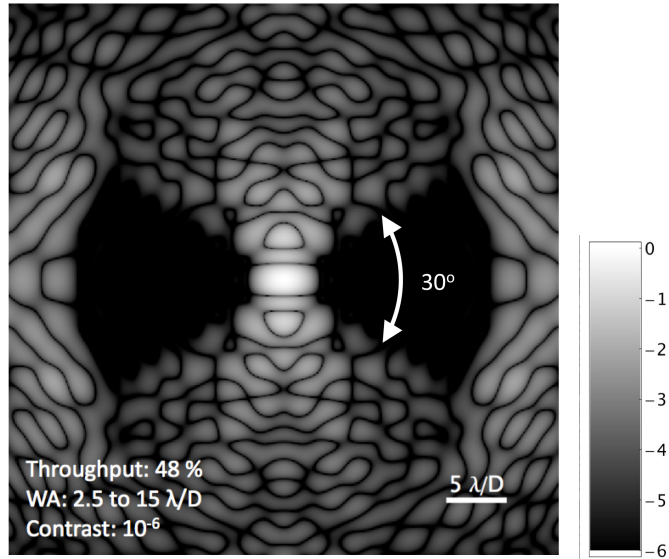
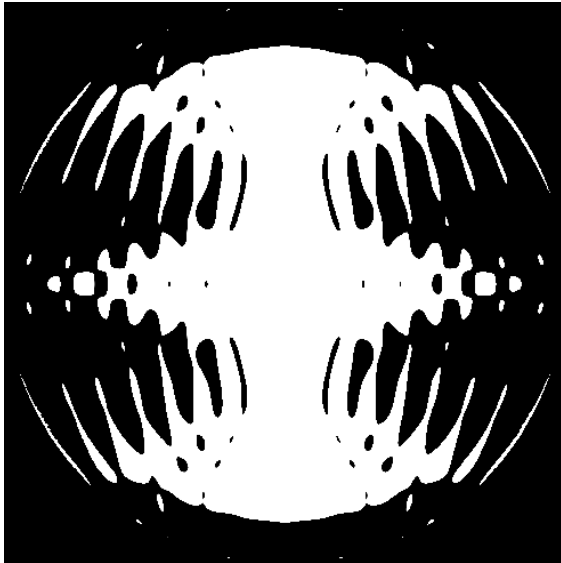
For the *FFT*,  $M$  and  $N$  are chosen to be a power of 3, or more commonly 2. In this case, the recursively applied algorithm is an  $N \log_2 N$  algorithm.



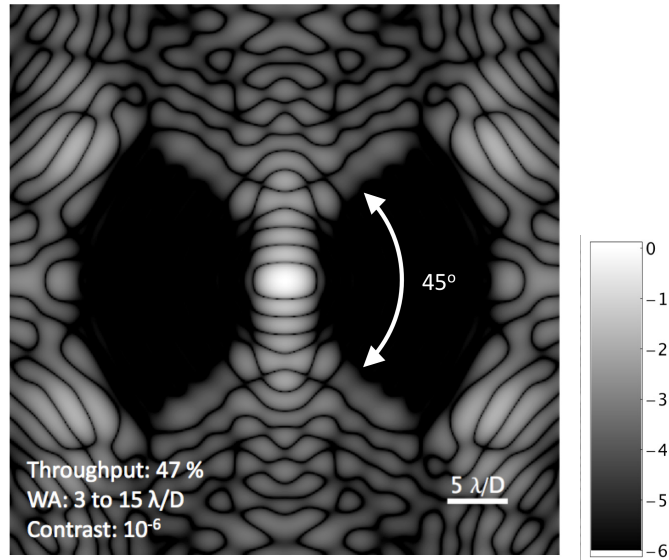
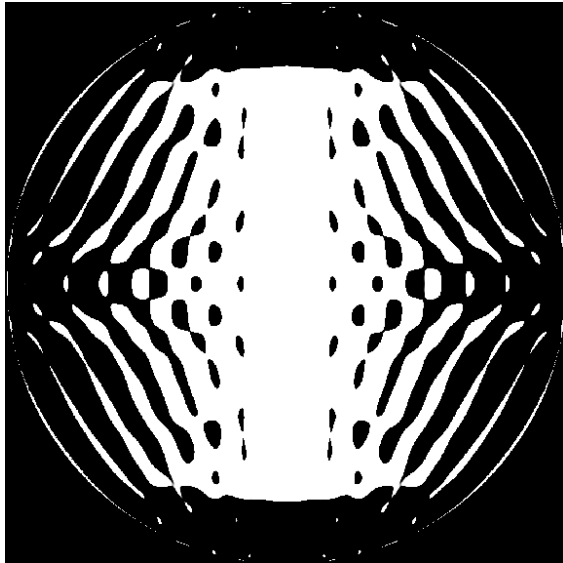
# Concentric Ring Masks



# Circular Unobstructed Pupil



# Circular Unobstructed Pupil



# Circular Unobstructed Pupil

