Iterative Methods for Image Reconstruction

Jeffrey A. Fessler

EECS Department
The University of Michigan

Ewha University, Seoul

Nov. 13, 2008

Image Reconstruction Methods

(Simplified View)

Analytical

(FBP)

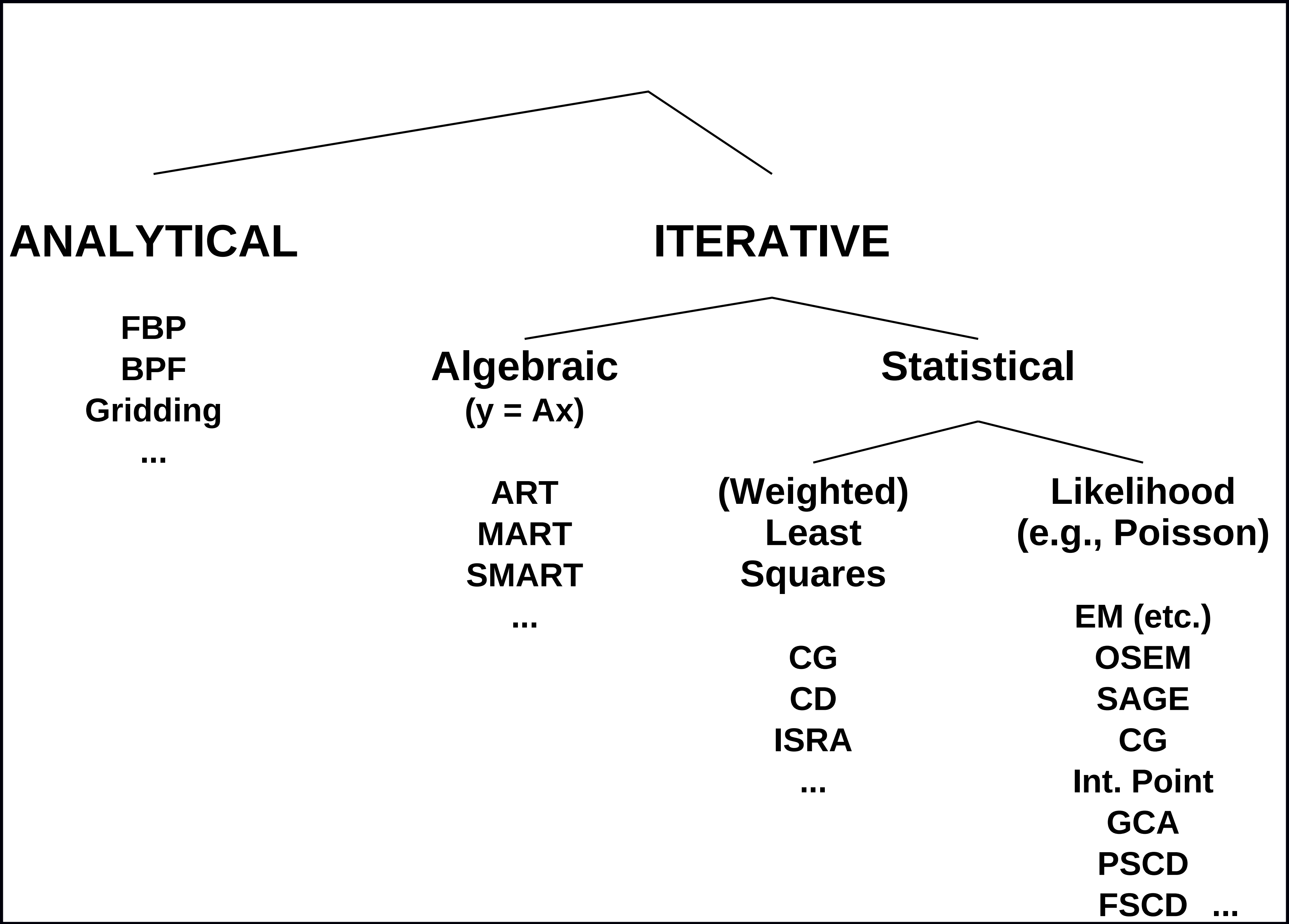
(MR: iFFT)

Iterative

(OSEM?)

(MR: CG?)

Image Reconstruction Methods / Algorithms



ANALYTICAL

FBP
BPF
Gridding
...

ITERATIVE

Algebraic
(y = Ax)

ART
MART
SMART
...

Statistical

(Weighted)
Least
Squares

CG
CD
ISRA
...

Likelihood
(e.g., Poisson)

EM (etc.)
OSEM
SAGE
CG
Int. Point
GCA
PSCD
FSCD ...

Outline :

Part 0: Introduction / Overview / Examples

Part 1: Problem Statements

- Continuous-discrete vs continuous-continuous vs discrete-discrete

Part 2: Four of Five Choices for Statistical Image Reconstruction

- Object parameterization
- System physical modeling
- Statistical modeling of measurements
- Cost functions and regularization

Part 3: Fifth Choice: Iterative algorithms

- Classical optimization methods
- Considerations: nonnegativity, convergence rate, ...
- Optimization transfer: EM etc.
- Ordered subsets / block iterative / incremental gradient methods

Part 4: Performance Analysis

- Spatial resolution properties
- Noise properties
- Detection performance

History

- Successive substitution method vs direct Fourier (Bracewell, 1956)
- Iterative method for emission tomography (Kuhl, 1963)
- Iterative method for X-ray CT (Hounsfield, 1968)
- ART for tomography (Gordon, Bender, Herman, JTB, 1970)
- Weighted least squares for 3D SPECT (Goitein, NIM, 1972)
- Richardson/Lucy iteration for image restoration (1972, 1974)
- Roughness regularized LS for tomography (Kashyap & Mittal, 1975)
- Proposals to use Poisson likelihood for emission and transmission tomography
  (Rockmore and Macovski, TNS, 1976, 1977)
- Expectation-maximization (EM) algorithms for Poisson model
  Emission: (Shepp and Vardi, TMI, 1982)
  Transmission: (Lange and Carson, JCAT, 1984)
- Regularized (aka Bayesian) Poisson emission reconstruction
  (Geman and McClure, ASA, 1985)
- Ordered-subsets EM algorithm (Hudson and Larkin, TMI, 1994)
- Commercial introduction of OSEM for PET scanners circa 1997

Why Statistical Methods?

- Object constraints (*e.g.*, nonnegativity, object support)
- Accurate physical models (less bias $\Longrightarrow$ improved quantitative accuracy) (*e.g.*, nonuniform attenuation in SPECT)
  improved spatial resolution?
- Appropriate statistical models (less variance $\Longrightarrow$ lower image noise) (FBP treats all rays equally)
- Side information (*e.g.*, MRI or CT boundaries)
- Nonstandard geometries (*e.g.*, irregular sampling or “missing” data)

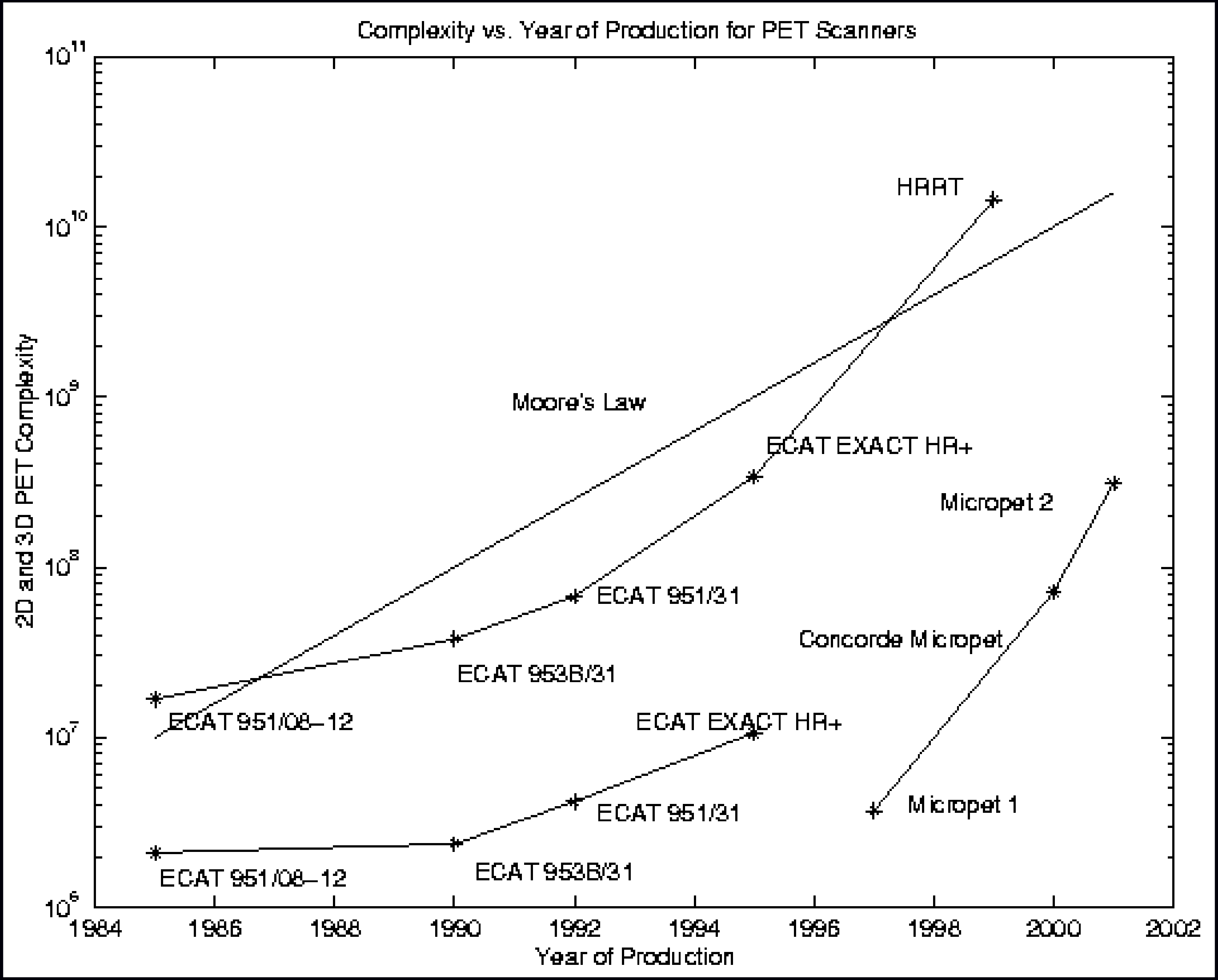
Disadvantages?

- Computation time
- Model complexity
- Software complexity

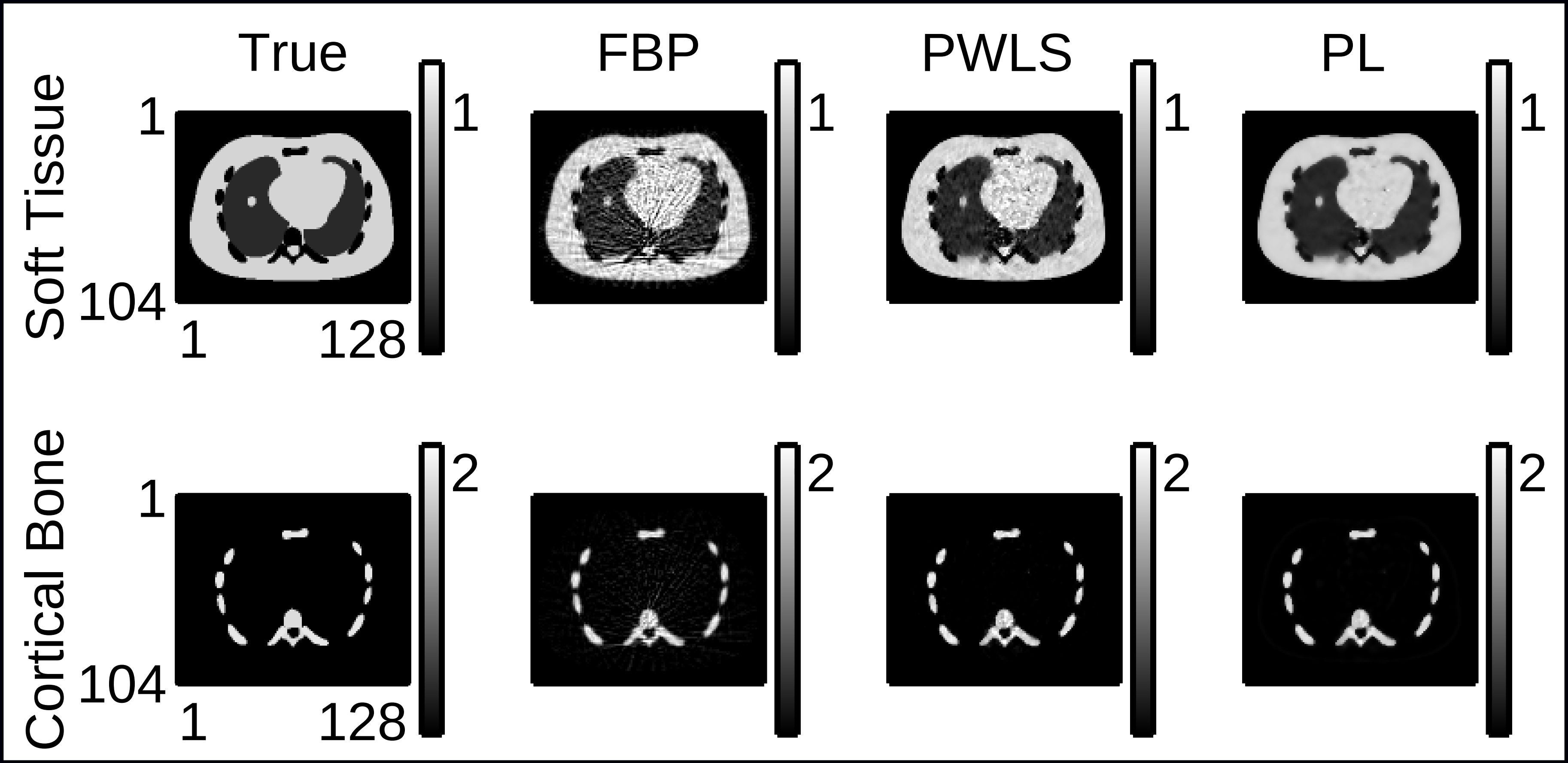
Analytical methods (a different short course!)

- Idealized mathematical model
  - Usually geometry only, greatly over-simplified physics
  - Continuum measurements (discretize/sample *after* solving)
- No statistical model
- Easier analysis of properties (due to linearity)
  e.g., Huesman (1984) FBP ROI variance for kinetic fitting

What about Moore’s Law?

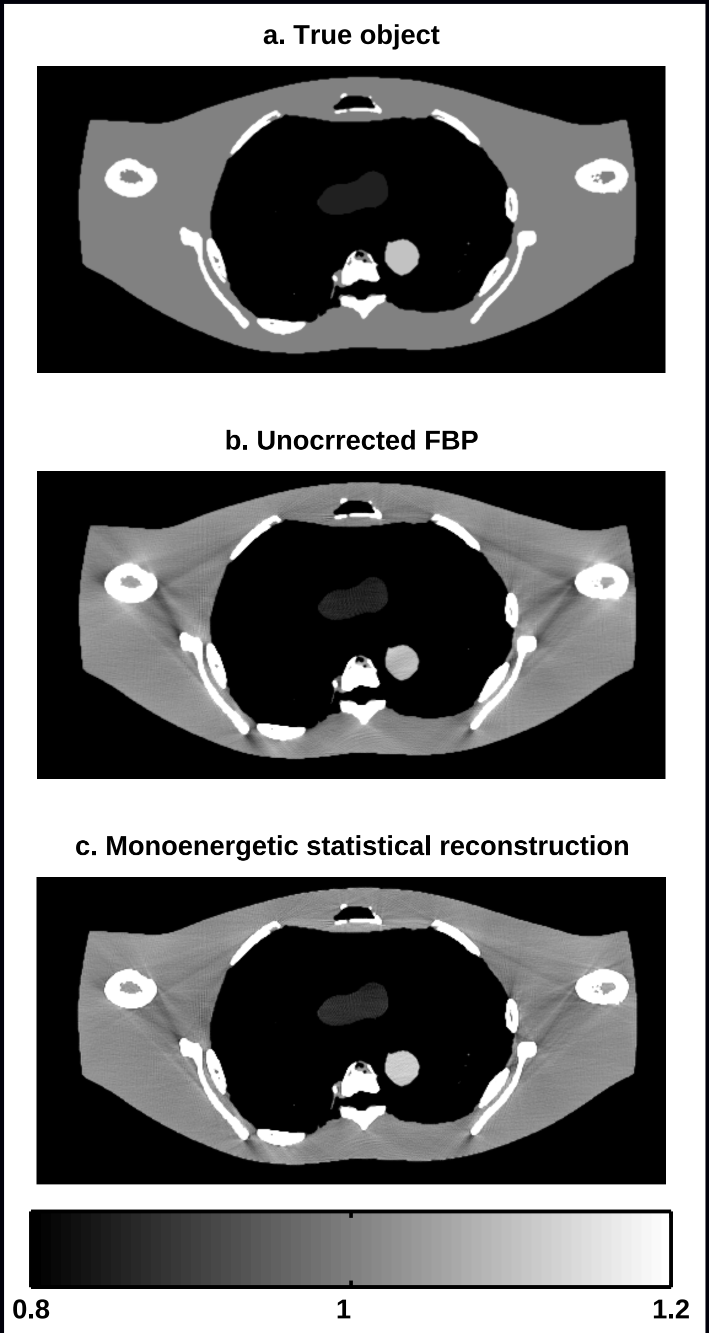


Benefit Example: Statistical Models



	NRMS Error	
Method	Soft Tissue	Cortical Bone
FBP	22.7%	29.6%
PWLS	13.6%	16.2%
PL	11.8%	15.8%

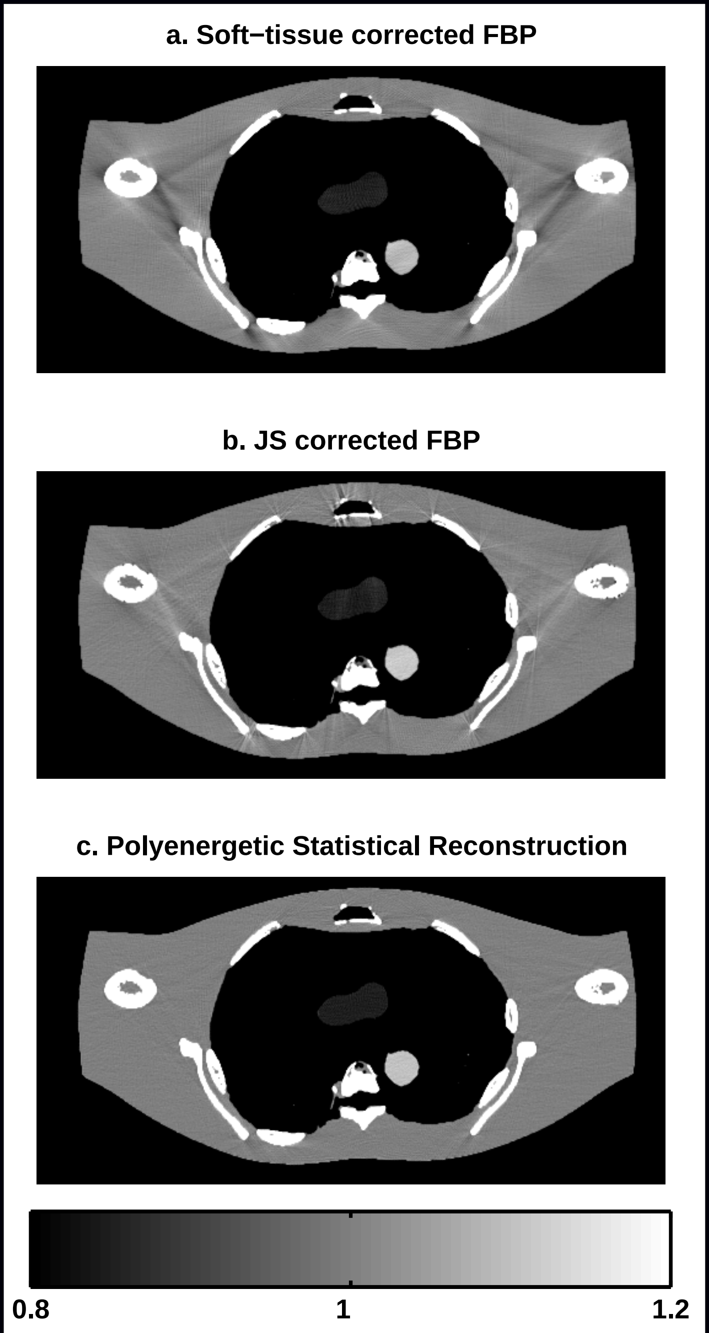
Benefit Example: Physical Models



a. True object

b. Unocrrected FBP

c. Monoenergetic statistical reconstruction

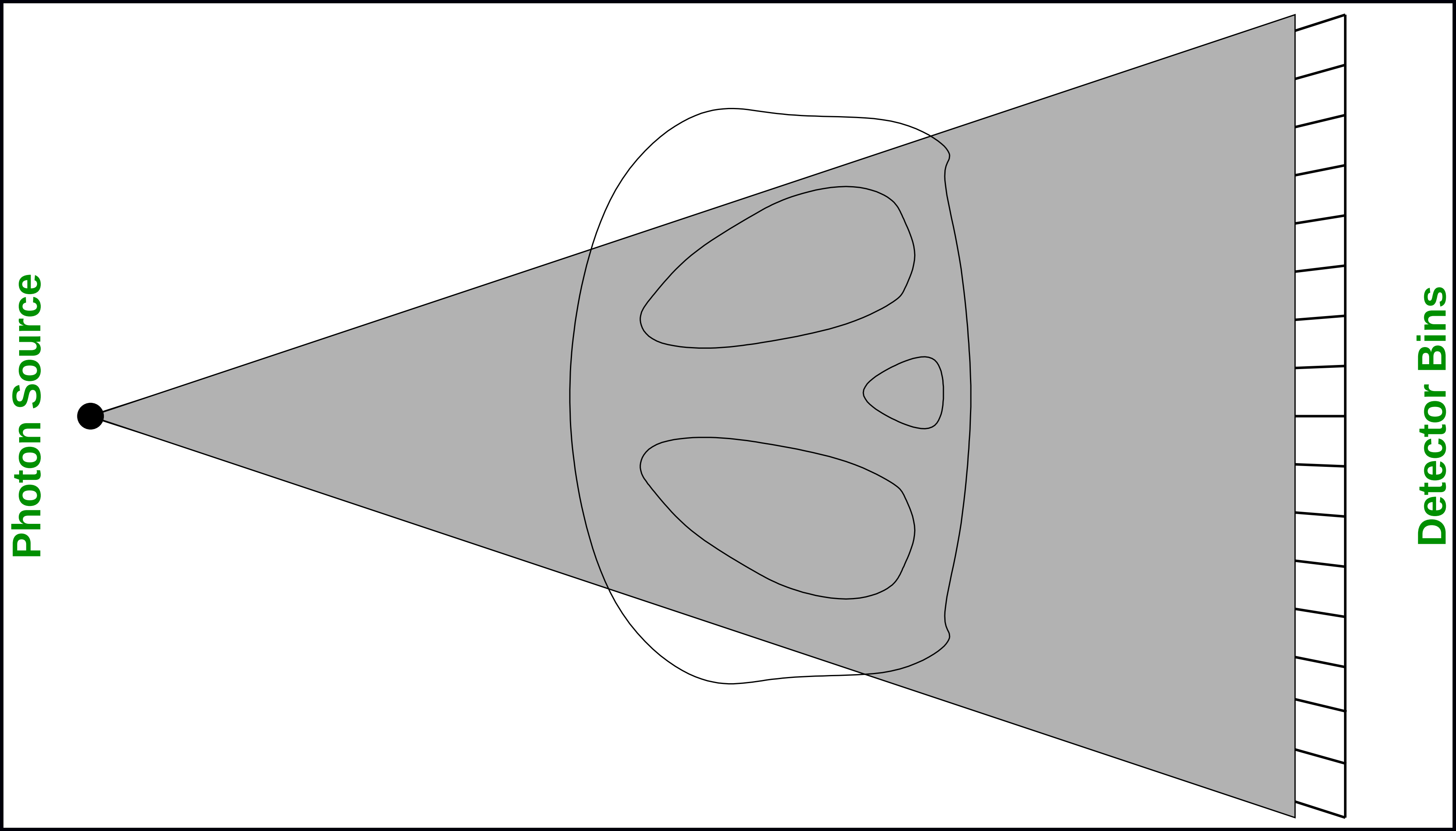


a. Soft−tissue corrected FBP

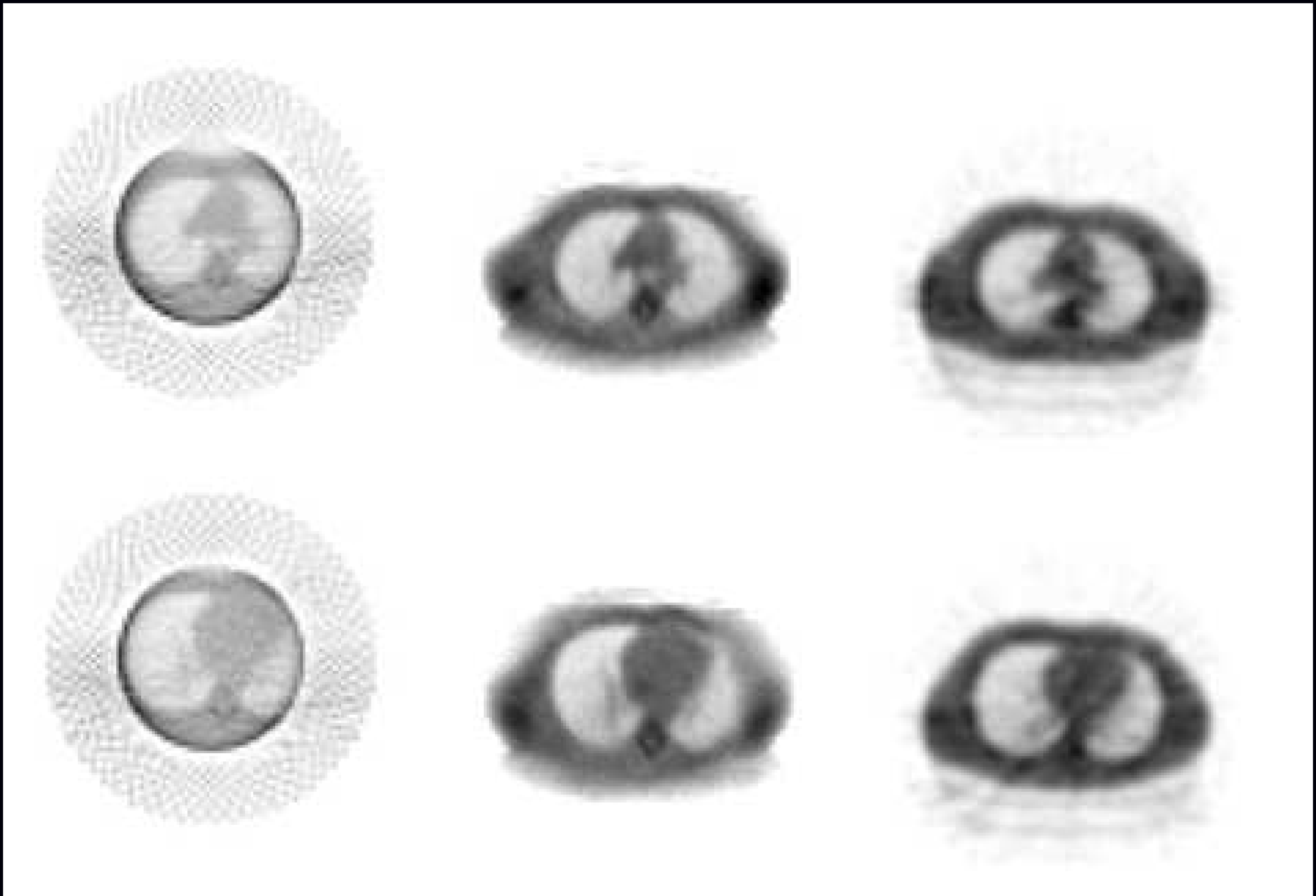
b. JS corrected FBP

c. Polyenergetic Statistical Reconstruction

Benefit Example: Nonstandard Geometries



Truncated Fan-Beam SPECT Transmission Scan

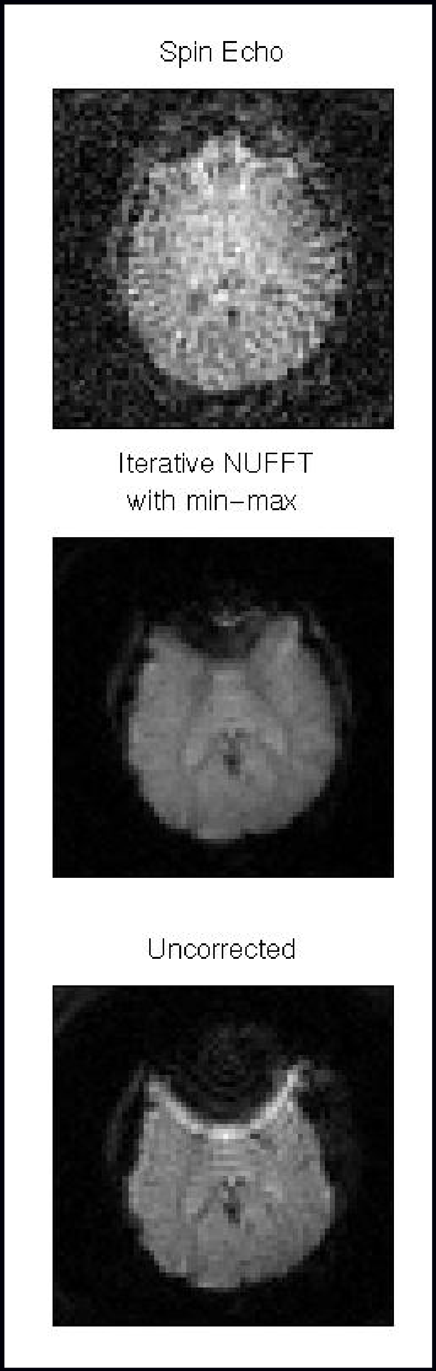


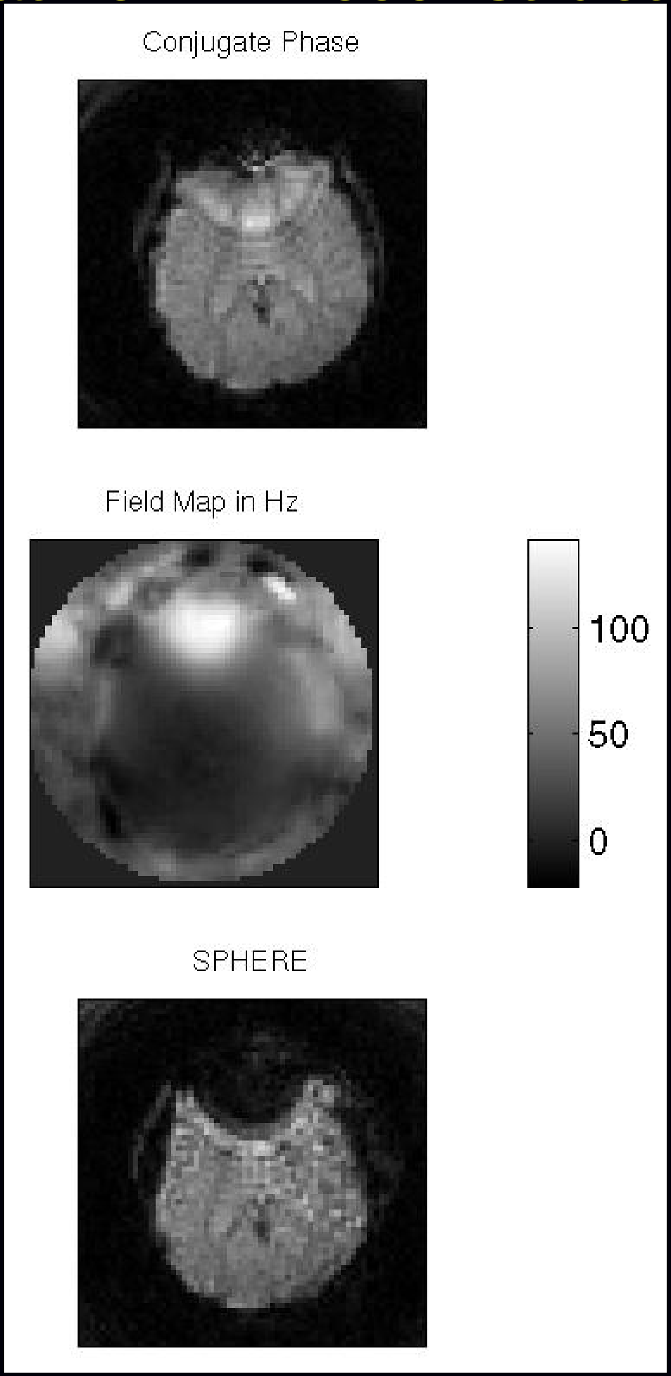
Truncated FBP

Truncated PWLS

Untruncated FBP

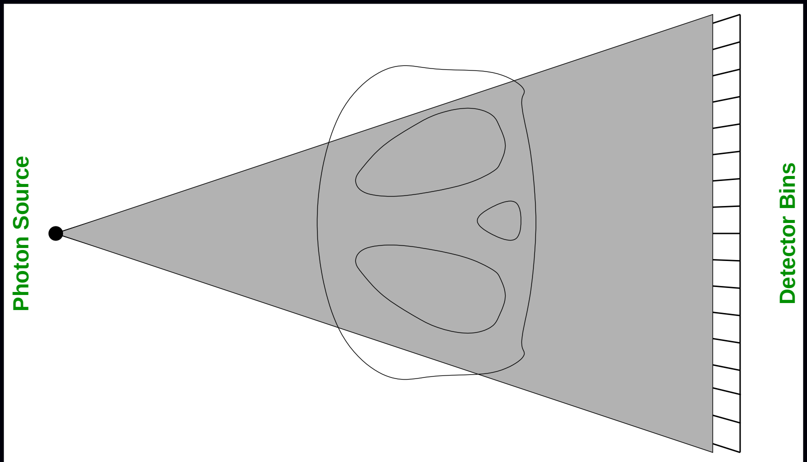
One Final Advertisement: Iterative MR Reconstruction





Part 1: Problem Statement(s)

Example:
in monoenergetic transmission tomography with photon counting detectors,
the goal is to reconstruct the *attenuation map* $\mu(\vec{x})$
from transmission measurements $\{y_i\}_{i=1}^{n_d}$,
given the system response $s_i(\vec{x})$, $i = 1, \ldots, n_d$, for each detector element.



Statistical model: $y_i \sim \text{Poisson}\{b_i \exp(-\int \mu(\vec{x})\, s_i(\vec{x})\, d\vec{x}) + r_i\}$

- b_i: blank/air scan
- $s_i(\vec{x})$: line impulse associated with line integral for ith ray, possibly including detector blur and finite source size (approximation)
- r_i: background due to Compton scatter

Continuous-Discrete Models

Emission tomography: $y_i \sim \text{Poisson}\{\int \lambda(\vec{x})\, s_i(\vec{x})\, d\vec{x} + r_i\}$

Transmission tomography (monoenergetic): $y_i \sim \text{Poisson}\left\{ b_i \exp\left(-\int_{\mathcal{L}_i} \mu(\vec{x})\, d\ell \right) + r_i \right\}$

Transmission (polyenergetic): $y_i \sim \text{Poisson}\left\{ \int I_i(\mathcal{E}) \exp\left(-\int_{\mathcal{L}_i} \mu(\vec{x}, \mathcal{E})\, d\ell \right) d\mathcal{E} + r_i \right\}$

Magnetic resonance imaging: $y_i = \int f(\vec{x})\, s_i(\vec{x})\, d\vec{x} + \varepsilon_i$

Discrete measurements $\boldsymbol{y} = (y_1, \ldots, y_{n_d})$
Continuous-space unknowns: $\lambda(\vec{x})$, $\mu(\vec{x})$, $f(\vec{x})$
Goal: estimate $f(\vec{x})$ given $\boldsymbol{y}$

Solution options:

- Continuous-continuous formulations (“analytical,” *cf.* FBP for tomography)
- Continuous-discrete formulations
  Usually $\hat{f}(\vec{x}) = \sum_{i=1}^{n_d} c_i\, s_i(\vec{x})$
- Discrete-discrete formulations $f(\vec{x}) \approx \sum_{j=1}^{n_p} x_j\, b_j(\vec{x})$

Textbook MRI Measurement Model

Ignoring *lots* of things, the standard measurement model is:

$$y_i = s(t_i) + \text{noise}_i, \qquad i = 1, \ldots, n_\text{d}$$

$$s(t) = \int f(\vec{x}) \, \mathrm{e}^{-\imath 2\pi \vec{\kappa}(t) \cdot \vec{x}} \, \mathrm{d}\vec{x} = F(\vec{\kappa}(t)) \, .$$

$\vec{x}$: spatial coordinates
$\vec{\kappa}(t)$: k-space trajectory of the MR pulse sequence
$f(\vec{x})$: object's unknown transverse magnetization
$F(\vec{\kappa})$: Fourier transform of $f(\vec{x})$. We get noisy samples of this!
$\mathrm{e}^{-\imath 2\pi \vec{\kappa}(t) \cdot \vec{x}}$ provides spatial information $\Longrightarrow$ Nobel Prize

Goal of image reconstruction: find $f(\vec{x})$ from measurements $\{y_i\}_{i=1}^{n_\text{d}}$.

The unknown object $f(\vec{x})$ is a continuous-space function,
but the recorded measurements $\boldsymbol{y} = (y_1, \ldots, y_{n_\text{d}})$ are finite.

Under-determined (ill posed) problem $\Longrightarrow$ no canonical solution.

All MR scans provide only "partial" k-space data.

Image Reconstruction Strategies

- Continuous-continuous formulation

Pretend that a continuum of measurements are available:

$$F(\vec{\kappa}) = \int f(\vec{x})\, e^{-\imath 2\pi \vec{\kappa}\cdot\vec{x}} \, d\vec{x}.$$

The “solution” is an inverse Fourier transform:

$$f(\vec{x}) = \int F(\vec{\kappa})\, e^{\imath 2\pi \vec{\kappa}\cdot\vec{x}} \, d\vec{\kappa}.$$

Now discretize the integral solution:

$$\hat{f}(\vec{x}) = \sum_{i=1}^{n_d} F(\vec{\kappa}_i)\, e^{\imath 2\pi \vec{\kappa}_i\cdot\vec{x}} w_i \approx \sum_{i=1}^{n_d} y_i w_i\, e^{\imath 2\pi \vec{\kappa}_i\cdot\vec{x}},$$

where w_i values are “sampling density compensation factors.”
Numerous methods for choosing w_i values in the literature.

For Cartesian sampling, using $w_i = 1/N$ suffices,
and the summation is an inverse FFT.
For non-Cartesian sampling, replace summation with gridding.

- Continuous-discrete formulation

Use many-to-one linear model:

$$\boldsymbol{y} = \mathcal{A}\, f + \boldsymbol{\varepsilon}, \text{ where } \mathcal{A} : \mathcal{L}_2(\mathbb{R}^{\bar{d}}) \to \mathbb{C}^{n_\mathrm{d}}.$$

Minimum norm solution (*cf.* “natural pixels”):

$$\min_{\hat{f}} \left\| \hat{f} \right\|_2 \text{ subject to } \boldsymbol{y} = \mathcal{A}\, \hat{f}$$

$$\hat{f} = \mathcal{A}^*(\mathcal{A}\mathcal{A}^*)^{-1}\boldsymbol{y} = \textstyle\sum_{i=1}^{n_\mathrm{d}} c_i \,\mathrm{e}^{-\imath 2\pi \vec{\kappa}_i \cdot \vec{\mathrm{x}}}, \text{ where } \mathcal{A}\mathcal{A}^*\boldsymbol{c} = \boldsymbol{y}.$$

- Discrete-discrete formulation

Assume parametric model for object:

$$f(\vec{\mathrm{x}}) = \sum_{j=1}^{n_\mathrm{p}} x_j \, b_j(\vec{\mathrm{x}}) \,.$$

Estimate parameter vector $\boldsymbol{x} = (x_1, \ldots, x_{n_\mathrm{p}})$ from data vector $\boldsymbol{y}$.

Part 2: Five Categories of Choices

- Object parameterization: function $f(\vec{r})$ vs finite coefficient vector $\boldsymbol{x}$
- System physical model: $\{s_i(\vec{r})\}$
- Measurement statistical model $y_i \sim \boxed{?}$
- Cost function: data-mismatch and regularization
- Algorithm / initialization

No perfect choices - one can critique all approaches!

Choice 1. Object Parameterization

Finite measurements: $\{y_i\}_{i=1}^{n_\mathrm{d}}$. Continuous object: $f(\vec{r})$. Hopeless?

“All models are wrong but some models are useful.”

Linear *series expansion* approach. Replace $f(\vec{r})$ by $\boldsymbol{x} = (x_1, \ldots, x_{n_\mathrm{p}})$ where

$$f(\vec{r}) \approx \tilde{f}(\vec{r}) = \sum_{j=1}^{n_\mathrm{p}} x_j b_j(\vec{r}) \leftarrow \text{“basis functions”}$$

Forward projection:

$$\int s_i(\vec{r})\, f(\vec{r})\, d\vec{r} = \int s_i(\vec{r}) \left[\sum_{j=1}^{n_\mathrm{p}} x_j b_j(\vec{r})\right] d\vec{r} = \sum_{j=1}^{n_\mathrm{p}} \left[\int s_i(\vec{r})\, b_j(\vec{r})\, d\vec{r}\right] x_j$$

$$= \sum_{j=1}^{n_\mathrm{p}} a_{ij} x_j = [\boldsymbol{A}\boldsymbol{x}]_i, \quad \text{where } a_{ij} \triangleq \int s_i(\vec{r})\, b_j(\vec{r})\, d\vec{r}$$

- Projection integrals become finite summations.
- a_{ij} is contribution of jth basis function (*e.g.*, voxel) to ith measurement.
- The units of a_{ij} and x_j depend on the user-selected units of $b_j(\vec{r})$.
- The $n_\mathrm{d} \times n_\mathrm{p}$ matrix $\boldsymbol{A} = \{a_{ij}\}$ is called the *system matrix*.

(Linear) Basis Function Choices

- Fourier series (complex / not sparse)
- Circular harmonics (complex / not sparse)
- Wavelets (negative values / not sparse)
- Kaiser-Bessel window functions (blobs)
- Overlapping circles (disks) or spheres (balls)
- Polar grids, logarithmic polar grids
- “Natural pixels” $\{s_i(\vec{r})\}$
- B-splines (pyramids)
- Rectangular pixels / voxels (rect functions)
- Point masses / bed-of-nails / lattice of points / “comb” function
- Organ-based voxels (*e.g.*, from CT in PET/CT systems)
- ...

Basis Function Considerations

Mathematical

- Represent $f(\vec{r})$ “well” with moderate n_{p} (approximation accuracy)
- *e.g.*, represent a constant (uniform) function
- Orthogonality? (not essential)
- Linear independence (ensures uniqueness of expansion)
- Insensitivity to shift of basis-function grid (approximate shift invariance)
- Rotation invariance

Computational

- “Easy” to compute a_{ij} values and/or $\boldsymbol{Ax}$
- If stored, the system matrix $\boldsymbol{A}$ should be sparse (mostly zeros).
- Easy to represent nonnegative functions *e.g.*, if $x_j \geq 0$, then $f(\vec{r}) \geq 0$.
  A sufficient condition is $b_j(\vec{r}) \geq 0$.

Nonlinear Object Parameterizations

Estimation of intensity *and* shape (*e.g.*, location, radius, etc.)

Surface-based (homogeneous) models

- Circles / spheres
- Ellipses / ellipsoids
- Superquadrics
- Polygons
- Bi-quadratic triangular Bezier patches, ...

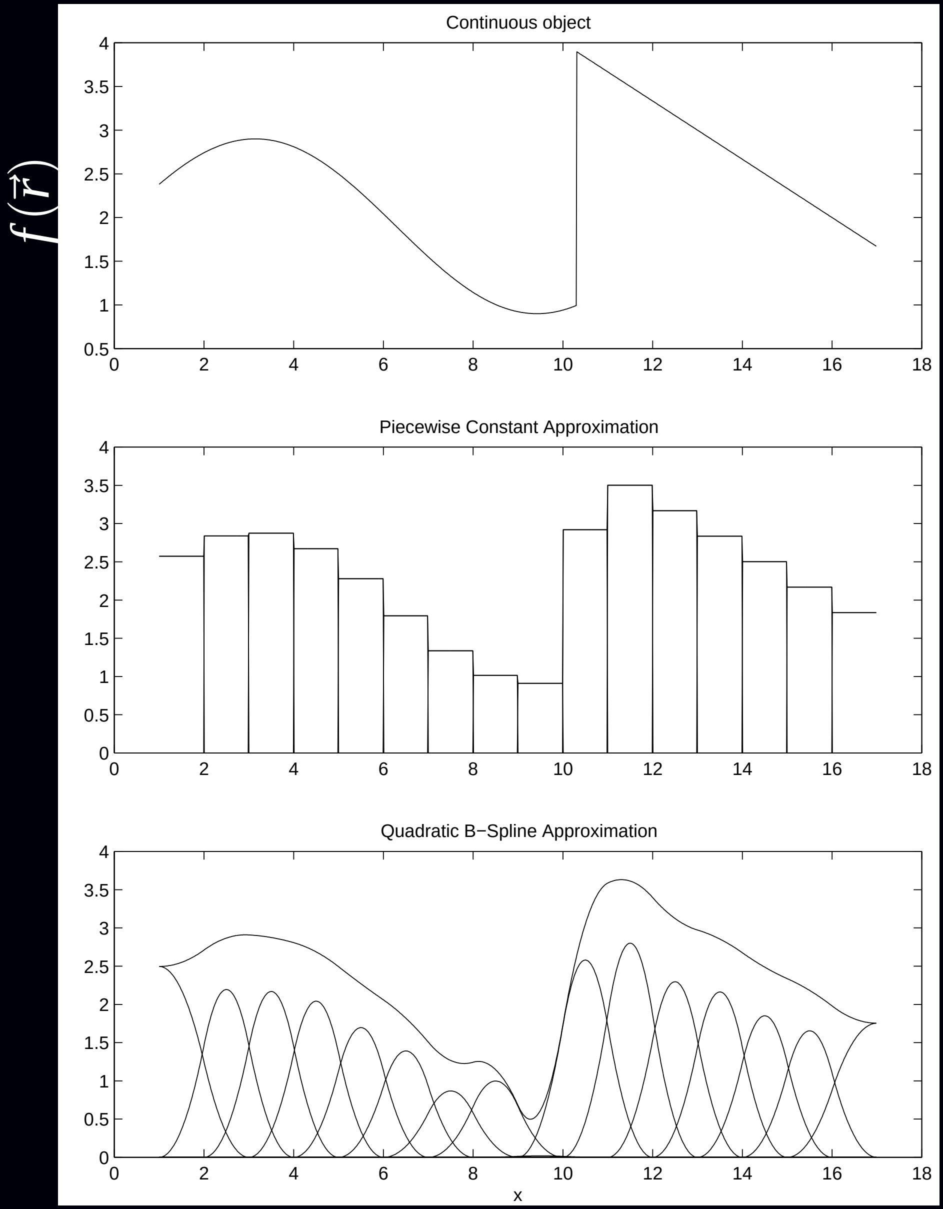
Other models

- Generalized series $f(\vec{r}) = \sum_j x_j b_j(\vec{r}, \boldsymbol{\theta})$
- Deformable templates $f(\vec{r}) = b(T_{\boldsymbol{\theta}}(\vec{r}))$
- ...

Considerations

- Can be considerably more parsimonious
- If correct, yield greatly reduced estimation error
- Particularly compelling in limited-data problems
- Often oversimplified (all models are wrong but...)
- Nonlinear dependence on location induces non-convex cost functions, complicating optimization

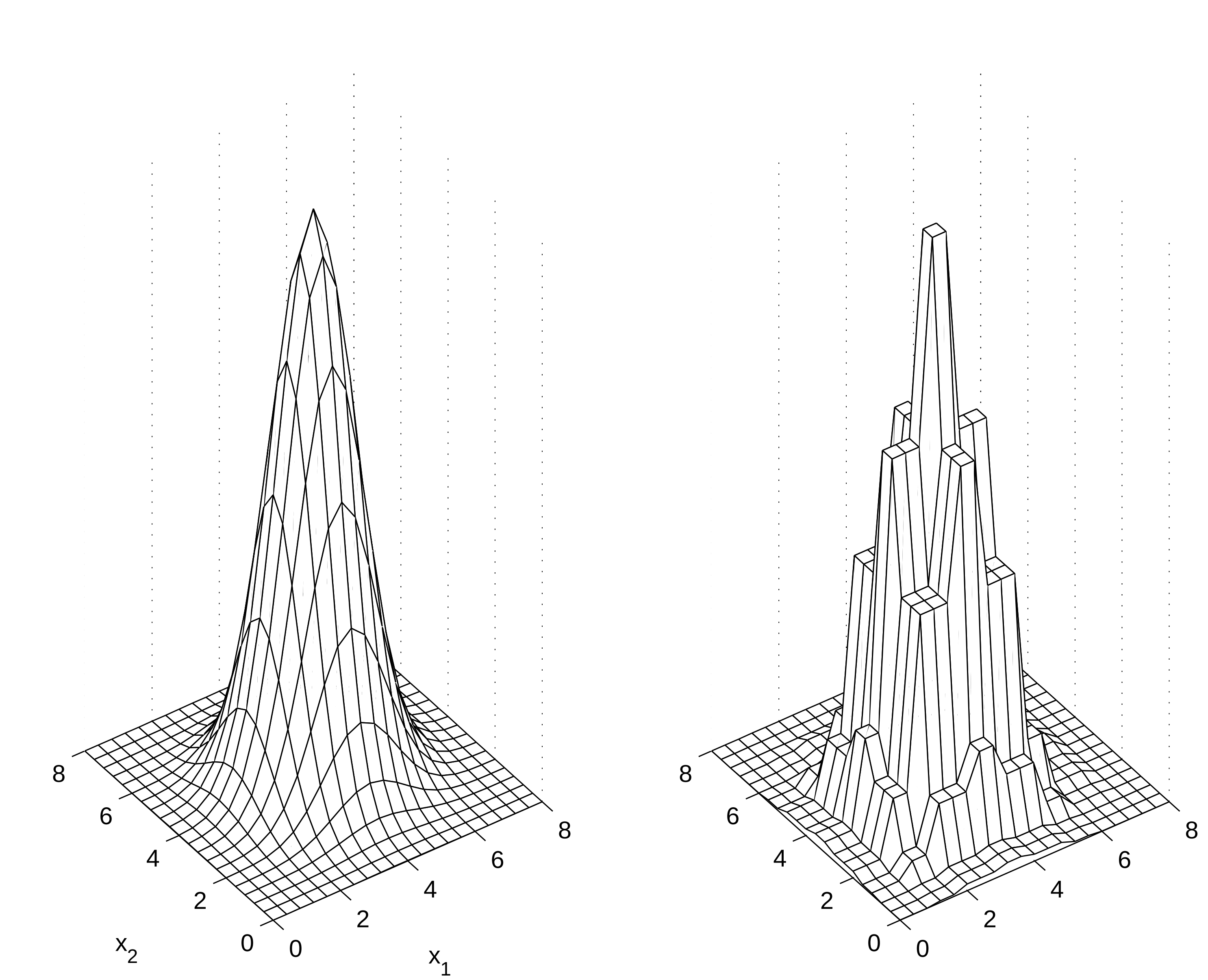
Example Basis Functions - 1D


Continuous object

Piecewise Constant Approximation

Quadratic B−Spline Approximation

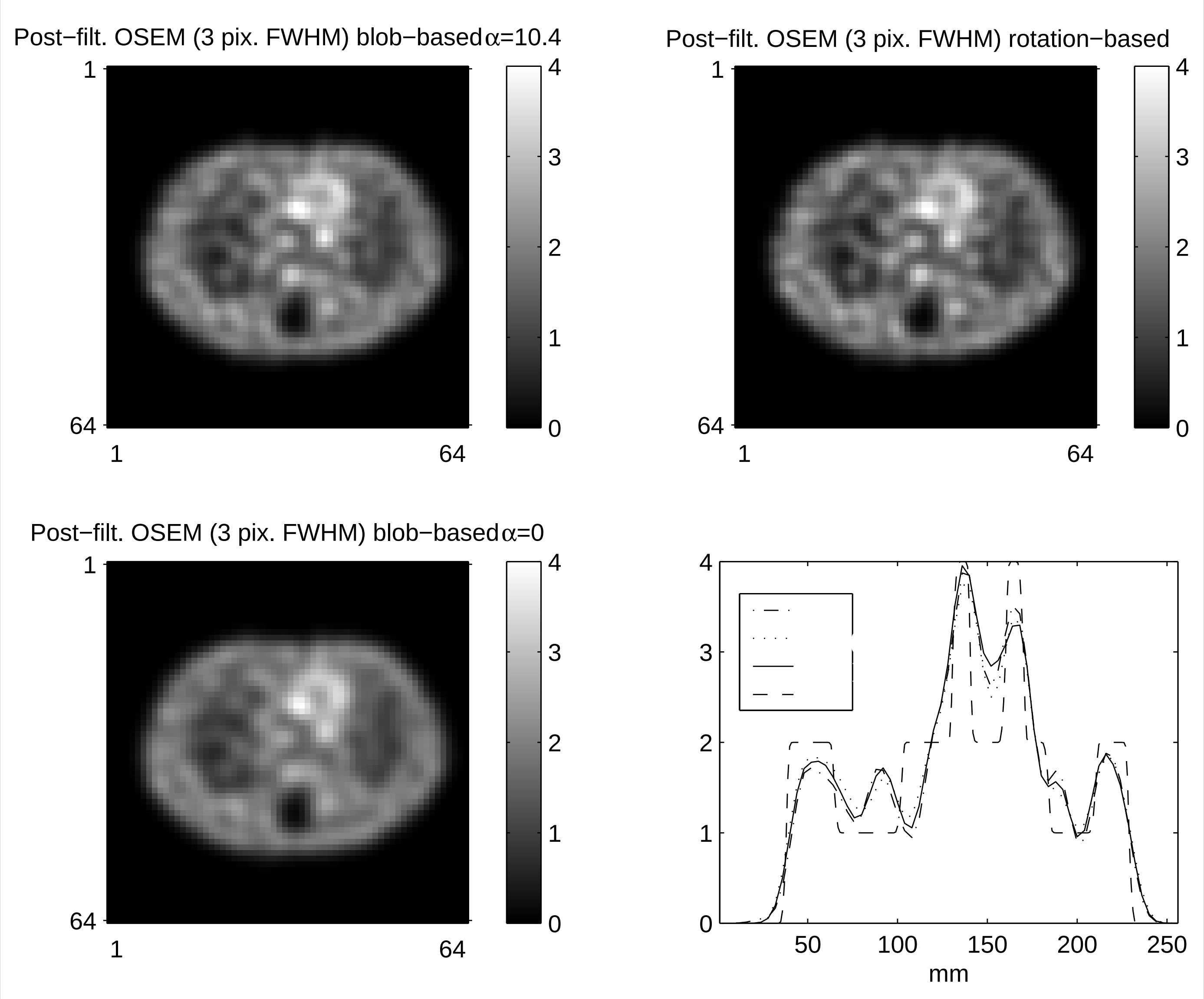
Pixel Basis Functions - 2D



Continuous image $f(\vec{r})$

Pixel basis approximation
$\sum_{j=1}^{n_{\mathrm{p}}} x_j b_j(\vec{r})$

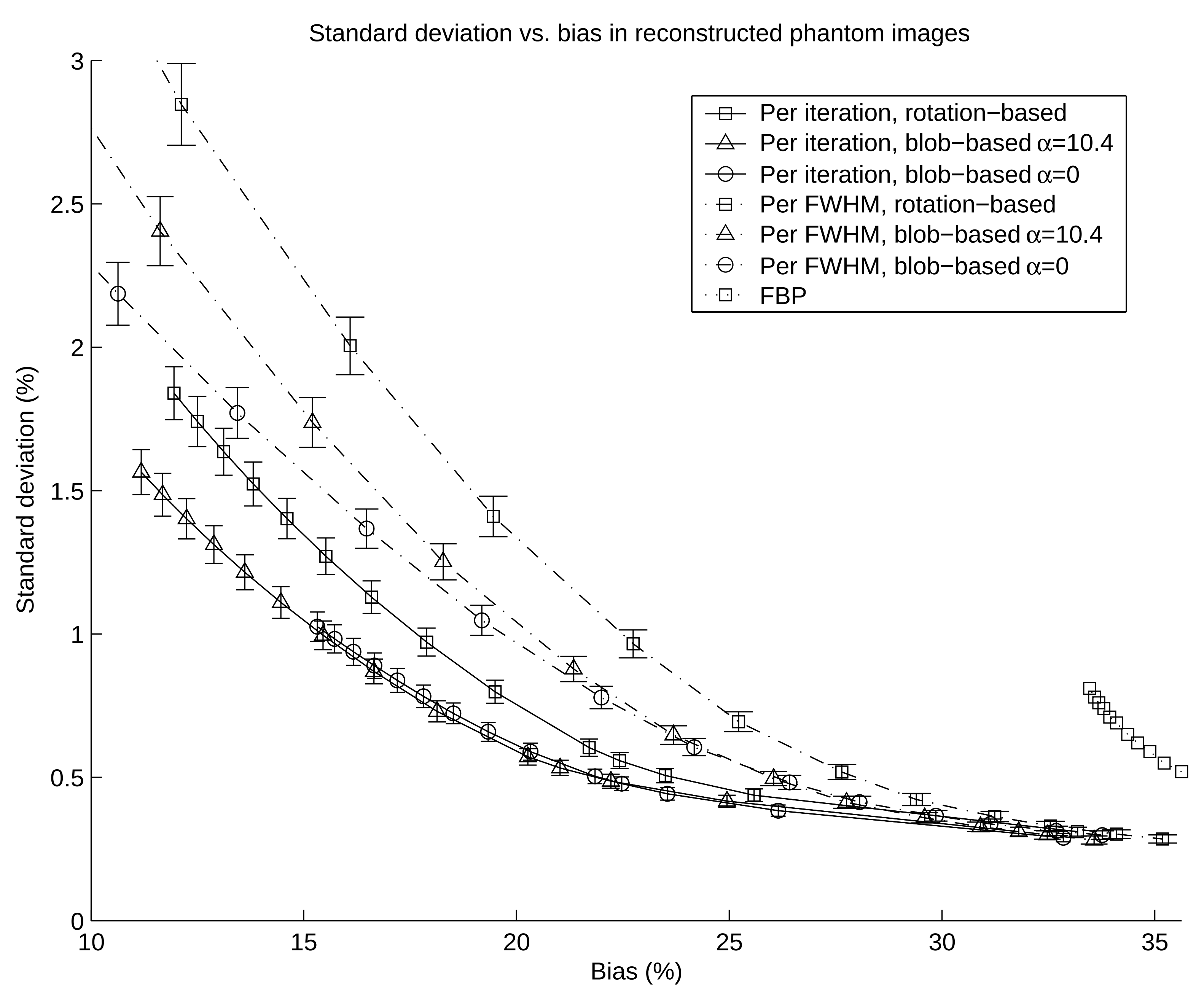
Blobs in SPECT: Qualitative



Post−filt. OSEM (3 pix. FWHM) blob−based α=10.4
Post−filt. OSEM (3 pix. FWHM) rotation−based
Post−filt. OSEM (3 pix. FWHM) blob−based α=0
1
64
1
64
0
1
2
3
4
50
100
150
200
250
mm

(2D SPECT thorax phantom simulations)

Blobs in SPECT: Quantitative



Discrete-Discrete Emission Reconstruction Problem

Having chosen a basis and *linearly* parameterized the emission density...

Estimate the emission density coefficient vector $\boldsymbol{x} = (x_1, \ldots, x_{n_\mathrm{p}})$
(aka “image”) using (something like) this statistical model:

$$y_i \sim \text{Poisson}\left\{\sum_{j=1}^{n_\mathrm{p}} a_{ij} x_j + r_i\right\}, \qquad i = 1, \ldots, n_\mathrm{d}.$$

- $\{y_i\}_{i=1}^{n_\mathrm{d}}$: observed counts from each detector unit
- $\boldsymbol{A} = \{a_{ij}\}$: system matrix (determined by system models)
- r_i values : background contributions (determined separately)

Many image reconstruction problems are “find $\boldsymbol{x}$ given $\boldsymbol{y}$” where

$$y_i = g_i([\boldsymbol{A}\boldsymbol{x}]_i) + \varepsilon_i, \qquad i = 1, \ldots, n_\mathrm{d}.$$

Choice 2. System Model, aka Physics

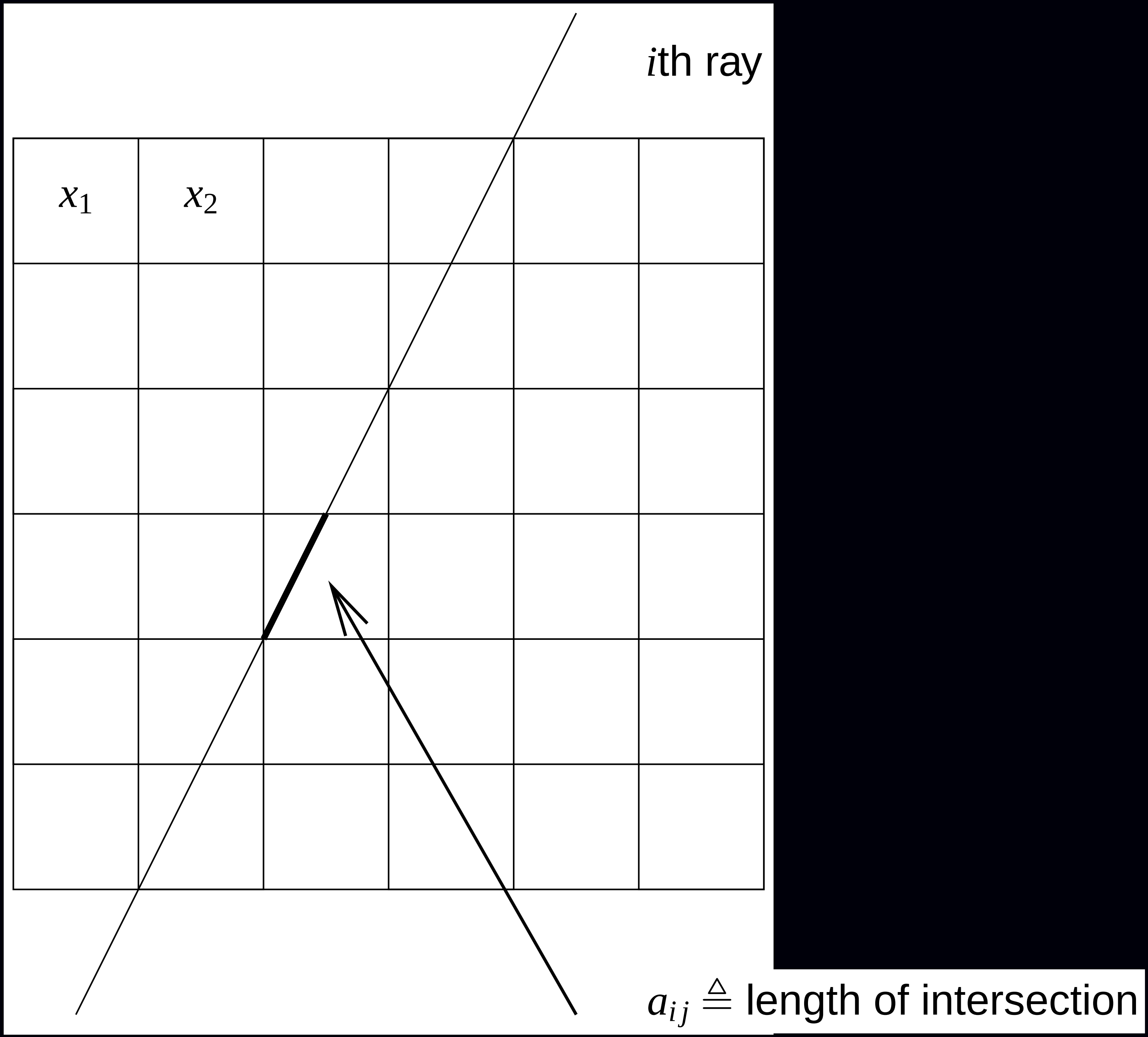
System matrix elements: $a_{ij} = \int s_i(\vec{r})\, b_j(\vec{r})\, \mathrm{d}\vec{r}$

- scan geometry
- collimator/detector response
- attenuation
- scatter (object, collimator, scintillator)
- duty cycle (dwell time at each angle)
- detector efficiency / dead-time losses
- positron range, noncollinearity, crystal penetration, ...
- ...

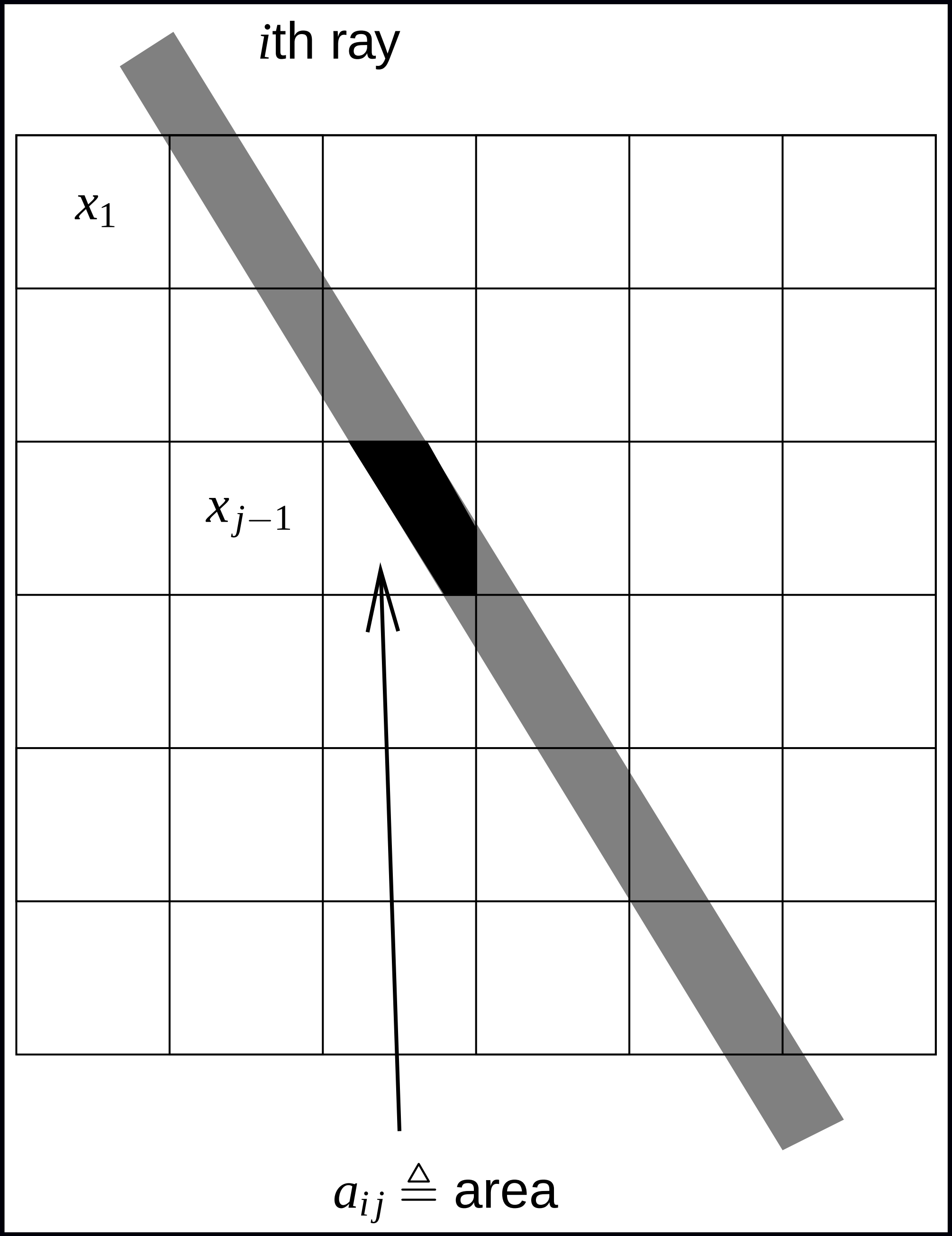
Considerations

- Improving system model can improve
  - Quantitative accuracy
  - Spatial resolution
  - Contrast, SNR, detectability
- Computation time (and storage vs compute-on-fly)
- Model uncertainties
  (*e.g.*, calculated scatter probabilities based on noisy attenuation map)
- Artifacts due to over-simplifications

“Line Length” System Model for Tomography

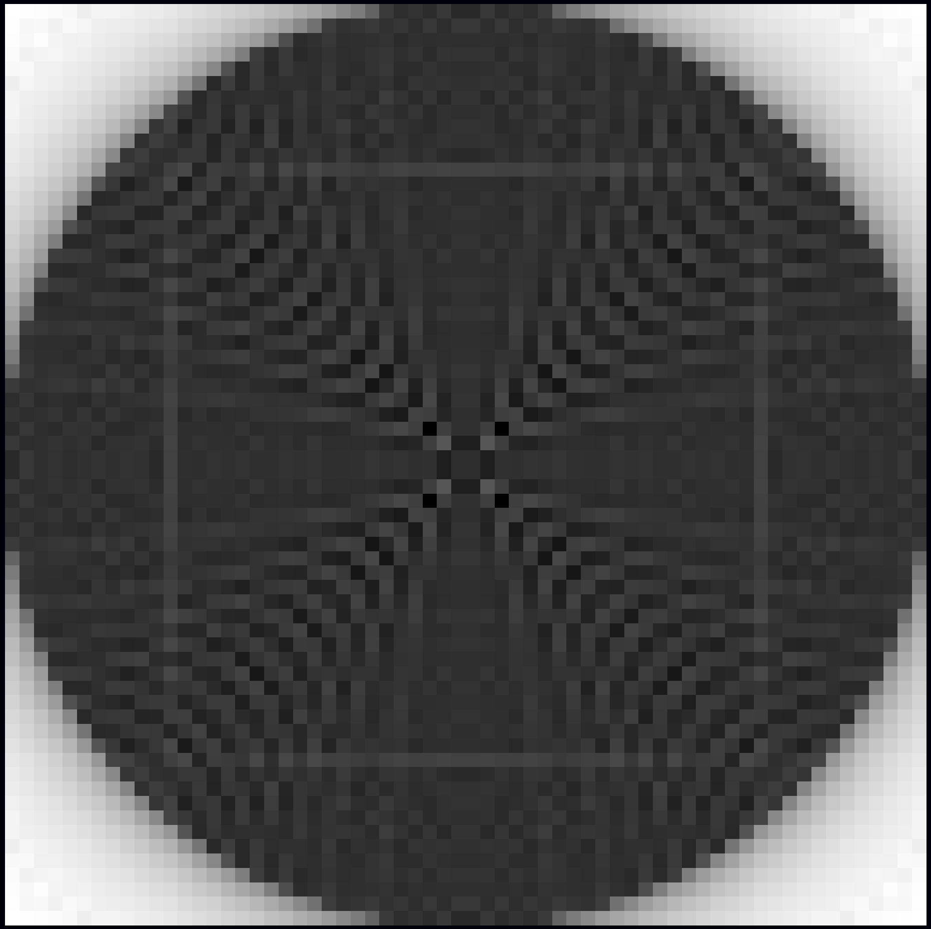


“Strip Area” System Model for Tomography

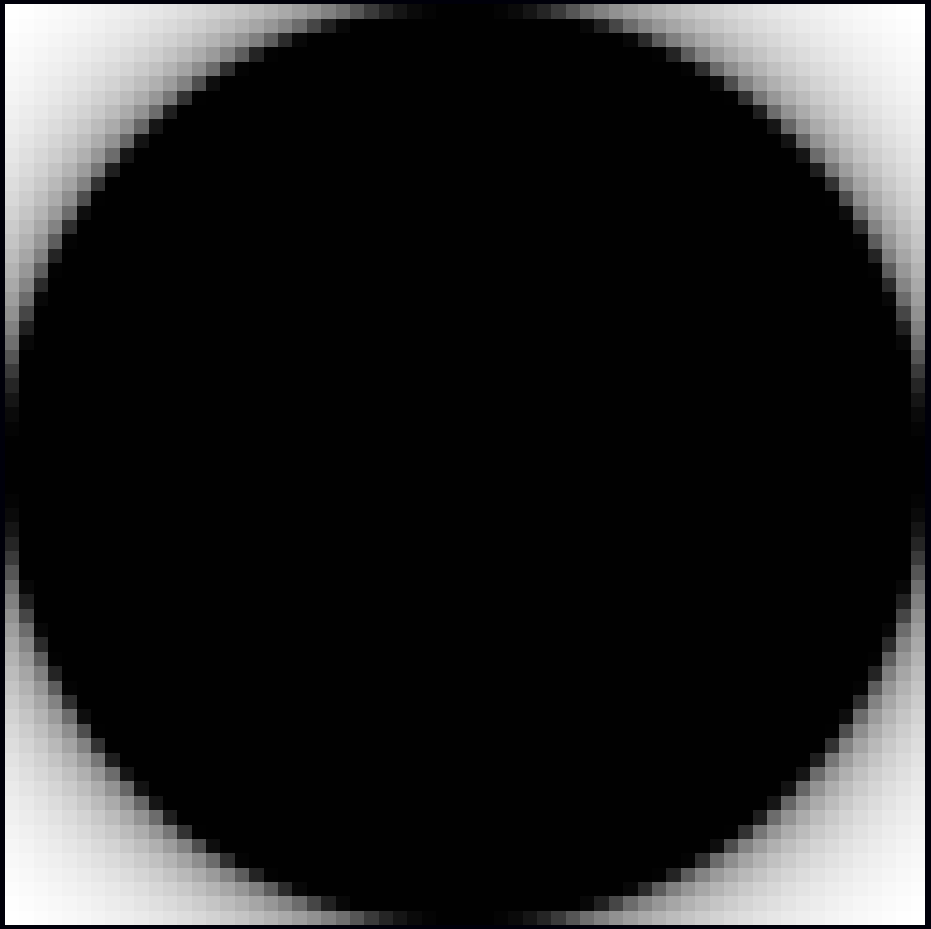


(Implicit) System Sensitivity Patterns

$$\sum_{i=1}^{n_\mathrm{d}} a_{ij} \approx s(\vec{r}_j) = \sum_{i=1}^{n_\mathrm{d}} s_i(\vec{r}_j)$$



Line Length



Strip Area

Forward- / Back-projector “Pairs”

Forward projection (image domain to projection domain):

$$\bar{y}_i = \int s_i(\vec{r})\, f(\vec{r})\, \mathrm{d}\vec{r} = \sum_{j=1}^{n_\mathrm{p}} a_{ij} x_j = [\boldsymbol{A}\boldsymbol{x}]_i\,, \quad \text{or} \quad \bar{\boldsymbol{y}} = \boldsymbol{A}\boldsymbol{x}$$

Backprojection (projection domain to image domain):

$$\boldsymbol{A}'\boldsymbol{y} = \left\{ \sum_{i=1}^{n_\mathrm{d}} a_{ij} y_i \right\}_{j=1}^{n_\mathrm{p}}$$

The term “forward/backprojection pair” often corresponds to an implicit choice for the object basis and the system model.

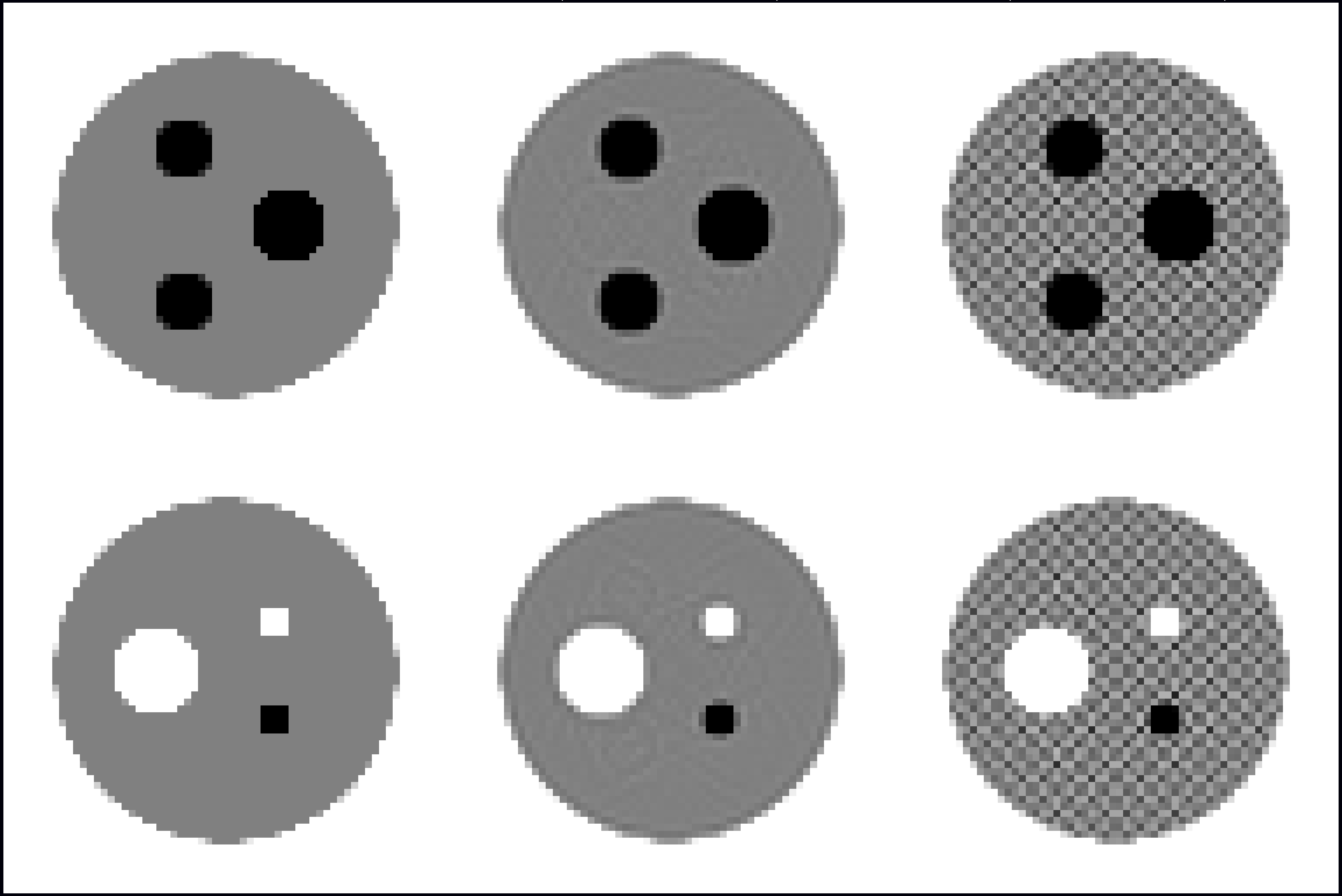
Sometimes $\boldsymbol{A}'\boldsymbol{y}$ is implemented as $\boldsymbol{B}\boldsymbol{y}$ for some “backprojector” $\boldsymbol{B} \neq \boldsymbol{A}'$

Least-squares solutions (for example):

$$\hat{\boldsymbol{x}} = [\boldsymbol{A}'\boldsymbol{A}]^{-1}\boldsymbol{A}'\boldsymbol{y} \neq [\boldsymbol{B}\boldsymbol{A}]^{-1}\boldsymbol{B}\boldsymbol{y}$$

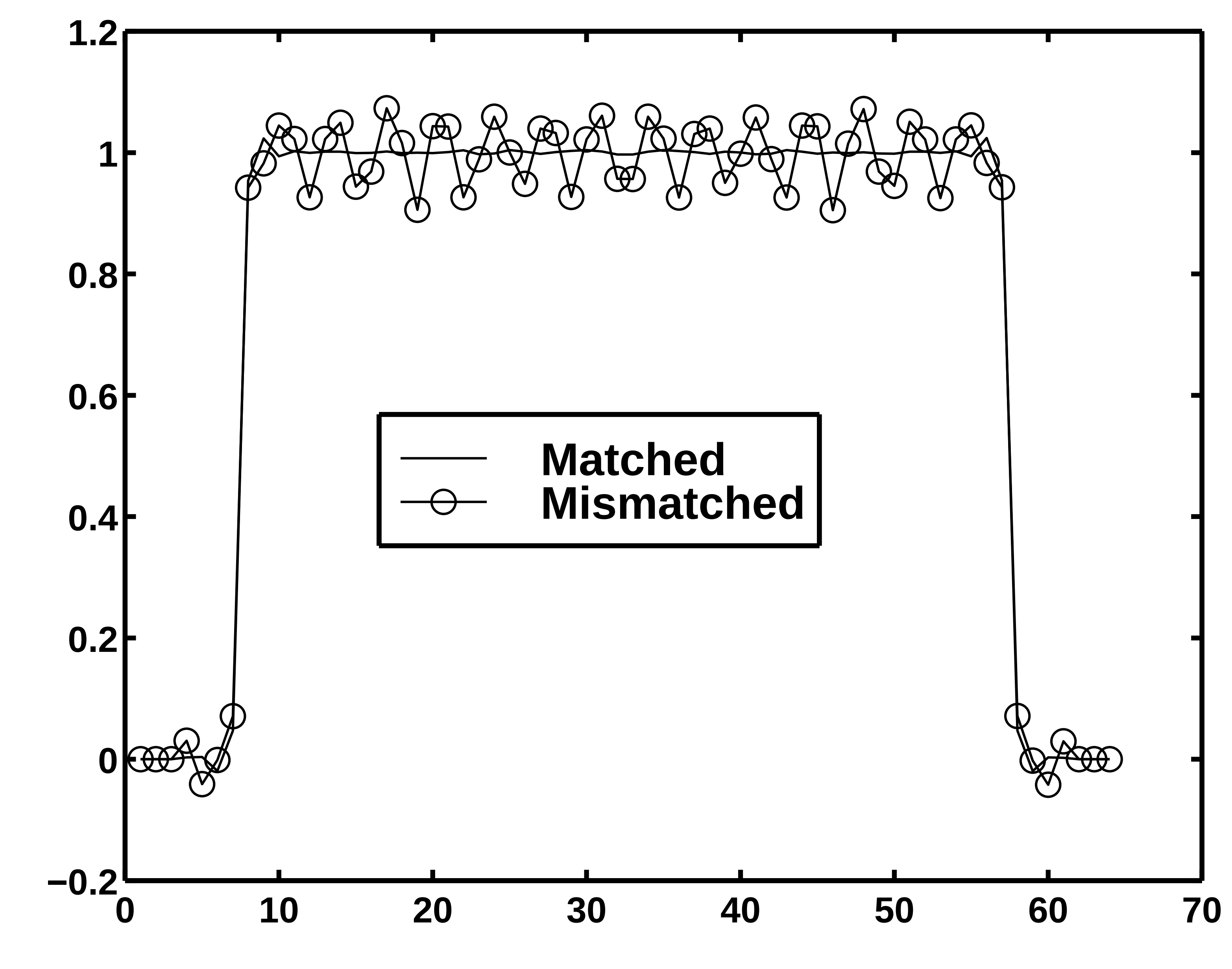
Mismatched Backprojector $B \neq A'$

$\boldsymbol{x}$ $\hat{\boldsymbol{x}}(PWLS-CG)$ $\hat{\boldsymbol{x}}(PWLS-CG)$

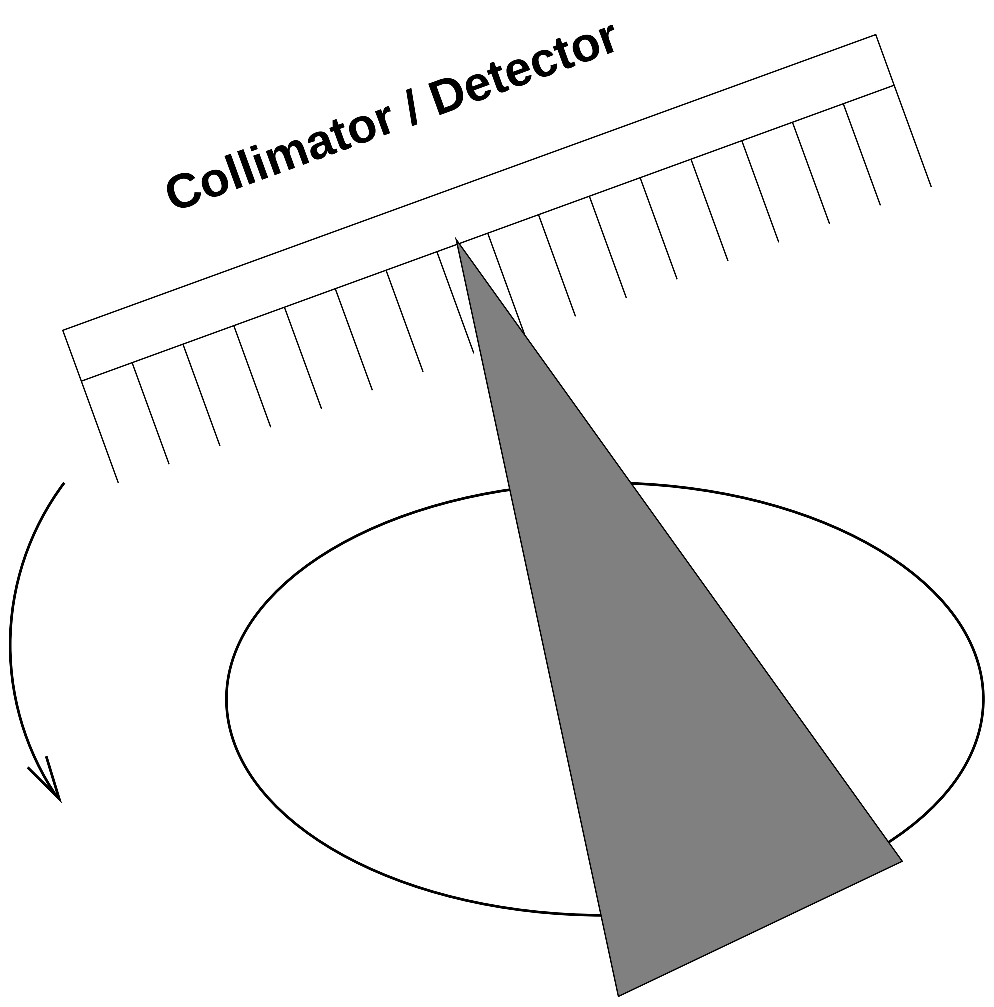


Matched Mismatched

Horizontal Profiles



SPECT System Modeling



Complications: nonuniform attenuation, depth-dependent PSF, Compton scatter

Choice 3. Statistical Models

After modeling the system physics, we have a deterministic “model:”

$$y_i \approx g_i([\boldsymbol{Ax}]_i)$$

for some functions g_i, *e.g.*, $g_i(l) = l + r_i$ for emission tomography.

Statistical modeling is concerned with the “ $\approx$ ” aspect.

Considerations

- More accurate models:
  - can lead to lower variance images,
  - may incur additional computation,
  - may involve additional algorithm complexity (*e.g.*, proper transmission Poisson model has nonconcave log-likelihood)
- Statistical model errors (*e.g.*, deadtime)
- Incorrect models (*e.g.*, log-processed transmission data)

Statistical Model Choices for Emission Tomography

- "None." Assume $\boldsymbol{y} - \boldsymbol{r} = \boldsymbol{A}\boldsymbol{x}$. "Solve algebraically" to find $\boldsymbol{x}$.
- White Gaussian noise. Ordinary least squares: minimize $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2$
  (This is the appropriate statistical model for MR.)
- Non-white Gaussian noise. Weighted least squares: minimize

$$\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_{\boldsymbol{W}}^2 = \sum_{i=1}^{n_{\mathrm{d}}} w_i (y_i - [\boldsymbol{A}\boldsymbol{x}]_i)^2, \quad \text{where} \quad [\boldsymbol{A}\boldsymbol{x}]_i \triangleq \sum_{j=1}^{n_{\mathrm{p}}} a_{ij} x_j$$

  (*e.g.*, for Fourier rebinned (FORE) PET data)
- Ordinary Poisson model (ignoring or precorrecting for background)

$$y_i \sim \text{Poisson}\{[\boldsymbol{A}\boldsymbol{x}]_i\}$$

- Poisson model

$$y_i \sim \text{Poisson}\{[\boldsymbol{A}\boldsymbol{x}]_i + r_i\}$$

- Shifted Poisson model (for randoms precorrected PET)

$$y_i = y_i^{\text{prompt}} - y_i^{\text{delay}} \sim \text{Poisson}\{[\boldsymbol{A}\boldsymbol{x}]_i + 2r_i\} - 2r_i$$

Shifted-Poisson Model for X-ray CT

A model that includes both photon variability and electronic readout noise:

$$y_i \sim \mathsf{Poisson}\{\bar{y}_i(\boldsymbol{\mu})\} + \mathsf{N}\left(0, \sigma^2\right)$$

Shifted Poisson approximation (matches first two moments):

$$\left[y_i + \sigma^2\right]_+ \sim \mathsf{Poisson}\left\{\bar{y}_i(\boldsymbol{\mu}) + \sigma^2\right\}$$

or just use WLS...

Complications:

- Intractability of likelihood for Poisson+Gaussian
- Compound Poisson distribution due to photon-energy-dependent detector signal.

X-ray statistical modeling is a current research area in several groups!

Choice 4. Cost Functions

Components:

- *Data-mismatch* term
- *Regularization* term (and regularization parameter β)
- Constraints (*e.g.*, nonnegativity)

Cost function:

$$\Psi(\boldsymbol{x}) = \text{DataMismatch}(\boldsymbol{y}, \boldsymbol{A}\boldsymbol{x}) + \beta\,\text{Roughness}(\boldsymbol{x})$$

Reconstruct image $\hat{\boldsymbol{x}}$ by minimization:

$$\hat{\boldsymbol{x}} \triangleq \arg\min_{\boldsymbol{x} \geq \boldsymbol{0}} \Psi(\boldsymbol{x})$$

Actually *several* sub-choices to make for Choice 4 ...

Distinguishes “statistical methods” from “algebraic methods” for “$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$.”

Why Cost Functions?

(vs “procedure” *e.g.*, adaptive neural net with wavelet denoising)

Theoretical reasons

ML is based on minimizing a cost function: the negative log-likelihood

- ML is asymptotically consistent
- ML is asymptotically unbiased
- ML is asymptotically efficient (under true statistical model...)
- Estimation: Penalized-likelihood achieves uniform CR bound asymptotically
- Detection: Qi and Huesman showed analytically that MAP reconstruction outperforms FBP for SKE/BKE lesion detection (T-MI, Aug. 2001)

Practical reasons

- Stability of estimates (if Ψ and algorithm chosen properly)
- Predictability of properties (despite nonlinearities)
- Empirical evidence (?)

Bayesian Framework

Given a prior distribution $\mathsf{p}(\boldsymbol{x})$ for image vectors $\boldsymbol{x}$, by Bayes' rule:

$$\text{posterior:} \quad \mathsf{p}(\boldsymbol{x}|\boldsymbol{y}) = \mathsf{p}(\boldsymbol{y}|\boldsymbol{x})\, \mathsf{p}(\boldsymbol{x}) \,/\, \mathsf{p}(\boldsymbol{y})$$

so

$$\log \mathsf{p}(\boldsymbol{x}|\boldsymbol{y}) = \log \mathsf{p}(\boldsymbol{y}|\boldsymbol{x}) + \log \mathsf{p}(\boldsymbol{x}) - \log \mathsf{p}(\boldsymbol{y})$$

- $-\log \mathsf{p}(\boldsymbol{y}|\boldsymbol{x})$ corresponds to data mismatch term (negative log-likelihood)
- $-\log \mathsf{p}(\boldsymbol{x})$ corresponds to regularizing penalty function

Maximum a posteriori (MAP) estimator:

$$\hat{\boldsymbol{x}} = \arg\max_{\boldsymbol{x}} \log \mathsf{p}(\boldsymbol{x}|\boldsymbol{y}) = \arg\max_{\boldsymbol{x}} \log \mathsf{p}(\boldsymbol{y}|\boldsymbol{x}) + \log \mathsf{p}(\boldsymbol{x})$$

- Has certain optimality properties (provided $\mathsf{p}(\boldsymbol{y}|\boldsymbol{x})$ and $\mathsf{p}(\boldsymbol{x})$ are correct).
- Same form as Ψ

Choice 4.1: Data-Mismatch Term

Options (for emission tomography):

- Negative log-likelihood of statistical model. Poisson *emission* case:

$$-L(\boldsymbol{x};\boldsymbol{y}) = -\log \mathrm{p}(\boldsymbol{y}|\boldsymbol{x}) = \sum_{i=1}^{n_\mathrm{d}} ([\boldsymbol{A}\boldsymbol{x}]_i + r_i) - y_i \log([\boldsymbol{A}\boldsymbol{x}]_i + r_i) + \log y_i!$$

- Ordinary (unweighted) least squares: $\sum_{i=1}^{n_\mathrm{d}} \frac{1}{2}(y_i - \hat{r}_i - [\boldsymbol{A}\boldsymbol{x}]_i)^2$
- Data-weighted least squares: $\sum_{i=1}^{n_\mathrm{d}} \frac{1}{2}(y_i - \hat{r}_i - [\boldsymbol{A}\boldsymbol{x}]_i)^2/\hat{\sigma}_i^2, \; \hat{\sigma}_i^2 = \max\left(y_i + \hat{r}_i, \sigma_\mathrm{min}^2\right),$ (causes bias due to data-weighting).
- Reweighted least-squares: $\hat{\sigma}_i^2 = [\boldsymbol{A}\hat{\boldsymbol{x}}]_i + \hat{r}_i$
- Model-weighted least-squares (nonquadratic, but convex!)

$$\sum_{i=1}^{n_\mathrm{d}} \frac{1}{2}(y_i - \hat{r}_i - [\boldsymbol{A}\boldsymbol{x}]_i)^2/([\boldsymbol{A}\boldsymbol{x}]_i + \hat{r}_i)$$

- Nonquadratic cost-functions that are robust to outliers
- ...

Considerations

- Faithfulness to statistical model vs computation
- Ease of optimization (convex?, quadratic?)
- Effect of statistical modeling errors

Choice 4.2: Regularization

Forcing too much “data fit” gives noisy images
Ill-conditioned problems: small data noise causes large image noise

Solutions:
- Noise-reduction methods
- True regularization methods

Noise-reduction methods
- Modify the *data*
  - Prefilter or “denoise” the sinogram measurements
  - Extrapolate missing (*e.g.*, truncated) data
- Modify an *algorithm* derived for an ill-conditioned problem
  - Stop algorithm before convergence
  - Run to convergence, post-filter
  - Toss in a filtering step every iteration or couple iterations
  - Modify update to “dampen” high-spatial frequencies

Noise-Reduction vs True Regularization

Advantages of noise-reduction methods

- Simplicity (?)
- Familiarity
- Appear less subjective than using penalty functions or priors
- Only fiddle factors are # of iterations, or amount of smoothing
- Resolution/noise tradeoff usually varies with iteration
  (stop when image looks good - in principle)
- Changing post-smoothing does not require re-iterating

Advantages of true regularization methods

- Stability (unique minimizer & convergence $\Longrightarrow$ initialization independence)
- Faster convergence
- Predictability
- Resolution can be made object independent
- Controlled resolution (*e.g.*, spatially uniform, edge preserving)
- Start with reasonable image (*e.g.*, FBP) $\Longrightarrow$ reach solution faster.

True Regularization Methods

Redefine the *problem* to eliminate ill-conditioning,
rather than patching the data or algorithm!

Options

- Use bigger pixels (fewer basis functions)
  - Visually unappealing
  - Can only preserve edges coincident with pixel edges
  - Results become even less invariant to translations
- Method of sieves (constrain image roughness)
  - Condition number for “pre-emission space” can be even worse
  - Lots of iterations
  - Commutability condition rarely holds exactly in practice
  - Degenerates to post-filtering in some cases
- Change cost function by adding a roughness penalty / prior

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \Psi(\boldsymbol{x}), \qquad \Psi(\boldsymbol{x}) = \text{Ł}(\boldsymbol{x}) + \beta R(\boldsymbol{x})$$

  - Disadvantage: apparently subjective choice of penalty
  - Apparent difficulty in choosing penalty parameter(s), *e.g.*, β
    (*cf.* apodizing filter / cutoff frequency in FBP)

Penalty Function Considerations

- Computation
- Algorithm complexity
- Uniqueness of minimizer of $\Psi(\boldsymbol{x})$
- Resolution properties (edge preserving?)
- # of adjustable parameters
- Predictability of properties (resolution and noise)

Choices

- separable vs non-separable
- quadratic vs non-quadratic
- convex vs non-convex

Penalty Functions: Separable vs Non-separable

Separable

- Identity norm: $\mathsf{R}(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}'\boldsymbol{I}\boldsymbol{x} = \sum_{j=1}^{n_\mathrm{p}} x_j^2/2$
  penalizes large values of $\boldsymbol{x}$, but causes "squashing bias"
- Entropy: $\mathsf{R}(\boldsymbol{x}) = \sum_{j=1}^{n_\mathrm{p}} x_j \log x_j$
- Gaussian prior with mean μ_j, variance σ_j^2: $\mathsf{R}(\boldsymbol{x}) = \sum_{j=1}^{n_\mathrm{p}} \frac{(x_j-\mu_j)^2}{2\sigma_j^2}$
- Gamma prior $\mathsf{R}(\boldsymbol{x}) = \sum_{j=1}^{n_\mathrm{p}} \mathsf{p}(x_j, \mu_j, \sigma_j)$ where $\mathsf{p}(x, \mu, \sigma)$ is Gamma pdf

The first two basically keep pixel values from "blowing up."
The last two encourage pixels values to be close to prior means μ_j.

$$\text{General separable form:} \quad \mathsf{R}(\boldsymbol{x}) = \sum_{j=1}^{n_\mathrm{p}} f_j(x_j)$$

Slightly simpler for minimization, but these do not explicitly enforce smoothness.
The simplicity advantage has been overcome in newer algorithms.

Penalty Functions: Separable vs Nonseparable

Nonseparable (partially couple pixel values) to penalize *roughness*

x_1	x_2	x_3
x_4	x_5	

Example

$$\mathsf{R}(\boldsymbol{x}) = (x_2 - x_1)^2 + (x_3 - x_2)^2 + (x_5 - x_4)^2 + (x_4 - x_1)^2 + (x_5 - x_2)^2$$

2	2	2
2	1	

$\mathsf{R}(\boldsymbol{x}) = 1$

3	3	1
2	2	

$\mathsf{R}(\boldsymbol{x}) = 6$

1	3	1
2	2	

$\mathsf{R}(\boldsymbol{x}) = 10$

Rougher images $\Longrightarrow$ larger $\mathsf{R}(\boldsymbol{x})$ values

Roughness Penalty Functions

First-order neighborhood and pairwise pixel differences:

$$\mathsf{R}(\boldsymbol{x}) = \sum_{j=1}^{n_\mathrm{p}} \frac{1}{2} \sum_{k \in \mathcal{N}_j} \psi(x_j - x_k)$$

$\mathcal{N}_j \triangleq$ *neighborhood* of jth pixel (*e.g.*, left, right, up, down)
ψ called the *potential function*

Finite-difference approximation to continuous roughness measure:

$$R(f(\cdot)) = \int \| \nabla f(\vec{r}) \|^2 \, \mathrm{d}\vec{r} = \int \left| \frac{\partial}{\partial x} f(\vec{r}) \right|^2 + \left| \frac{\partial}{\partial y} f(\vec{r}) \right|^2 + \left| \frac{\partial}{\partial z} f(\vec{r}) \right|^2 \mathrm{d}\vec{r}.$$

Second derivatives also useful:
(More choices!)

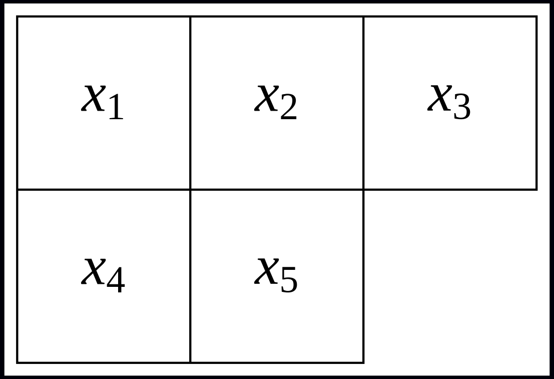
$$\left. \frac{\partial^2}{\partial x^2} f(\vec{r}) \right|_{\vec{r}=\vec{r}_j} \approx f(\vec{r}_{j+1}) - 2 f(\vec{r}_j) + f(\vec{r}_{j-1})$$

$$\mathsf{R}(\boldsymbol{x}) = \sum_{j=1}^{n_\mathrm{p}} \psi(x_{j+1} - 2x_j + x_{j-1}) + \cdots$$

Penalty Functions: General Form

$$\boxed{\mathsf{R}(\boldsymbol{x}) = \sum_k \psi_k([\boldsymbol{C}\boldsymbol{x}]_k)} \quad \text{where} \quad [\boldsymbol{C}\boldsymbol{x}]_k = \sum_{j=1}^{n_\mathrm{p}} c_{kj} x_j$$

Example:



x_1	x_2	x_3
x_4	x_5	

$$\boldsymbol{C}\boldsymbol{x} = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 \\ -1 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} x_2 - x_1 \\ x_3 - x_2 \\ x_5 - x_4 \\ x_4 - x_1 \\ x_5 - x_2 \end{bmatrix}$$

$$\begin{aligned} \mathsf{R}(\boldsymbol{x}) &= \sum_{k=1}^{5} \psi_k([\boldsymbol{C}\boldsymbol{x}]_k) \\ &= \psi_1(x_2 - x_1) + \psi_2(x_3 - x_2) + \psi_3(x_5 - x_4) + \psi_4(x_4 - x_1) + \psi_5(x_5 - x_2) \end{aligned}$$

Penalty Functions: Quadratic vs Nonquadratic

$$\mathsf{R}(\boldsymbol{x}) = \sum_k \psi_k([\boldsymbol{C}\boldsymbol{x}]_k)$$

Quadratic ψ_k

If $\psi_k(t) = t^2/2$, then $\mathsf{R}(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}'\boldsymbol{C}'\boldsymbol{C}\boldsymbol{x}$, a quadratic form.

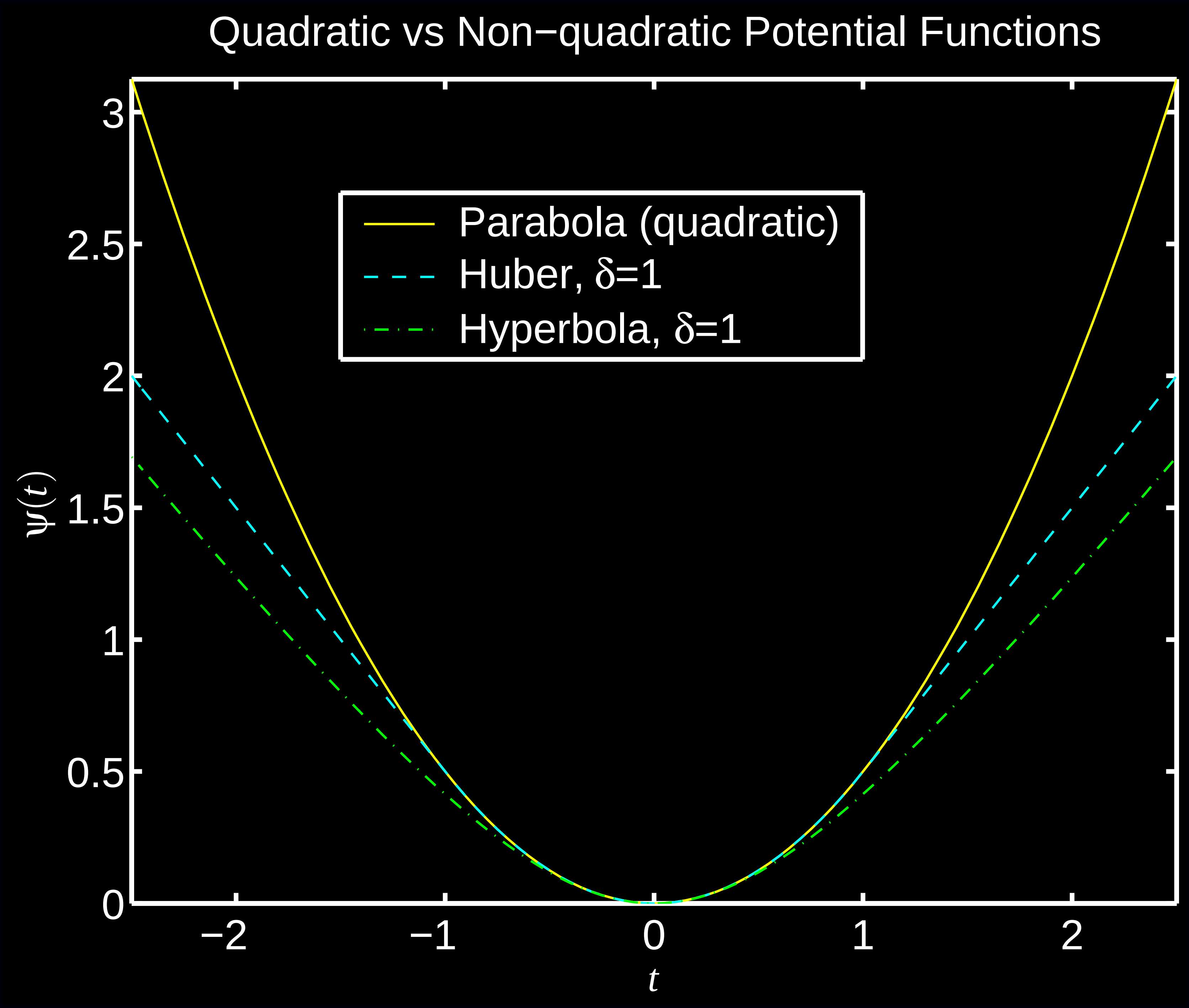
- Simpler optimization
- Global smoothing

Nonquadratic ψ_k

- Edge preserving
- More complicated optimization. (This is essentially solved in convex case.)
- Unusual noise properties
- Analysis/prediction of resolution and noise properties is difficult
- More adjustable parameters (*e.g.*, δ)

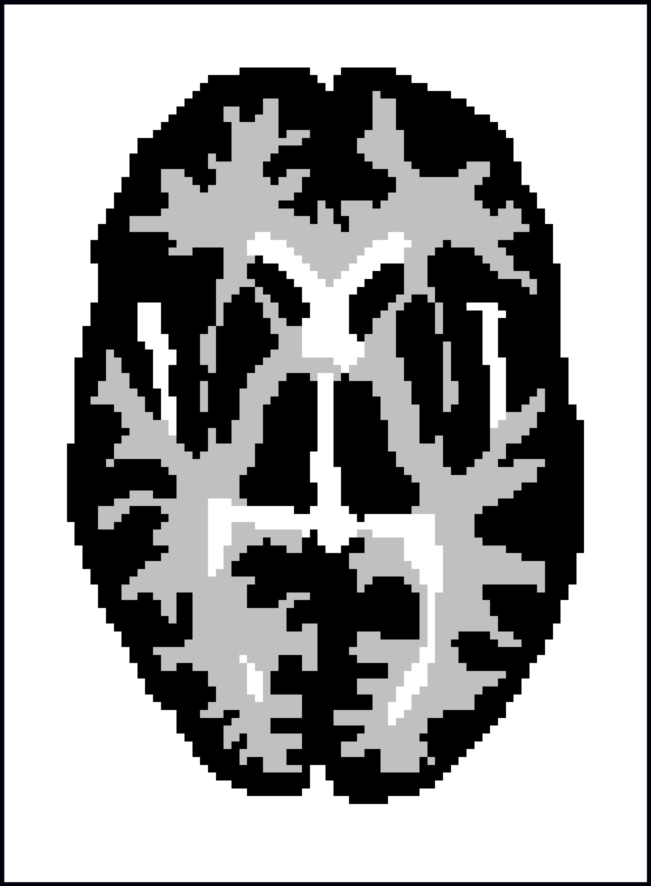
Example: Huber function. $\psi(t) \triangleq \begin{cases} t^2/2, & |t| \leq \delta \\ \delta|t| - \delta^2/2, & |t| > \delta \end{cases}$

Example: Hyperbola function. $\psi(t) \triangleq \delta^2 \left(\sqrt{1 + (t/\delta)^2} - 1\right)$

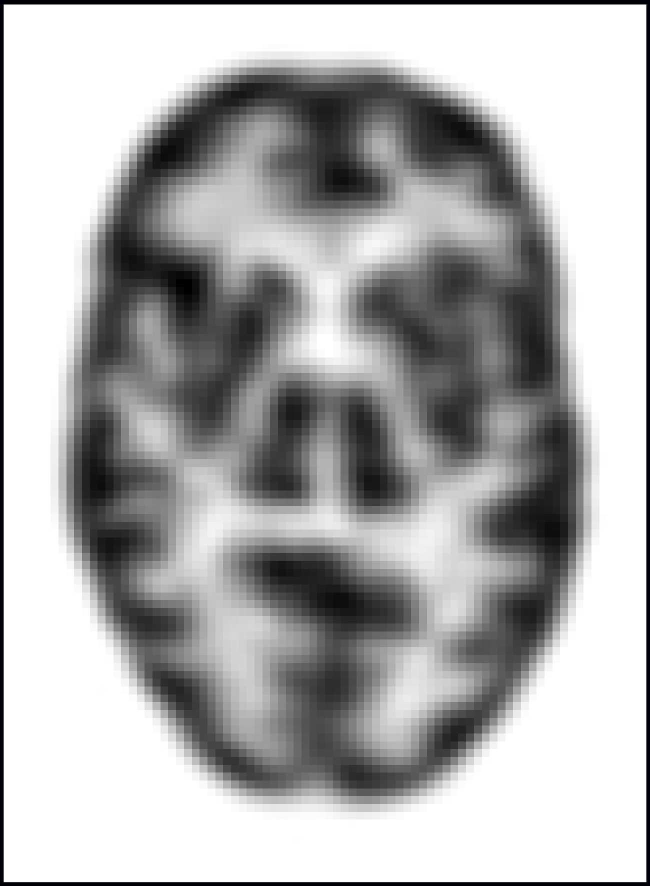


Lower cost for large differences $\Longrightarrow$ edge preservation

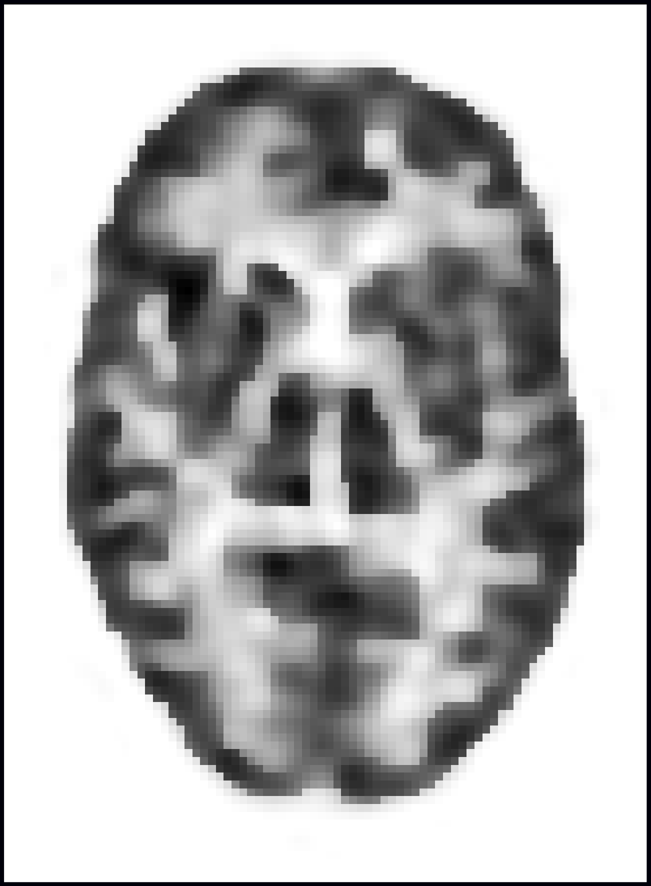
Edge-Preserving Reconstruction Example



Phantom

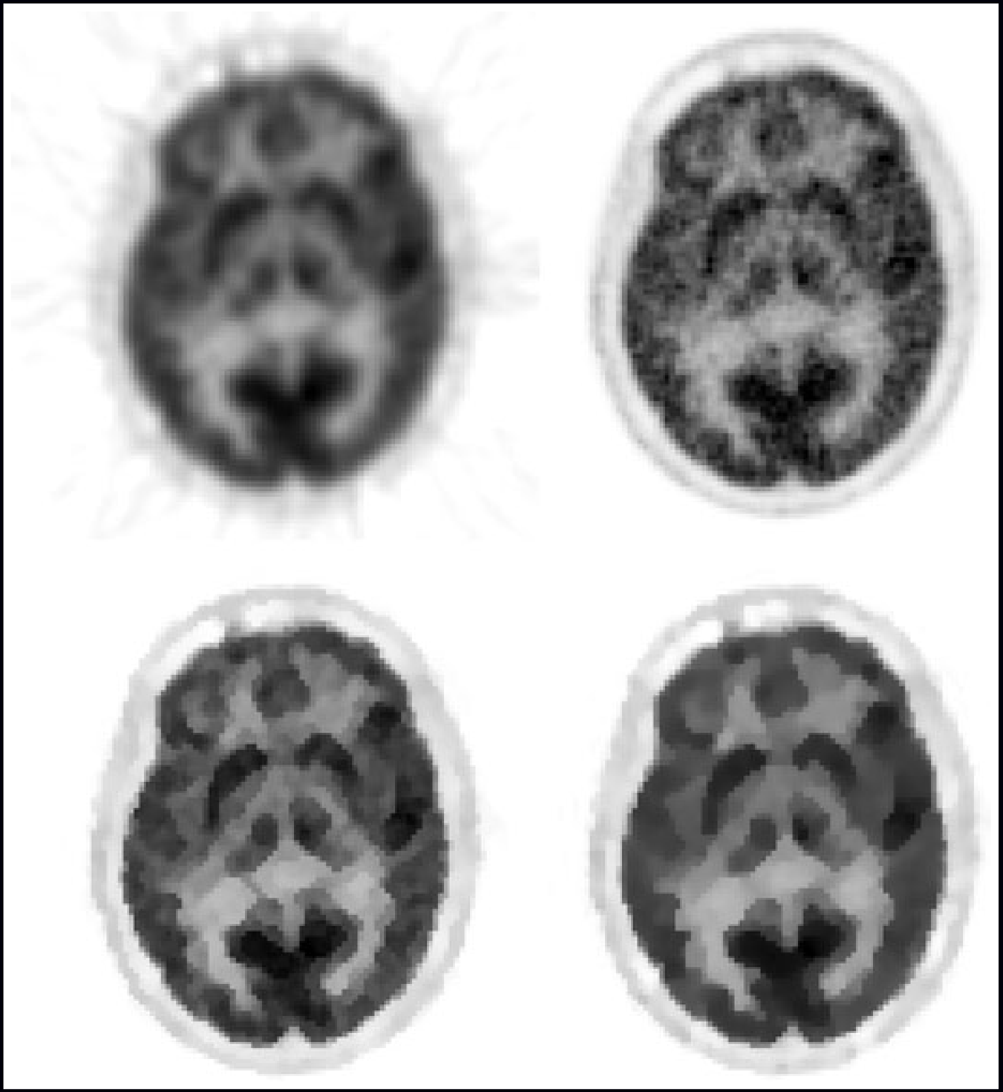


Quadratic Penalty



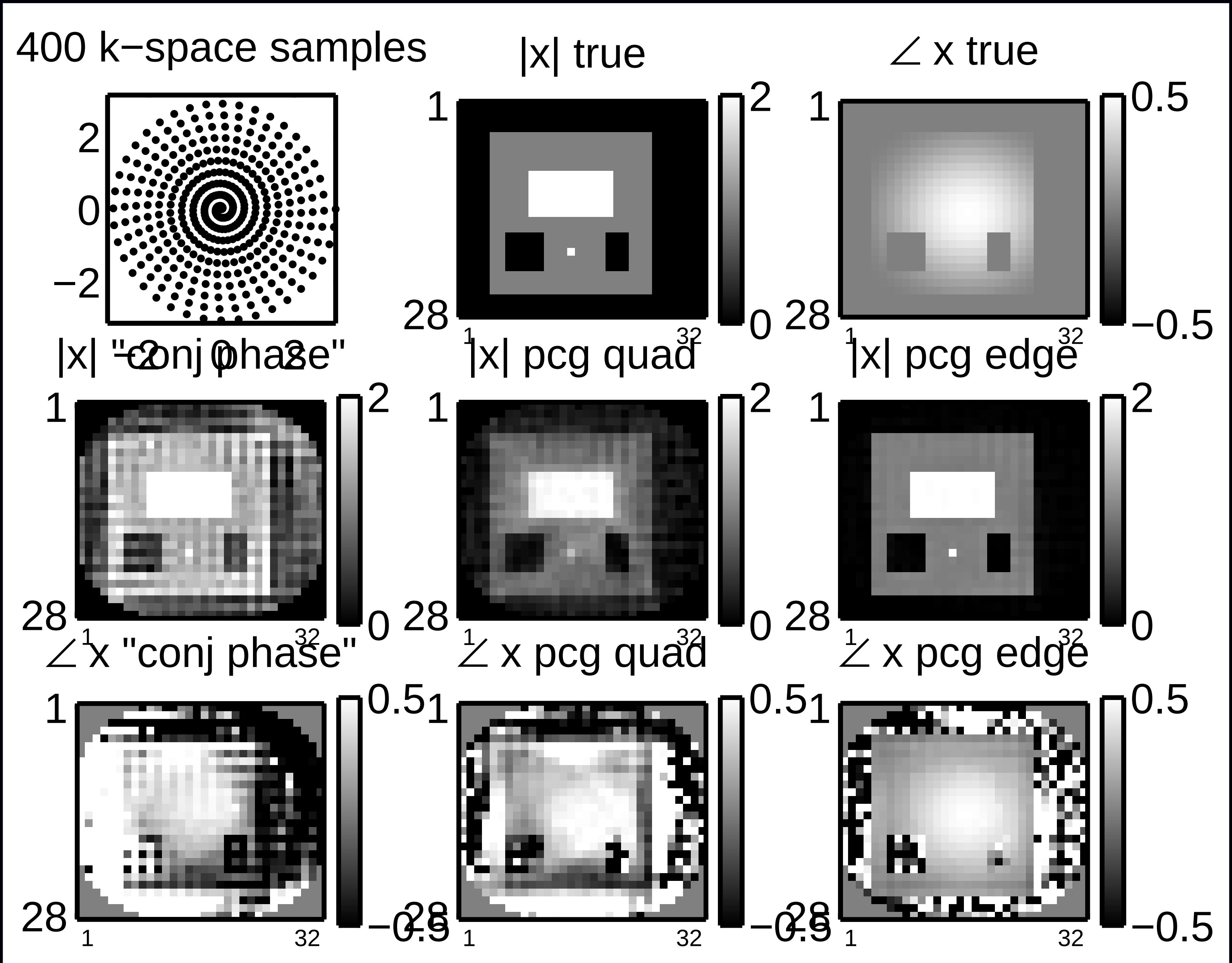
Huber Penalty

More “Edge Preserving” Regularization



Chlewicki *et al.*, PMB, Oct. 2004: “Noise reduction and convergence of Bayesian algorithms with blobs based on the Huber function and median root prior”

Piecewise Constant “Cartoon” Objects



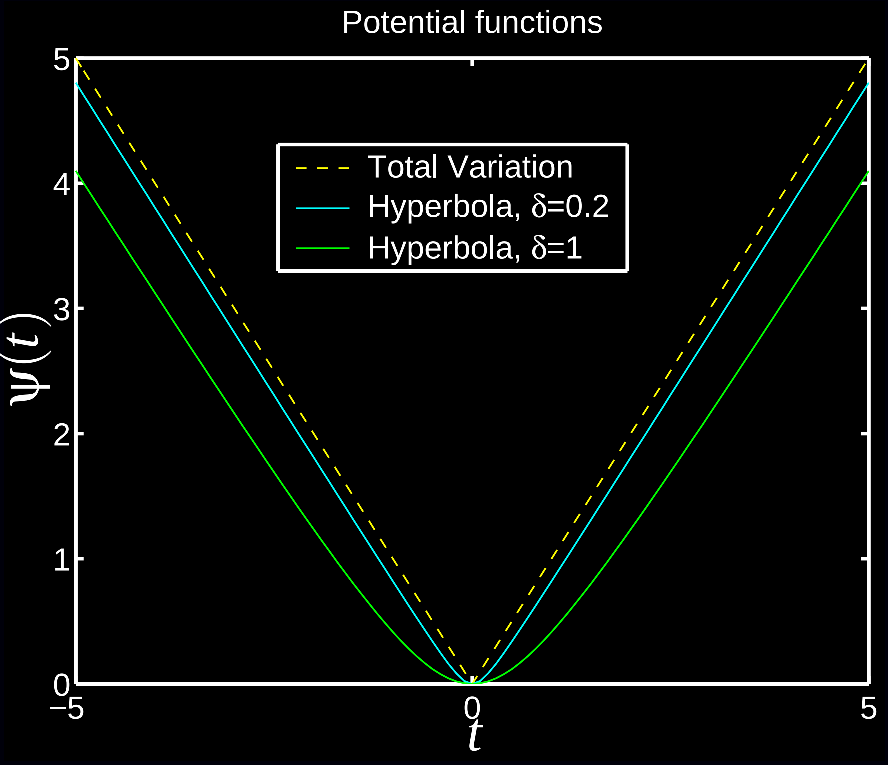
Total Variation Regularization

Non-quadratic roughness penalty:

$$\int \|\nabla f(\vec{r})\| \, \mathrm{d}\vec{r} \approx \sum_k |[\boldsymbol{Cx}]_k|$$

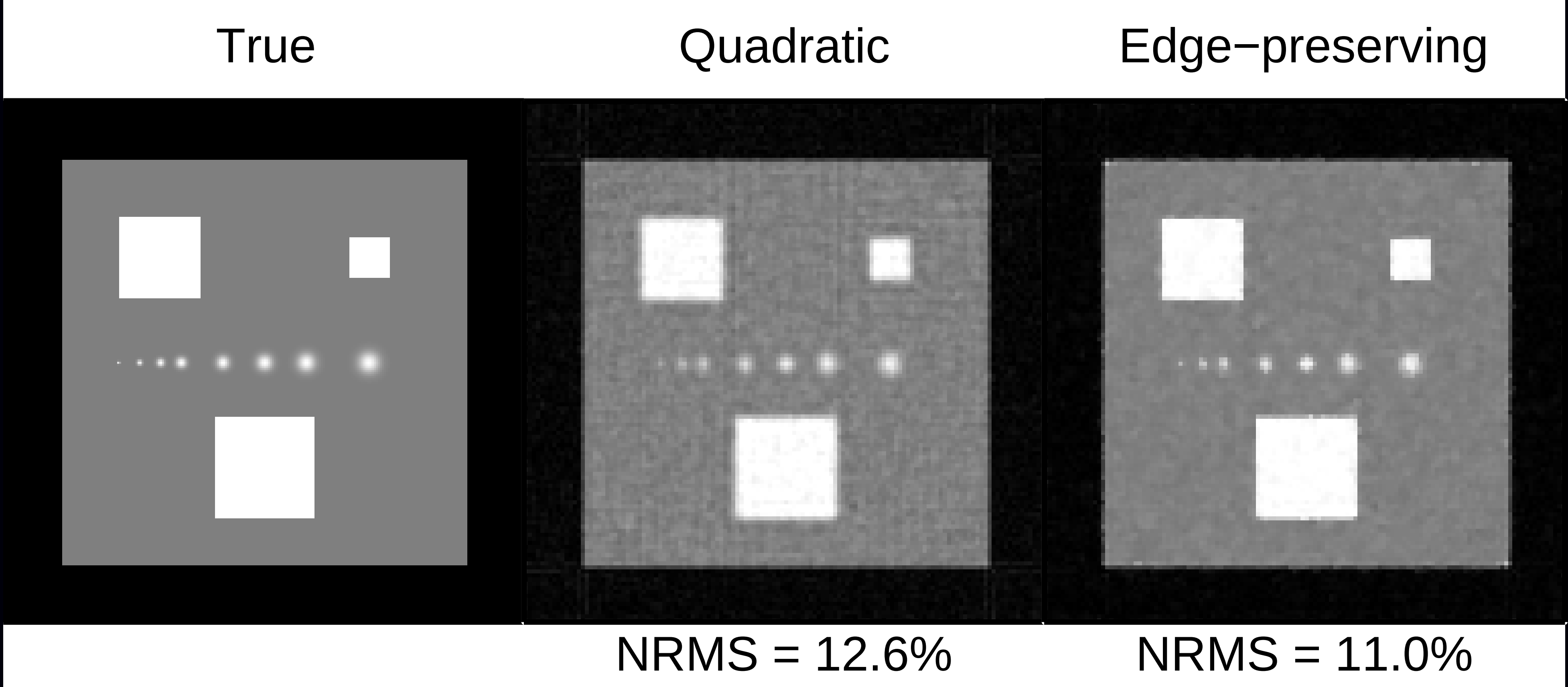
Uses *magnitude* instead of *squared magnitude* of gradient.

Problem: $|\cdot|$ is not differentiable.
Practical solution: $|t| \approx \delta\left(\sqrt{1+(t/\delta)^2}-1\right)$ (hyperbola!)



Total Variation Example

MRI with under-sampled radial acquisition



True

Quadratic

Edge−preserving

NRMS = 12.6%

NRMS = 11.0%

Compressed Sensing

aka compressive sampling or sparsity regularization

Idea: find a basis $\boldsymbol{B}$ for representing $\boldsymbol{x}$ in terms of coefficients $\boldsymbol{\theta}$:
$\boxed{\boldsymbol{x} = \boldsymbol{B}\boldsymbol{\theta},}$ *where only a "small number" of* θ_j *values are nonzero*.

Previous cost function: $\Psi(\boldsymbol{x}) = \text{DataMismatch}(\boldsymbol{y}, \boldsymbol{A}\boldsymbol{x}) + \beta\,\text{Roughness}(\boldsymbol{x})$

New cost function with sparsity regularization:

$$\Psi(\boldsymbol{\theta}) = \text{DataMismatch}(\boldsymbol{y}, \boldsymbol{A}\boldsymbol{B}\boldsymbol{\theta}) + \beta \left\|\boldsymbol{\theta}\right\|_0$$

Recall:

$$\begin{aligned}
\left\|\boldsymbol{\theta}\right\|_p &\triangleq \left(\sum_j |\theta_j|^p\right)^{1/p} \\
\left\|\boldsymbol{\theta}\right\|_\infty &\triangleq \lim_{p\to\infty} \left\|\boldsymbol{\theta}\right\|_p = \max_j |\theta_j| \\
\left\|\boldsymbol{\theta}\right\|_0 &\triangleq \lim_{p\to 0} \left\|\boldsymbol{\theta}\right\|_p^p = \sum_j 1_{\{\theta_j \neq 0\}}
\end{aligned}$$

(not a norm in the Banach sense)

Because $\left\|\boldsymbol{\theta}\right\|_0$ is nonconvex, it usually is replaced with $\left\|\boldsymbol{\theta}\right\|_1$.
Because $\left\|\boldsymbol{\theta}\right\|_1$ is nondifferentiable, the corner is often rounded (hyperbola).
If $\boldsymbol{B}$ is the Harr wavelet basis, then $\left\|\boldsymbol{\theta}\right\|_1 = \left\|\boldsymbol{B}^{-1}\boldsymbol{x}\right\|_1$ is similar to TV regularization.

For certain types of under-sampled measurements $\boldsymbol{A}$, "good" reconstructions are possible! Example: radial k-space sampling for Shepp-Logan.

Penalty Functions: Convex vs Nonconvex

Convex

- Easier to optimize
- Guaranteed unique minimizer of Ψ (for convex negative log-likelihood)

Nonconvex

- Greater degree of edge preservation
- Nice images for piecewise-constant phantoms!
- Even more unusual noise properties
- Multiple extrema
- More complicated optimization (simulated / deterministic annealing)
- Estimator $\hat{\boldsymbol{x}}$ becomes a discontinuous function of data $\boldsymbol{Y}$

Nonconvex examples

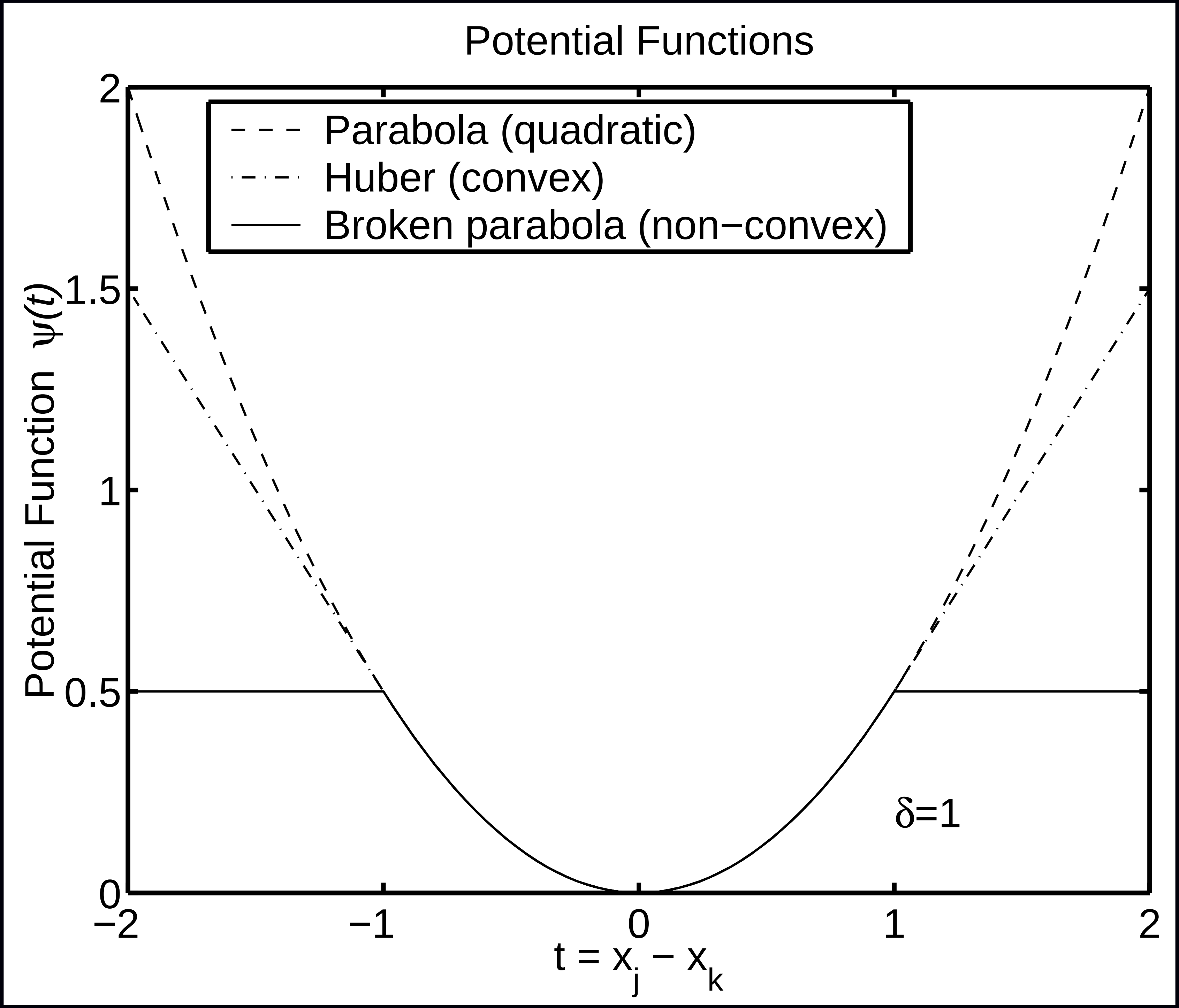
- “broken parabola”

$$\psi(t) = \min(t^2, \ t_{\max}^2)$$

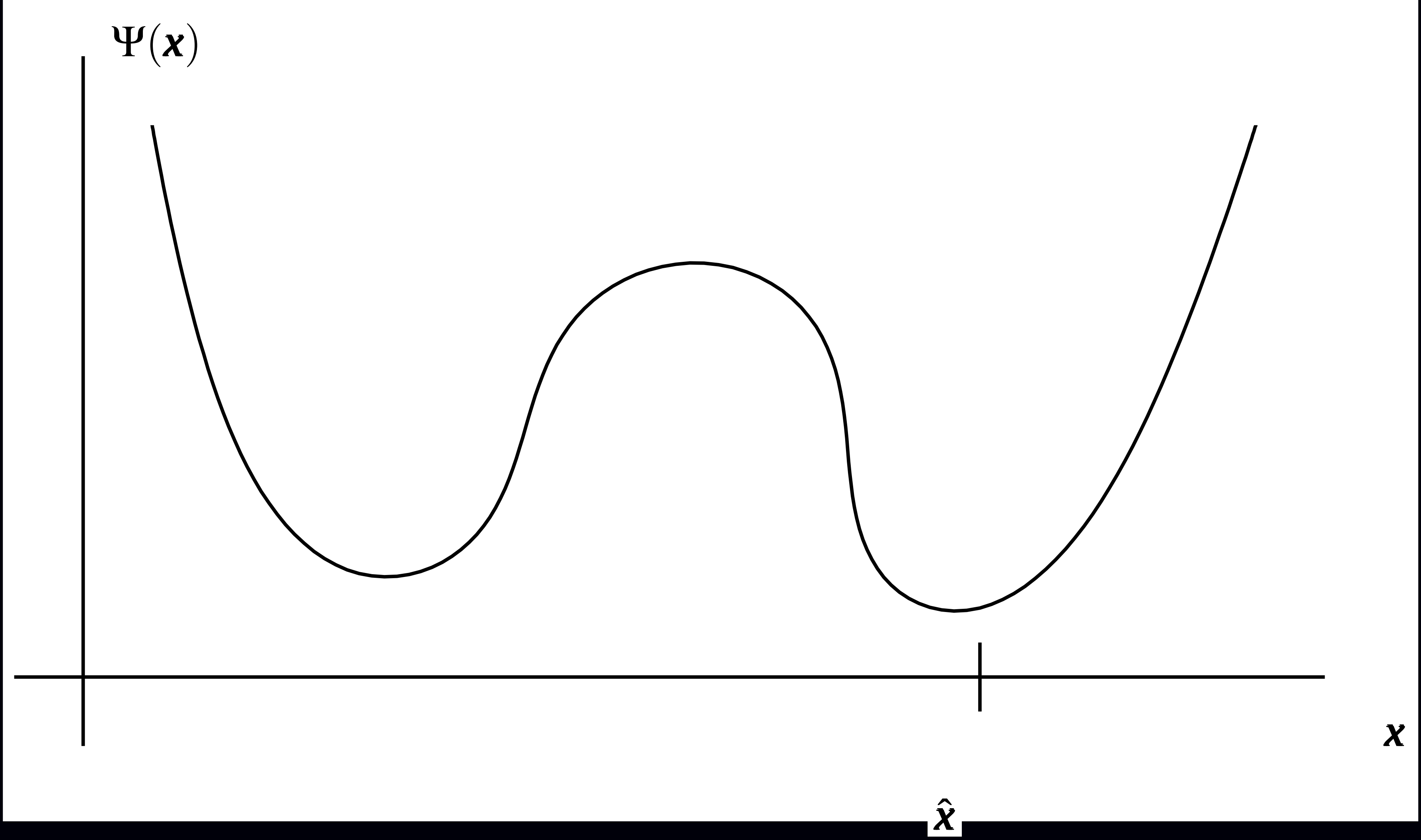
- true median root prior:

$$\mathrm{R}(\boldsymbol{x}) = \sum_{j=1}^{n_\mathrm{p}} \frac{(x_j - \mathrm{median}_j(\boldsymbol{x}))^2}{\mathrm{median}_j(\boldsymbol{x})} \quad \text{where} \quad \mathrm{median}_j(\boldsymbol{x}) \quad \text{is local median}$$

Exception: orthonormal wavelet threshold *denoising* via nonconvex potentials!

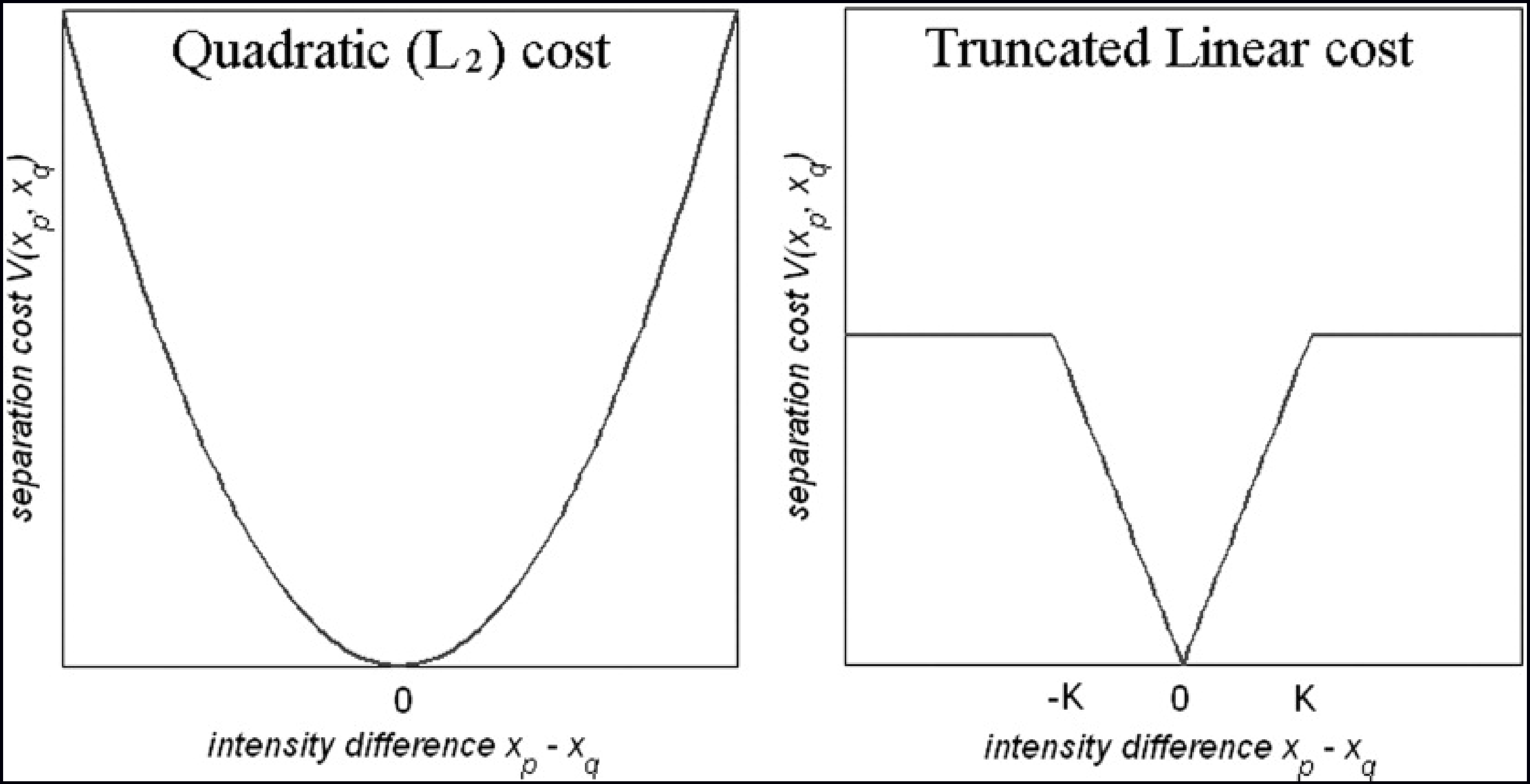

Potential Functions
Parabola (quadratic)
Huber (convex)
Broken parabola (non−convex)
Potential Function ψ(t)
t = x_j − x_k
δ=1
0
0.5
1
1.5
2
−2
−1
0
1
2

Local Extrema and Discontinuous Estimators



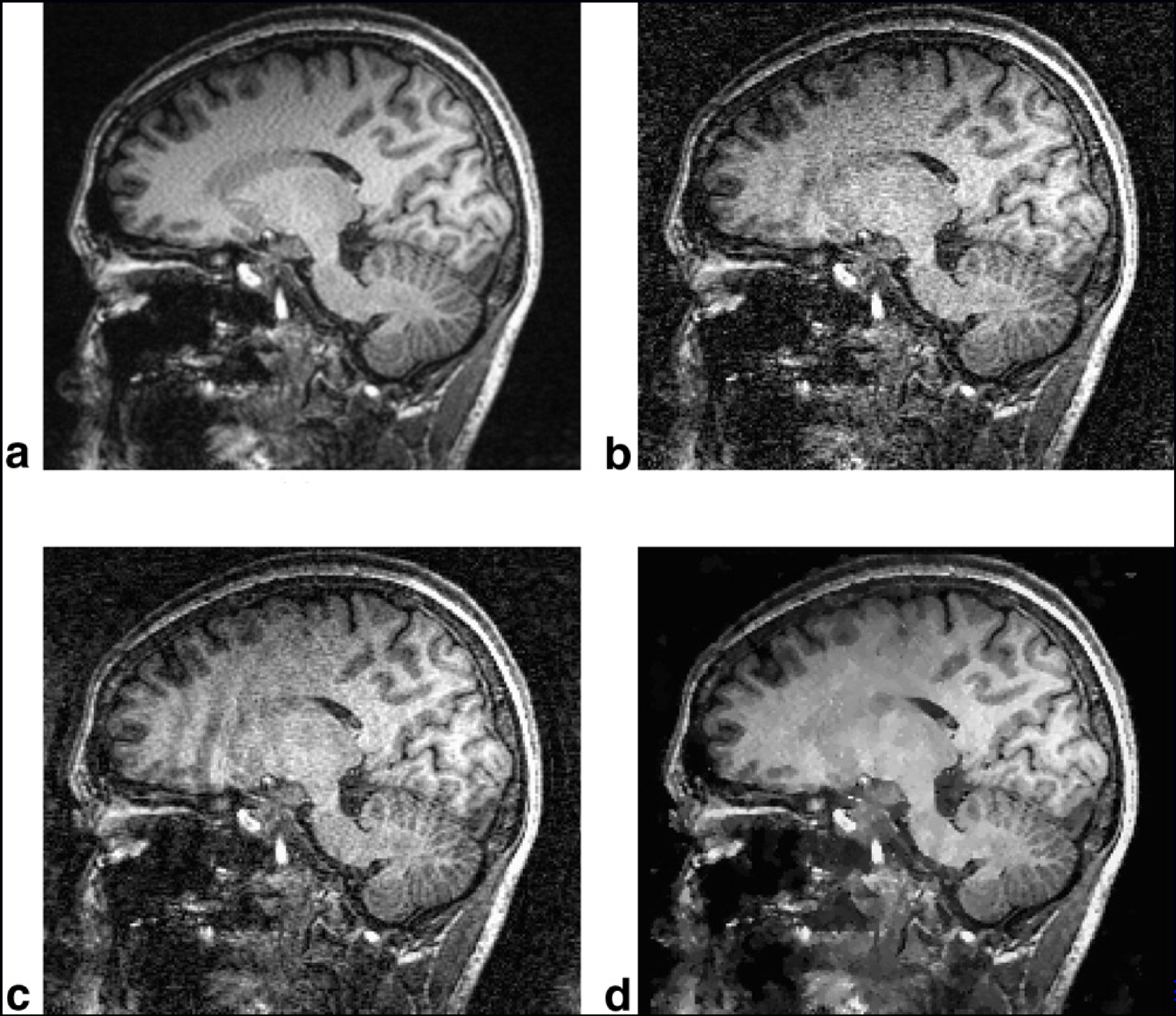
Small change in data $\Longrightarrow$ large change in minimizer $\hat{\boldsymbol{x}}$.
Using convex penalty functions obviates this problem.

Nonconvex Edge-Preserving Regularization



Raj *et al.*, MRM, Jan. 2007

Applied to MR parallel imaging (multiple receive coils)


a
b
c
d

Augmented Regularization Functions

Replace roughness penalty R($\boldsymbol{x}$) with $R(\boldsymbol{x}|\boldsymbol{b}) + \alpha R(\boldsymbol{b})$,
where the elements of $\boldsymbol{b}$ (often binary) indicate boundary locations.

- Line-site methods
- Level-set methods

Joint estimation problem:

$$(\hat{\boldsymbol{x}}, \hat{\boldsymbol{b}}) = \arg\min_{\boldsymbol{x},\boldsymbol{b}} \Psi(\boldsymbol{x}, \boldsymbol{b}), \qquad \Psi(\boldsymbol{x}, \boldsymbol{b}) = \text{Ł}(\boldsymbol{x}; \boldsymbol{y}) + \beta R(\boldsymbol{x}|\boldsymbol{b}) + \alpha R(\boldsymbol{b}).$$

Example: b_{jk} indicates the presence of edge between pixels j and k:

$$R(\boldsymbol{x}|\boldsymbol{b}) = \sum_{j=1}^{n_\mathrm{p}} \sum_{k \in \mathcal{N}_j} (1 - b_{jk}) \frac{1}{2} (x_j - x_k)^2$$

Penalty to discourage too many edges (*e.g.*):

$$R(\boldsymbol{b}) = \sum_{jk} b_{jk}.$$

- Can encourage local edge continuity
- May require annealing methods for minimization

Modified Penalty Functions

$$\mathrm{R}(\boldsymbol{x}) = \sum_{j=1}^{n_\mathrm{p}} \frac{1}{2} \sum_{k \in \mathcal{N}_j} w_{jk} \psi(x_j - x_k)$$

Adjust weights $\{w_{jk}\}$ to

- Control resolution properties
- Incorporate anatomical side information (MR/CT)
  (avoid smoothing across anatomical boundaries)

Recommendations

- Emission tomography:
  - ○ Begin with quadratic (nonseparable) penalty functions
  - ○ Consider modified penalty for resolution control and choice of β
  - ○ Use modest regularization and post-filter more if desired
- Transmission tomography (attenuation maps), X-ray CT
  - ○ consider convex nonquadratic (*e.g.*, Huber) penalty functions
  - ○ choose δ based on attenuation map units (water, bone, etc.)
  - ○ choice of regularization parameter β remains nontrivial,
    learn appropriate values by experience for given study type

Choice 4.3: Constraints

- Nonnegativity
- Known support
- Count preserving
- Upper bounds on values
  e.g., maximum μ of attenuation map in transmission case

Considerations

- Algorithm complexity
- Computation
- Convergence rate
- Bias (in low-count regions)
- . . .

Open Problems

- Performance prediction for nonquadratic regularization
- Effect of nonquadratic regularization on detection tasks
- Choice of regularization parameters for nonquadratic regularization

Summary

- 1. Object parameterization: function $f(\vec{r})$ vs vector $\boldsymbol{x}$
- 2. System physical model: $s_i(\vec{r})$
- 3. Measurement statistical model $Y_i \sim \boxed{?}$
- 4. Cost function: data-mismatch / regularization / constraints

$$\boxed{\textbf{Reconstruction Method} \triangleq \textbf{Cost Function + Algorithm}}$$

Naming convention: “criterion”-“algorithm”:

- ML-EM, MAP-OSL, PL-SAGE, PWLS+SOR, PWLS-CG, . . .

Part 3. Algorithms

Method = Cost Function + Algorithm

Outline

- Ideal algorithm
- Classical general-purpose algorithms
- Considerations:
  - nonnegativity
  - parallelization
  - convergence rate
  - monotonicity
- Algorithms tailored to cost functions for imaging
  - Optimization transfer
  - EM-type methods
  - Poisson emission problem
  - Poisson transmission problem
- Ordered-subsets / block-iterative algorithms
  - Recent convergent versions (relaxation, incrementalism)

Why iterative algorithms?

- For nonquadratic Ψ, no closed-form solution for minimizer.
- For quadratic Ψ with nonnegativity constraints, no closed-form solution.
- For quadratic Ψ without constraints, closed-form solutions:

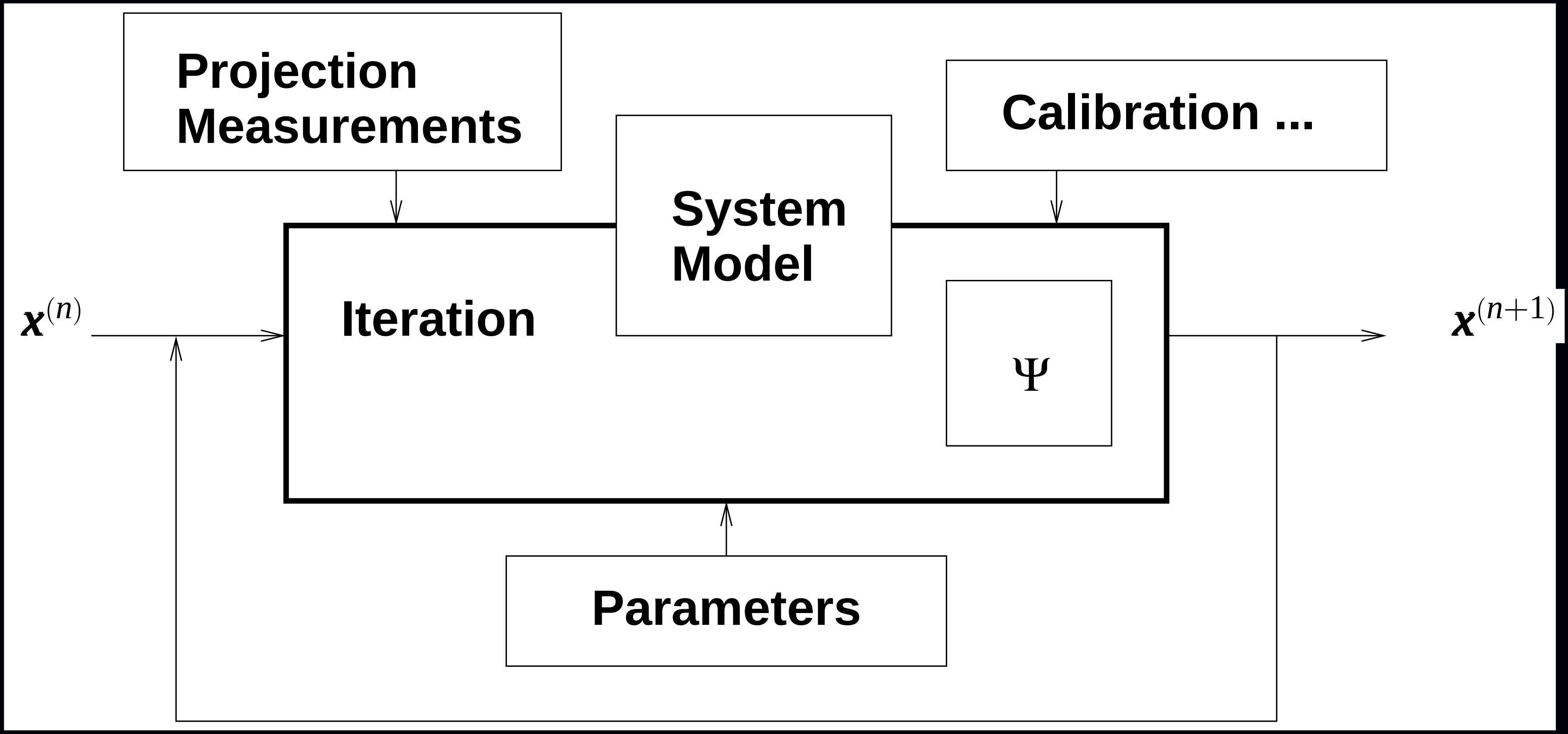
$$\text{PWLS:} \quad \hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2_{\boldsymbol{W}^{1/2}} + \boldsymbol{x}'\boldsymbol{R}\boldsymbol{x} = [\boldsymbol{A}'\boldsymbol{W}\boldsymbol{A} + \boldsymbol{R}]^{-1}\boldsymbol{A}'\boldsymbol{W}\boldsymbol{y}$$

$$\text{OLS:} \quad \hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2 = [\boldsymbol{A}'\boldsymbol{A}]^{-1}\boldsymbol{A}'\boldsymbol{y}$$

Impractical (memory and computation) for realistic problem sizes.
$\boldsymbol{A}$ is sparse, but $\boldsymbol{A}'\boldsymbol{A}$ is not.

All algorithms are imperfect. No single best solution.

General Iteration



Deterministic iterative mapping: $\boldsymbol{x}^{(n+1)} = \mathcal{M}\left(\boldsymbol{x}^{(n)}\right)$

Ideal Algorithm

$$\boldsymbol{x}^\star \triangleq \arg\min_{\boldsymbol{x} \geq \mathbf{0}} \Psi(\boldsymbol{x}) \qquad \text{(global minimizer)}$$

Properties

stable and convergent	$\{\boldsymbol{x}^{(n)}\}$ converges to $\boldsymbol{x}^\star$ if run indefinitely
converges quickly	$\{\boldsymbol{x}^{(n)}\}$ gets “close” to $\boldsymbol{x}^\star$ in just a few iterations
globally convergent	$\lim_n \boldsymbol{x}^{(n)}$ independent of starting image $\boldsymbol{x}^{(0)}$
fast	requires minimal computation per iteration
robust	insensitive to finite numerical precision
user friendly	nothing to adjust (*e.g.*, acceleration factors)
parallelizable	(when necessary)
simple	easy to program and debug
flexible	accommodates any type of system model

(matrix stored by row or column, or factored, or projector/backprojector)

Choices: forgo one or more of the above

Classic Algorithms

Non-gradient based

- Exhaustive search
- Nelder-Mead simplex (amoeba)

Converge very slowly, but work with nondifferentiable cost functions.

Gradient based

- Gradient descent

$$\boldsymbol{x}^{(n+1)} \triangleq \boldsymbol{x}^{(n)} - \alpha \nabla \Psi\left(\boldsymbol{x}^{(n)}\right)$$

  Choosing α to ensure convergence is nontrivial.

- Steepest descent

$$\boldsymbol{x}^{(n+1)} \triangleq \boldsymbol{x}^{(n)} - \alpha_n \nabla \Psi\left(\boldsymbol{x}^{(n)}\right) \quad \text{where} \quad \alpha_n \triangleq \arg\min_{\alpha} \Psi\left(\boldsymbol{x}^{(n)} - \alpha \nabla \Psi\left(\boldsymbol{x}^{(n)}\right)\right)$$

  Computing stepsize α_n can be expensive or inconvenient.

Limitations

- Converge slowly.
- Do not easily accommodate nonnegativity constraint.

Gradients & Nonnegativity - A Mixed Blessing

Unconstrained optimization of differentiable cost functions:

$$\nabla \Psi(\boldsymbol{x}) = \mathbf{0} \text{ when } \boldsymbol{x} = \boldsymbol{x}^\star$$

- A necessary condition always.
- A sufficient condition for strictly convex cost functions.
- Iterations search for zero of gradient.

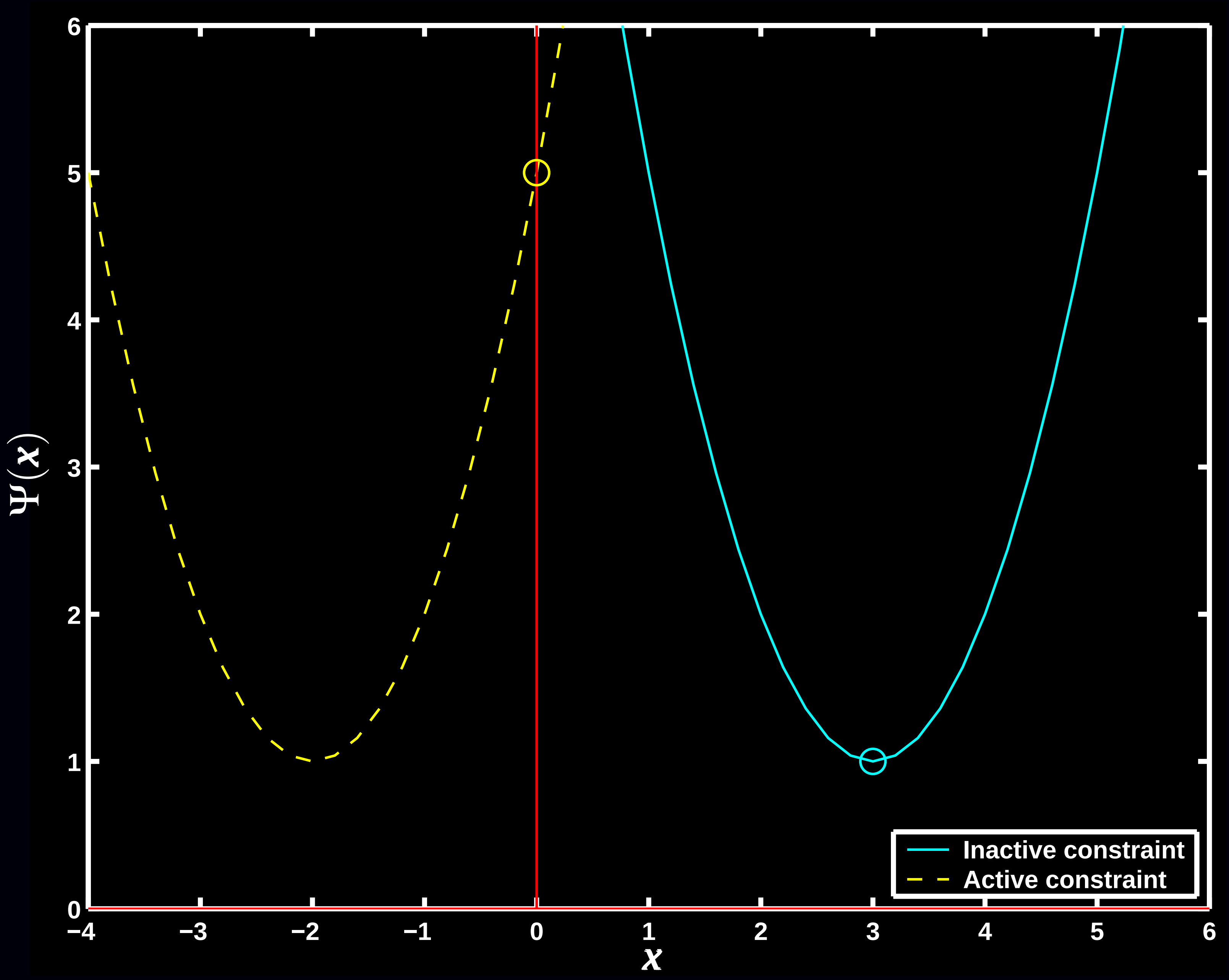
Nonnegativity-constrained minimization:

Karush-Kuhn-Tucker conditions

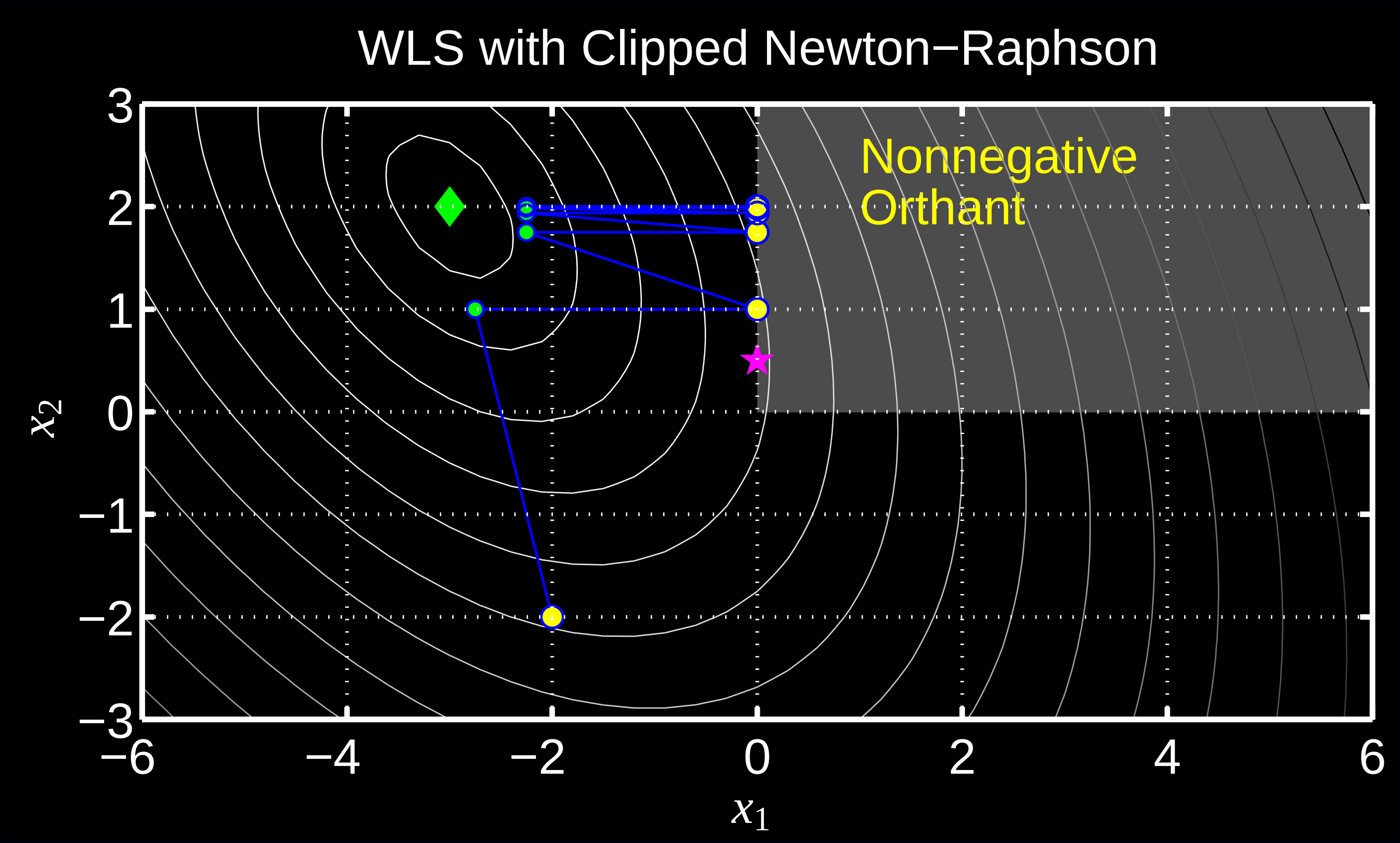
$$\left.\frac{\partial}{\partial x_j}\Psi(\boldsymbol{x})\right|_{\boldsymbol{x}=\boldsymbol{x}^\star} \text{ is } \begin{cases} = 0, & x_j^\star > 0 \\ \geq 0, & x_j^\star = 0 \end{cases}$$

- A necessary condition always.
- A sufficient condition for strictly convex cost functions.
- Iterations search for ???
- $0 = x_j^\star \frac{\partial}{\partial x_j}\Psi(\boldsymbol{x}^\star)$ is a necessary condition, but never sufficient condition.

Karush-Kuhn-Tucker Illustrated



Why Not Clip Negatives?



Newton-Raphson with negatives set to zero each iteration.
Fixed-point of iteration is not the constrained minimizer!

Newton-Raphson Algorithm

$$\boxed{\boldsymbol{x}^{(n+1)} = \boldsymbol{x}^{(n)} - \left[\nabla^2 \Psi\left(\boldsymbol{x}^{(n)}\right)\right]^{-1} \nabla \Psi\left(\boldsymbol{x}^{(n)}\right)}$$

Advantage:

- Super-linear convergence rate (if convergent)

Disadvantages:

- Requires twice-differentiable Ψ
- Not guaranteed to converge
- Not guaranteed to monotonically decrease Ψ
- Does not enforce nonnegativity constraint
- Computing Hessian $\nabla^2 \Psi$ often expensive
- Impractical for image recovery due to matrix inverse

General purpose remedy: bound-constrained Quasi-Newton algorithms

Newton’s Quadratic Approximation

2nd-order Taylor series:

$$\Psi(\boldsymbol{x}) \approx \phi(\boldsymbol{x};\boldsymbol{x}^{(n)}) \triangleq \Psi\left(\boldsymbol{x}^{(n)}\right) + \nabla\Psi\left(\boldsymbol{x}^{(n)}\right)(\boldsymbol{x}-\boldsymbol{x}^{(n)}) + \frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}^{(n)})^T \nabla^2\Psi\left(\boldsymbol{x}^{(n)}\right)(\boldsymbol{x}-\boldsymbol{x}^{(n)})$$

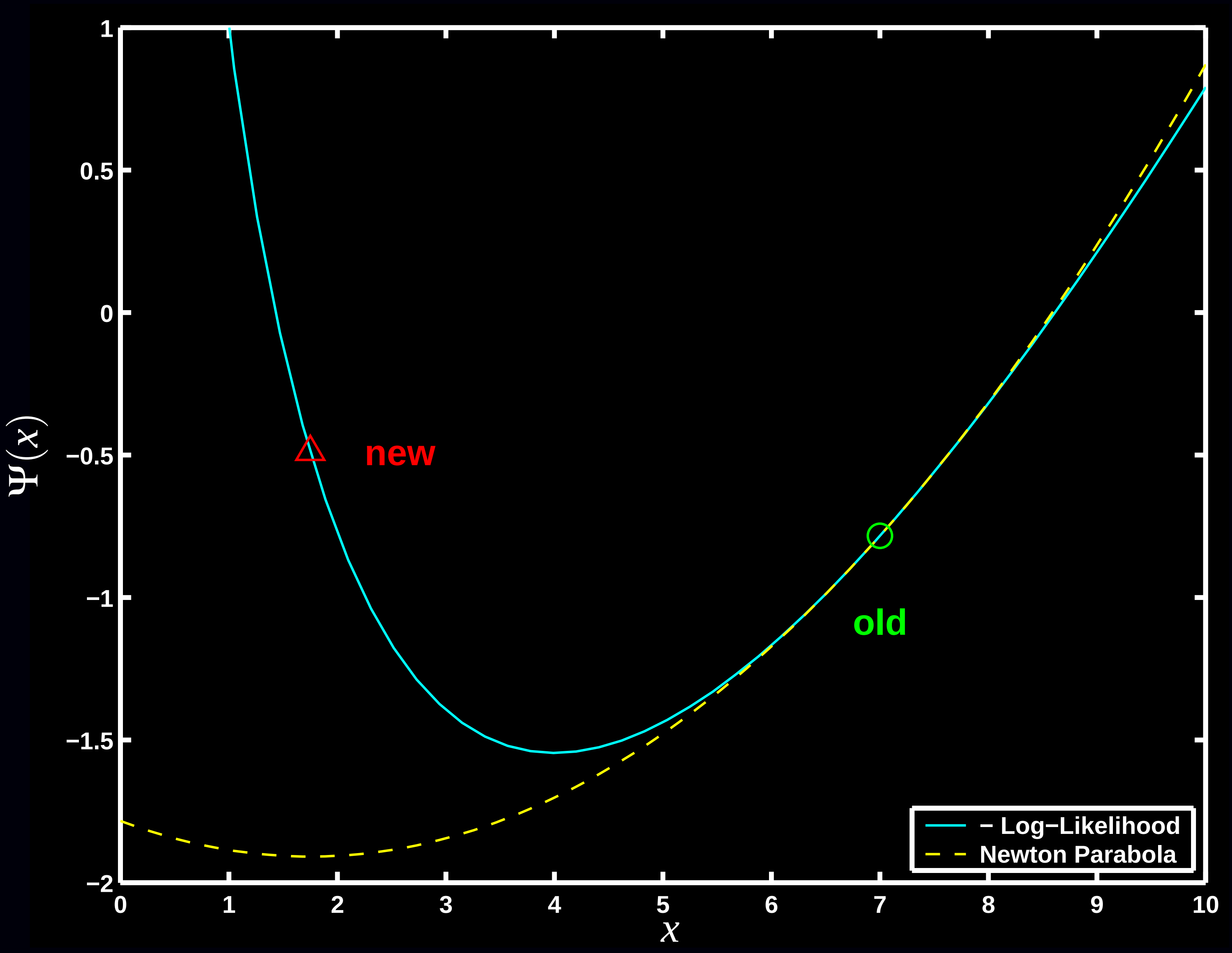
Set $\boldsymbol{x}^{(n+1)}$ to the (“easily” found) minimizer of this quadratic approximation:

$$\begin{aligned} \boldsymbol{x}^{(n+1)} &\triangleq \arg\min_{\boldsymbol{x}} \phi(\boldsymbol{x};\boldsymbol{x}^{(n)}) \\ &= \boldsymbol{x}^{(n)} - [\nabla^2\Psi\left(\boldsymbol{x}^{(n)}\right)]^{-1}\nabla\Psi\left(\boldsymbol{x}^{(n)}\right) \end{aligned}$$

Can be nonmonotone for Poisson emission tomography log-likelihood, even for a single pixel and single ray:

$$\Psi(x) = (x+r) - y\log(x+r)\,.$$

Nonmonotonicity of Newton-Raphson



Consideration: Monotonicity

An algorithm is monotonic if

$$\Psi\left(\boldsymbol{x}^{(n+1)}\right) \leq \Psi\left(\boldsymbol{x}^{(n)}\right), \quad \forall \boldsymbol{x}^{(n)}.$$

Three categories of algorithms:

- Nonmonotonic (or unknown)
- Forced monotonic (*e.g.*, by line search)
- Intrinsically monotonic (by design, simplest to implement)

Forced monotonicity

Most nonmonotonic algorithms can be converted to forced monotonic algorithms by adding a line-search step:

$$\boldsymbol{x}^{\text{temp}} \triangleq \mathcal{M}\left(\boldsymbol{x}^{(n)}\right), \quad \boldsymbol{d} = \boldsymbol{x}^{\text{temp}} - \boldsymbol{x}^{(n)}$$

$$\boldsymbol{x}^{(n+1)} \triangleq \boldsymbol{x}^{(n)} - \alpha_n \boldsymbol{d}^{(n)} \text{ where } \alpha_n \triangleq \arg\min_{\alpha} \Psi\left(\boldsymbol{x}^{(n)} - \alpha \boldsymbol{d}^{(n)}\right)$$

Inconvenient, sometimes expensive, nonnegativity problematic.

Conjugate Gradient (CG) Algorithm

Advantages:

- Fast converging (if suitably preconditioned) (in unconstrained case)
- Monotonic (forced by line search in nonquadratic case)
- Global convergence (unconstrained case)
- Flexible use of system matrix $\boldsymbol{A}$ and tricks
- Easy to implement in unconstrained quadratic case
- Highly parallelizable

Disadvantages:

- Nonnegativity constraint awkward (slows convergence?)
- Line-search somewhat awkward in nonquadratic cases
- Possible need to “restart” after many iterations

Highly recommended for unconstrained quadratic problems (*e.g.*, PWLS without nonnegativity). Useful (but perhaps not ideal) for Poisson case too.

Consideration: Parallelization

Simultaneous (fully parallelizable)
update all pixels simultaneously using all data
EM, Conjugate gradient, ISRA, OSL, SIRT, MART, ...

Block iterative (ordered subsets)
update (nearly) all pixels using one subset of the data at a time
OSEM, RBBI, ...

Row action
update many pixels using a single ray at a time
ART, RAMLA

Pixel grouped (multiple column action)
update some (but not all) pixels simultaneously a time, using all data
Grouped coordinate descent, multi-pixel SAGE
(Perhaps the most nontrivial to implement)

Sequential (column action)
update one pixel at a time, using all (relevant) data
Coordinate descent, SAGE

Coordinate Descent Algorithm

aka Gauss-Siedel, successive over-relaxation (SOR), iterated conditional modes (ICM)

Update one pixel at a time, holding others fixed to their most recent values:

$$x_j^{\text{new}} = \operatorname*{arg\,min}_{x_j \geq 0} \Psi\left(x_1^{\text{new}}, \ldots, x_{j-1}^{\text{new}}, x_j, x_{j+1}^{\text{old}}, \ldots, x_{n_{\text{p}}}^{\text{old}}\right), \qquad j = 1, \ldots, n_{\text{p}}$$

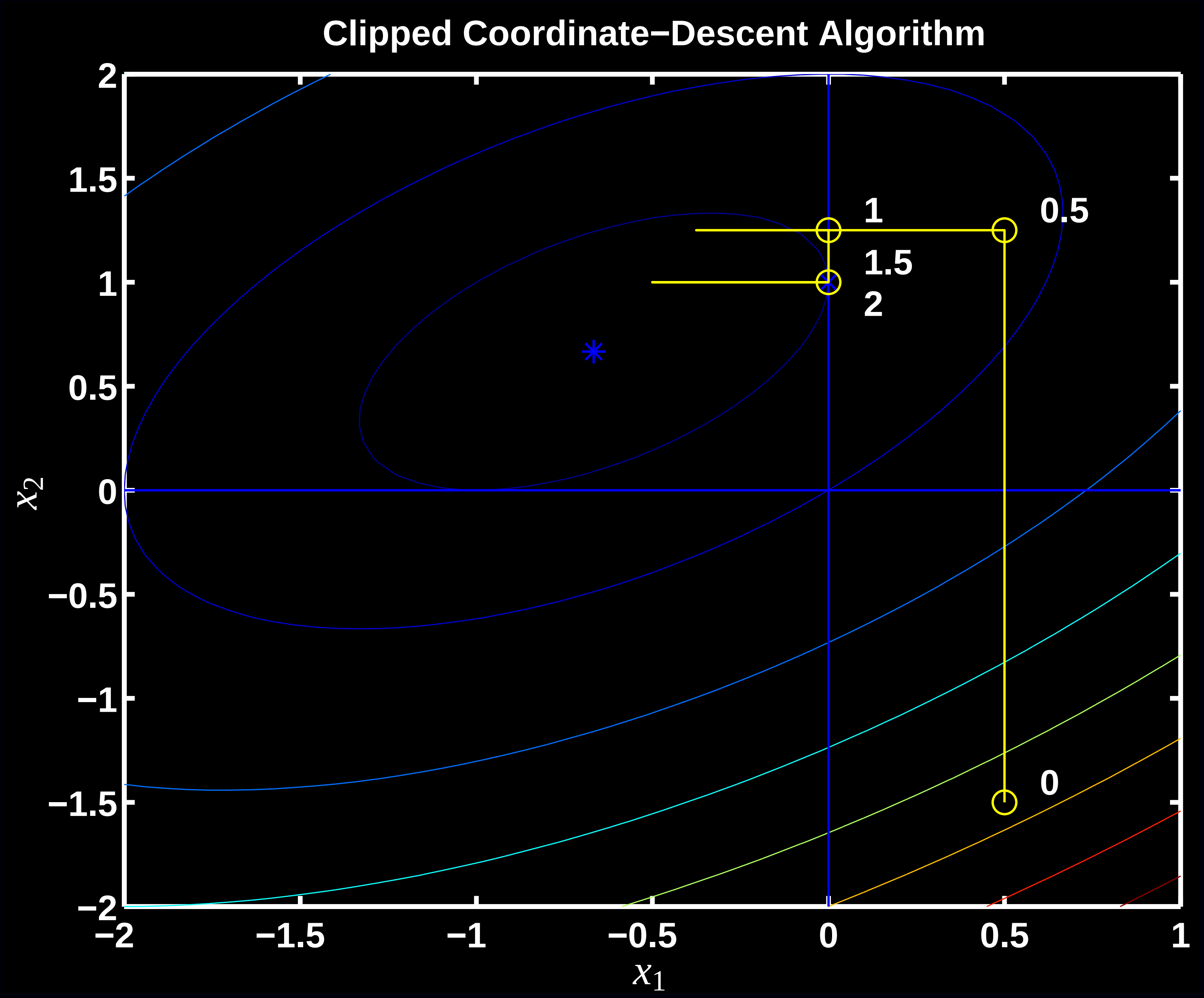
Advantages:

- Intrinsically monotonic
- Fast converging (from good initial image)
- Global convergence
- Nonnegativity constraint trivial

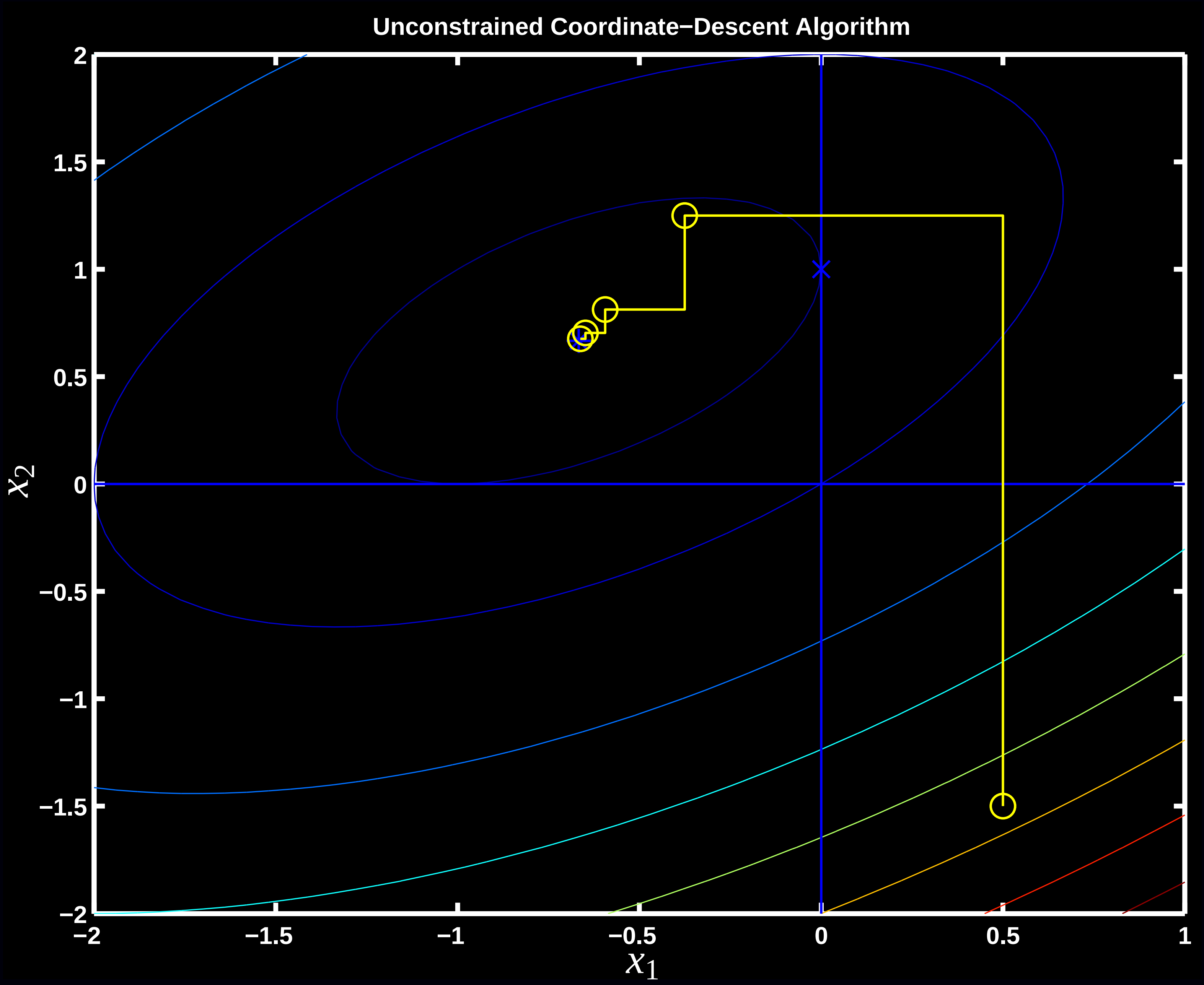
Disadvantages:

- Requires column access of system matrix $\boldsymbol{A}$
- Cannot exploit some “tricks” for $\boldsymbol{A}$, *e.g.*, factorizations
- Expensive “arg min” for nonquadratic problems
- Poorly parallelizable

Constrained Coordinate Descent Illustrated



Coordinate Descent - Unconstrained



Coordinate-Descent Algorithm Summary

Recommended when all of the following apply:

- quadratic or nearly-quadratic convex cost function
- nonnegativity constraint desired
- precomputed and stored system matrix $\boldsymbol{A}$ with column access
- parallelization not needed (standard workstation)

Cautions:

- Good initialization (*e.g.*, properly scaled FBP) essential.
  (Uniform image or zero image cause slow initial convergence.)
- Must be programmed carefully to be efficient.
  (Standard Gauss-Siedel implementation is suboptimal.)
- Updates high-frequencies fastest $\Longrightarrow$ poorly suited to unregularized case

Used daily in UM clinic for 2D SPECT / PWLS / nonuniform attenuation
Under investigation for 3D helical CT reconstruction by Thibault *et al.*

Summary of General-Purpose Algorithms

Gradient-based

- Fully parallelizable
- Inconvenient line-searches for nonquadratic cost functions
- Fast converging in unconstrained case
- Nonnegativity constraint inconvenient

Coordinate-descent

- Very fast converging
- Nonnegativity constraint trivial
- Poorly parallelizable
- Requires precomputed/stored system matrix

CD is well-suited to moderate-sized 2D problem (*e.g.*, 2D PET),
but challenging for large 2D problems (X-ray CT) and fully 3D problems

Neither is ideal.

$\therefore$ need *special-purpose algorithms* for image reconstruction!

Data-Mismatch Functions Revisited

For fast converging, intrinsically monotone algorithms, consider the form of Ψ.

WLS:

$$Ł(\boldsymbol{x}) = \sum_{i=1}^{n_\mathrm{d}} \frac{1}{2} w_i \left(y_i - [\boldsymbol{A}\boldsymbol{x}]_i\right)^2 = \sum_{i=1}^{n_\mathrm{d}} \mathsf{h}_i([\boldsymbol{A}\boldsymbol{x}]_i), \qquad \text{where} \quad \mathsf{h}_i(l) \triangleq \frac{1}{2} w_i \left(y_i - l\right)^2.$$

Emission Poisson (negative) log-likelihood:

$$Ł(\boldsymbol{x}) = \sum_{i=1}^{n_\mathrm{d}} ([\boldsymbol{A}\boldsymbol{x}]_i + r_i) - y_i \log([\boldsymbol{A}\boldsymbol{x}]_i + r_i) = \sum_{i=1}^{n_\mathrm{d}} \mathsf{h}_i([\boldsymbol{A}\boldsymbol{x}]_i)$$

$$\text{where} \quad \mathsf{h}_i(l) \triangleq (l + r_i) - y_i \log(l + r_i)\,.$$

Transmission Poisson log-likelihood:

$$Ł(\boldsymbol{x}) = \sum_{i=1}^{n_\mathrm{d}} \left(b_i\, \mathrm{e}^{-[\boldsymbol{A}\boldsymbol{x}]_i} + r_i\right) - y_i \log\left(b_i\, \mathrm{e}^{-[\boldsymbol{A}\boldsymbol{x}]_i} + r_i\right) = \sum_{i=1}^{n_\mathrm{d}} \mathsf{h}_i([\boldsymbol{A}\boldsymbol{x}]_i)$$

$$\text{where} \quad \mathsf{h}_i(l) \triangleq (b_i e^{-l} + r_i) - y_i \log\left(b_i e^{-l} + r_i\right).$$

MRI, polyenergetic X-ray CT, confocal microscopy, image restoration, ...
All have same *partially separable* form.

General Imaging Cost Function

General form for data-mismatch function:

$$Ł(\boldsymbol{x}) = \sum_{i=1}^{n_\mathrm{d}} \mathsf{h}_i([\boldsymbol{A}\boldsymbol{x}]_i)$$

General form for regularizing penalty function:

$$\mathsf{R}(\boldsymbol{x}) = \sum_k \psi_k([\boldsymbol{C}\boldsymbol{x}]_k)$$

General form for cost function:

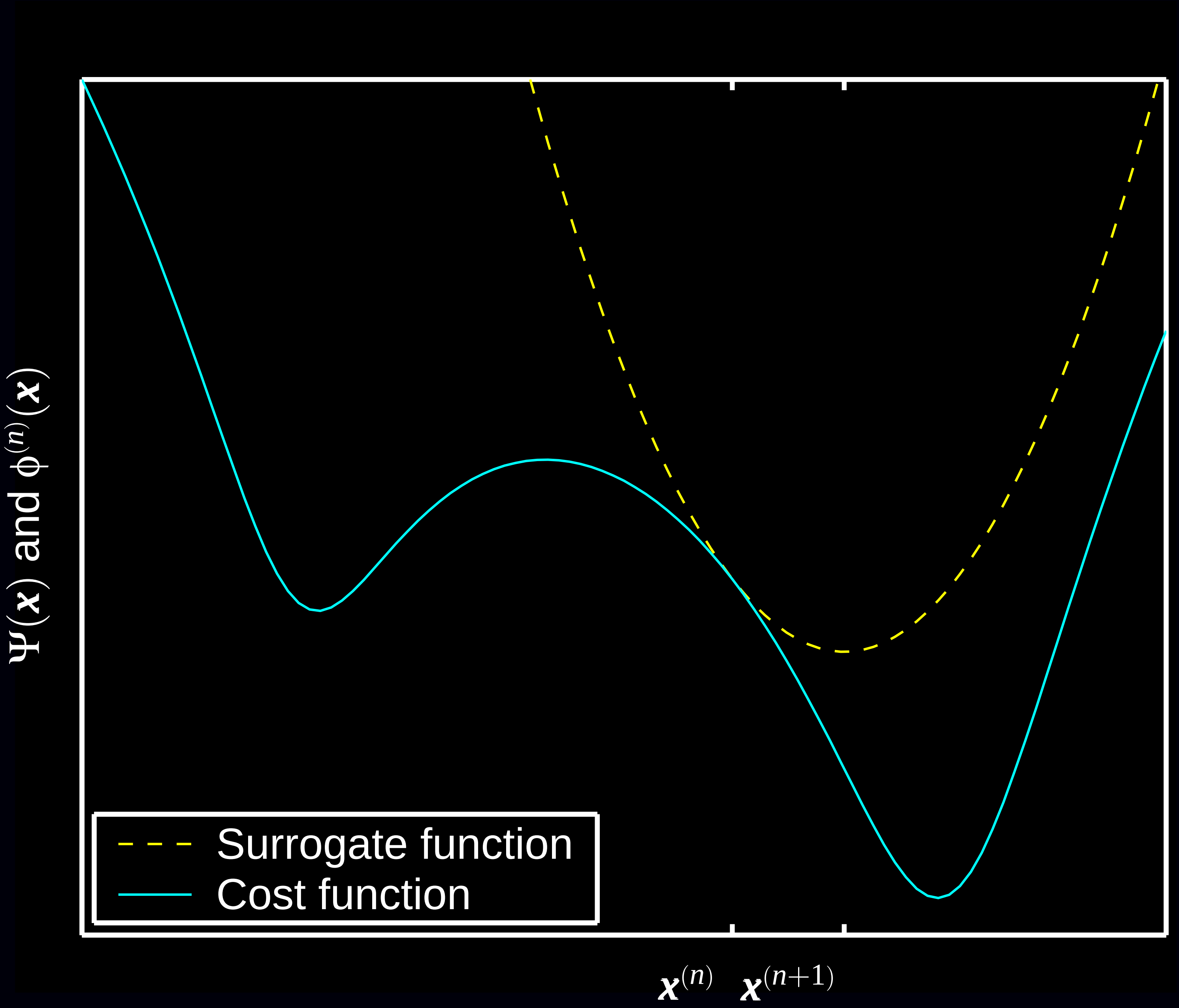
$$\boxed{\Psi(\boldsymbol{x}) = Ł(\boldsymbol{x}) + \beta\,\mathsf{R}(\boldsymbol{x}) = \sum_{i=1}^{n_\mathrm{d}} \mathsf{h}_i([\boldsymbol{A}\boldsymbol{x}]_i) + \beta \sum_k \psi_k([\boldsymbol{C}\boldsymbol{x}]_k)}$$

Properties of Ψ we can exploit:

- summation form (due to independence of measurements)
- convexity of h_i functions (usually)
- summation argument (inner product of $\boldsymbol{x}$ with ith row of $\boldsymbol{A}$)

Most methods that use these properties are forms of *optimization transfer* .

Optimization Transfer Illustrated



Optimization Transfer

General iteration:

$$\boxed{\boldsymbol{x}^{(n+1)} = \arg\min_{\boldsymbol{x} \geq \boldsymbol{0}} \phi\left(\boldsymbol{x}; \boldsymbol{x}^{(n)}\right)}$$

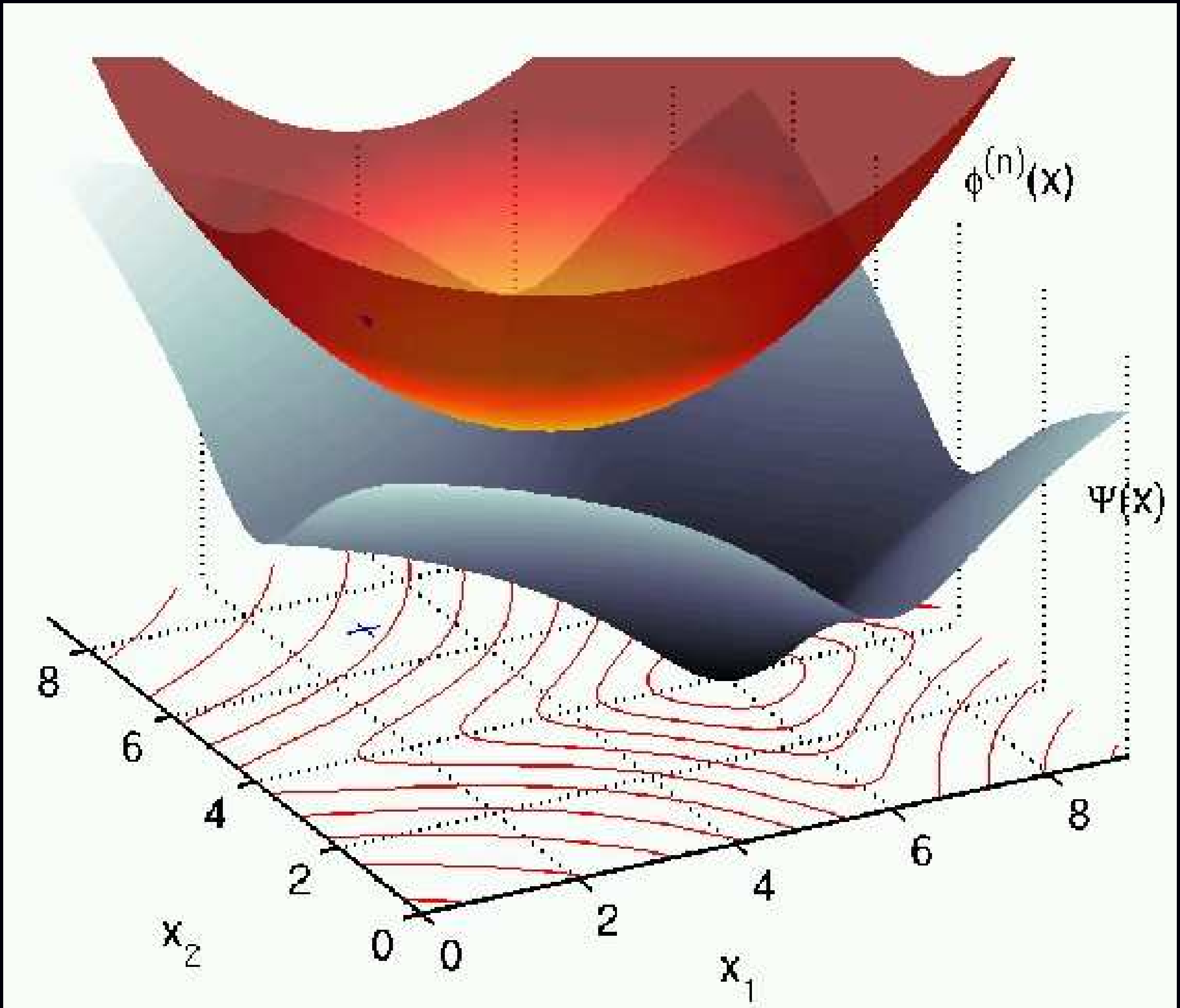
Monotonicity conditions (cost function Ψ decreases provided these hold):

- $\phi(\boldsymbol{x}^{(n)}; \boldsymbol{x}^{(n)}) = \Psi(\boldsymbol{x}^{(n)})$ (matched current value)
- $\nabla_{\boldsymbol{x}} \phi(\boldsymbol{x}; \boldsymbol{x}^{(n)}) \Big|_{\boldsymbol{x}=\boldsymbol{x}^{(n)}} = \nabla \Psi(\boldsymbol{x}) \Big|_{\boldsymbol{x}=\boldsymbol{x}^{(n)}}$ (matched gradient)
- $\phi(\boldsymbol{x}; \boldsymbol{x}^{(n)}) \geq \Psi(\boldsymbol{x}) \quad \forall \boldsymbol{x} \geq \boldsymbol{0}$ (lies above)

These 3 (sufficient) conditions are satisfied by the *Q* function of the EM algorithm (and its relatives like SAGE).

The 3rd condition is *not* satisfied by the Newton-Raphson quadratic approximation, which leads to its nonmonotonicity.

Optimization Transfer in 2d



Optimization Transfer cf EM Algorithm

E-step: choose surrogate function $\phi(\boldsymbol{x};\boldsymbol{x}^{(n)})$

M-step: minimize surrogate function

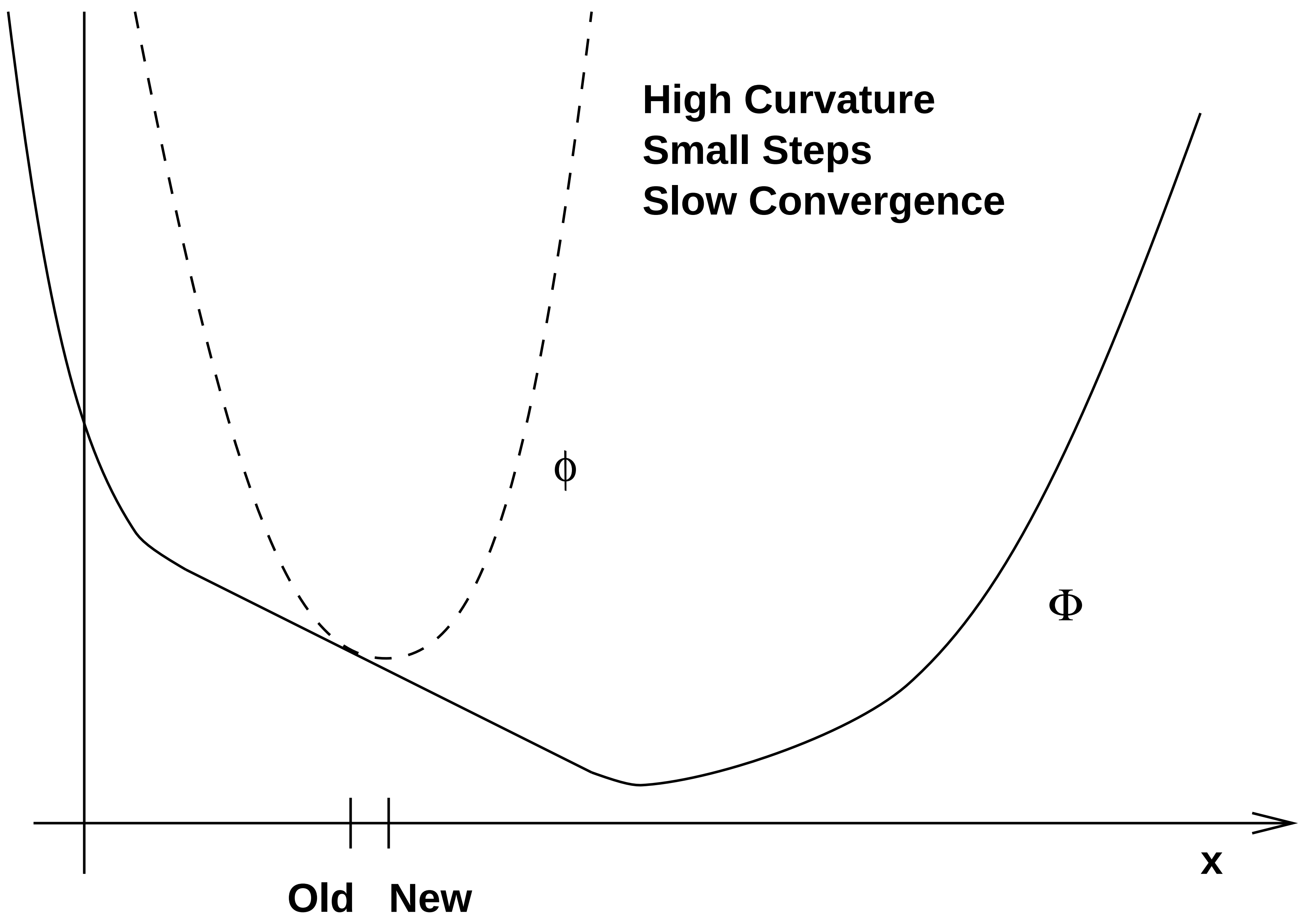
$$\boldsymbol{x}^{(n+1)} = \arg\min_{\boldsymbol{x}\geq\boldsymbol{0}} \phi\left(\boldsymbol{x};\boldsymbol{x}^{(n)}\right)$$

Designing surrogate functions

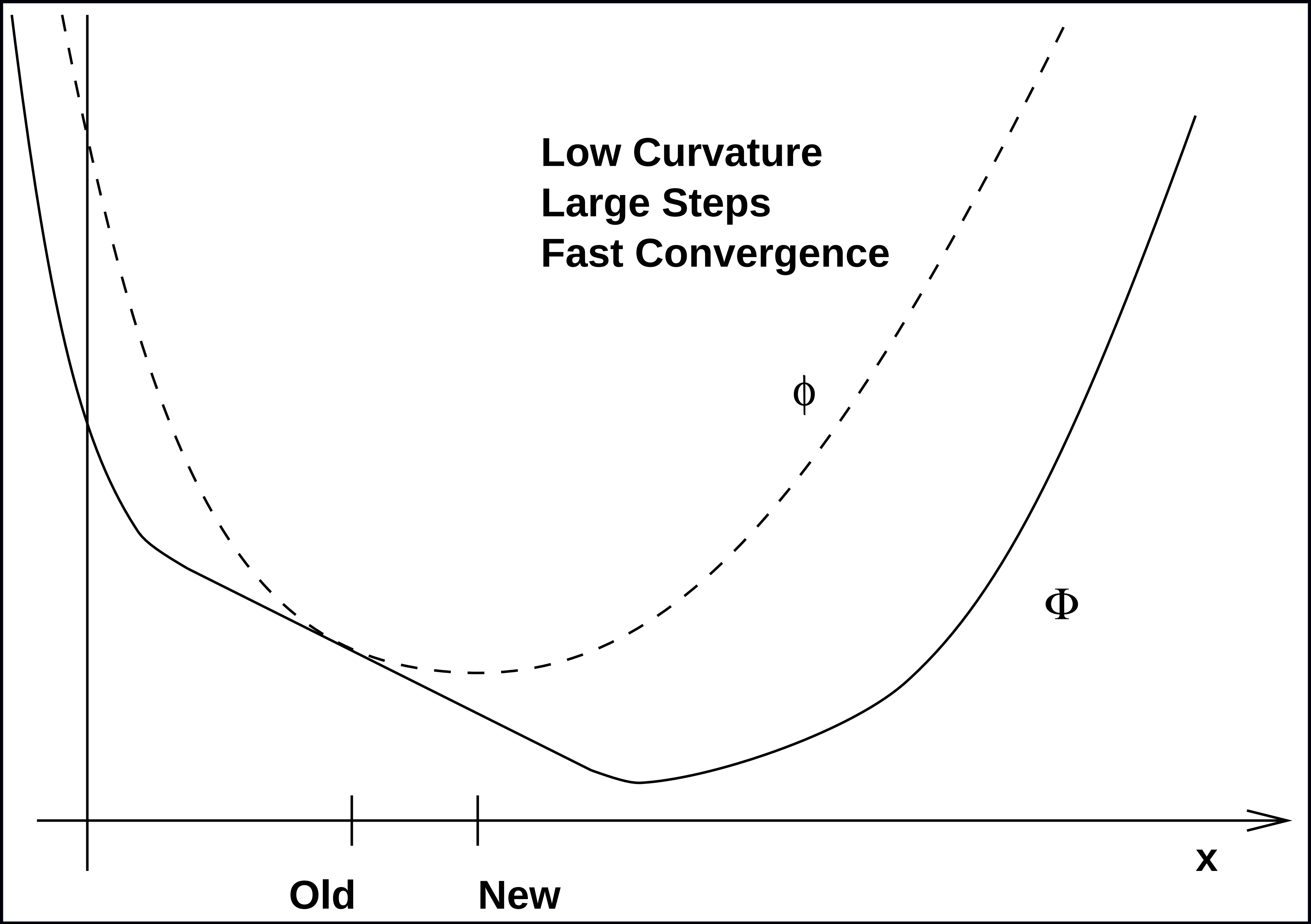
- Easy to “compute”
- Easy to minimize
- Fast convergence rate

Often mutually incompatible goals ∴ compromises

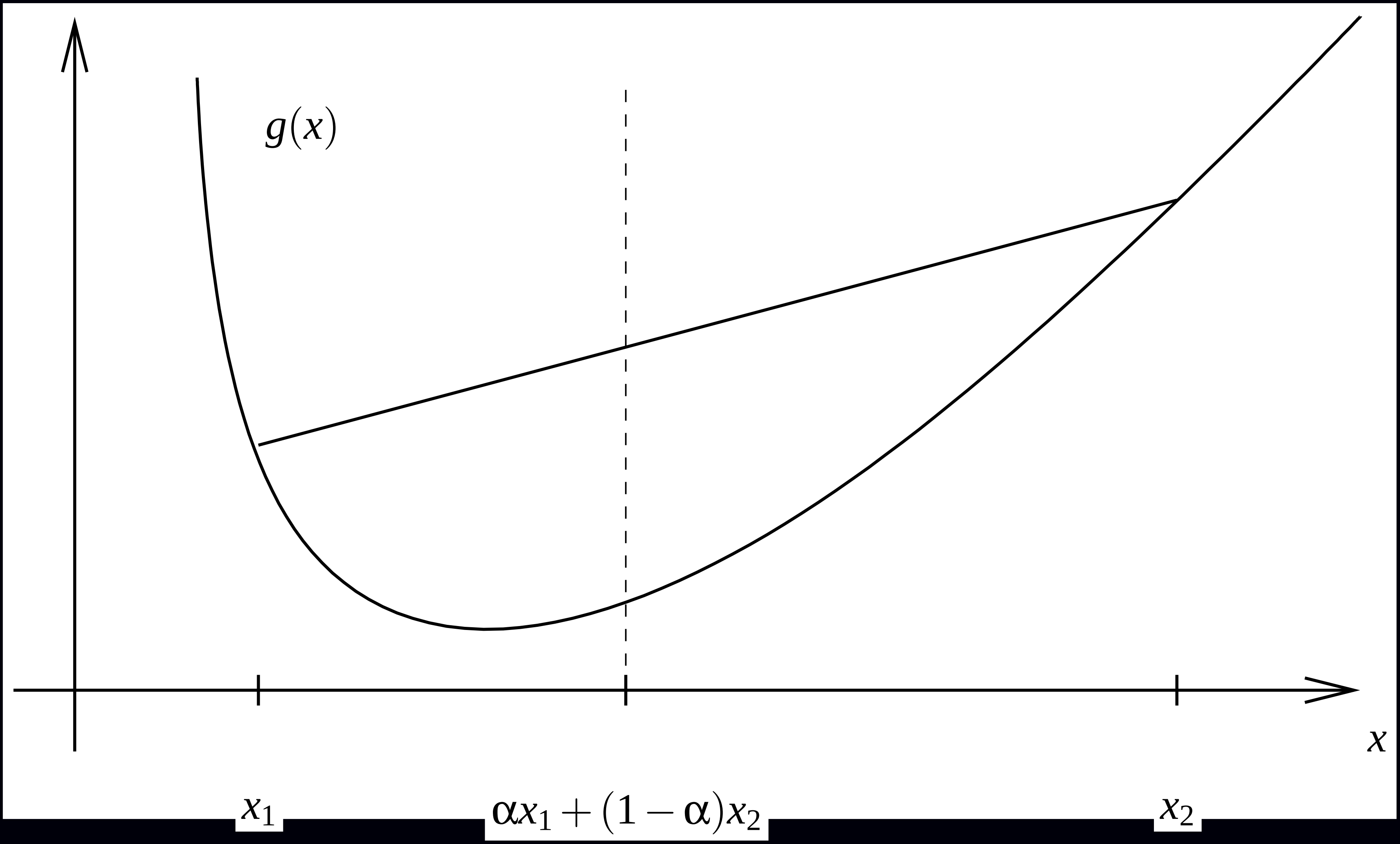
Convergence Rate: Slow



Convergence Rate: Fast



Tool: Convexity Inequality



g convex $\Longrightarrow g(\alpha x_1 + (1-\alpha)x_2) \le \alpha g(x_1) + (1-\alpha)g(x_2)$ for $\alpha \in [0,1]$

More generally: $\alpha_k \ge 0$ and $\sum_k \alpha_k = 1 \Longrightarrow g(\sum_k \alpha_k x_k) \le \sum_k \alpha_k g(x_k)$. Sum outside!

Example 1: Classical ML-EM Algorithm

Negative Poisson log-likelihood cost function (unregularized):

$$\Psi(\boldsymbol{x}) = \sum_{i=1}^{n_\mathrm{d}} \mathsf{h}_i([\boldsymbol{A}\boldsymbol{x}]_i), \qquad \mathsf{h}_i(l) = (l + r_i) - y_i \log(l + r_i)\,.$$

Intractable to minimize directly due to summation within logarithm.

Clever trick due to De Pierro (let $\bar{y}_i^{(n)} = [\boldsymbol{A}\boldsymbol{x}^{(n)}]_i + r_i$):

$$[\boldsymbol{A}\boldsymbol{x}]_i = \sum_{j=1}^{n_\mathrm{p}} a_{ij} x_j = \sum_{j=1}^{n_\mathrm{p}} \left[\frac{a_{ij} x_j^{(n)}}{\bar{y}_i^{(n)}}\right] \left(\frac{x_j}{x_j^{(n)}} \bar{y}_i^{(n)}\right).$$

Since the h_i's are *convex* in Poisson emission model:

$$\mathsf{h}_i([\boldsymbol{A}\boldsymbol{x}]_i) = \mathsf{h}_i\left(\sum_{j=1}^{n_\mathrm{p}} \left[\frac{a_{ij} x_j^{(n)}}{\bar{y}_i^{(n)}}\right] \left(\frac{x_j}{x_j^{(n)}} \bar{y}_i^{(n)}\right)\right) \le \sum_{j=1}^{n_\mathrm{p}} \left[\frac{a_{ij} x_j^{(n)}}{\bar{y}_i^{(n)}}\right] \mathsf{h}_i\left(\frac{x_j}{x_j^{(n)}} \bar{y}_i^{(n)}\right)$$

$$\Psi(\boldsymbol{x}) = \sum_{i=1}^{n_\mathrm{d}} \mathsf{h}_i([\boldsymbol{A}\boldsymbol{x}]_i) \le \phi\left(\boldsymbol{x}; \boldsymbol{x}^{(n)}\right) \triangleq \sum_{i=1}^{n_\mathrm{d}} \sum_{j=1}^{n_\mathrm{p}} \left[\frac{a_{ij} x_j^{(n)}}{\bar{y}_i^{(n)}}\right] \mathsf{h}_i\left(\frac{x_j}{x_j^{(n)}} \bar{y}_i^{(n)}\right)$$

Replace convex cost function $\Psi(\boldsymbol{x})$ with *separable* surrogate function $\phi(\boldsymbol{x}; \boldsymbol{x}^{(n)})$.

“ML-EM Algorithm” M-step

E-step gave separable surrogate function:

$$\phi\left(\boldsymbol{x};\boldsymbol{x}^{(n)}\right) = \sum_{j=1}^{n_{\mathrm{p}}} \phi_j\left(x_j;\boldsymbol{x}^{(n)}\right), \quad \text{where} \quad \phi_j\left(x_j;\boldsymbol{x}^{(n)}\right) \triangleq \sum_{i=1}^{n_{\mathrm{d}}} \left[\frac{a_{ij}x_j^{(n)}}{\bar{y}_i^{(n)}}\right] \mathsf{h}_i\left(\frac{x_j}{x_j^{(n)}}\bar{y}_i^{(n)}\right).$$

M-step separates:

$$\boldsymbol{x}^{(n+1)} = \arg\min_{\boldsymbol{x}\geq\boldsymbol{0}} \phi\left(\boldsymbol{x};\boldsymbol{x}^{(n)}\right) \Longrightarrow x_j^{(n+1)} = \arg\min_{x_j\geq 0} \phi_j\left(x_j;\boldsymbol{x}^{(n)}\right), \qquad j = 1,\ldots,n_{\mathrm{p}}$$

Minimizing:

$$\frac{\partial}{\partial x_j}\phi_j\left(x_j;\boldsymbol{x}^{(n)}\right) = \sum_{i=1}^{n_{\mathrm{d}}} a_{ij}\dot{\mathsf{h}}_i\left(\bar{y}_i^{(n)}x_j/x_j^{(n)}\right) = \sum_{i=1}^{n_{\mathrm{d}}} a_{ij}\left[1 - \frac{y_i}{\bar{y}_i^{(n)}x_j/x_j^{(n)}}\right]\Bigg|_{x_j=x_j^{(n+1)}} = 0.$$

Solving (in case $r_i = 0$):

$$\boxed{x_j^{(n+1)} = x_j^{(n)}\left[\sum_{i=1}^{n_{\mathrm{d}}} a_{ij}\frac{y_i}{[\boldsymbol{A}\boldsymbol{x}^{(n)}]_i}\right] \Big/ \left(\sum_{i=1}^{n_{\mathrm{d}}} a_{ij}\right), \qquad j = 1,\ldots,n_{\mathrm{p}}}$$

- Derived without any statistical considerations, unlike classical EM formulation.
- Uses only convexity and algebra.
- Guaranteed monotonic: surrogate function ϕ satisfies the 3 required properties.
- M-step trivial due to *separable surrogate*.

ML-EM is Scaled Gradient Descent

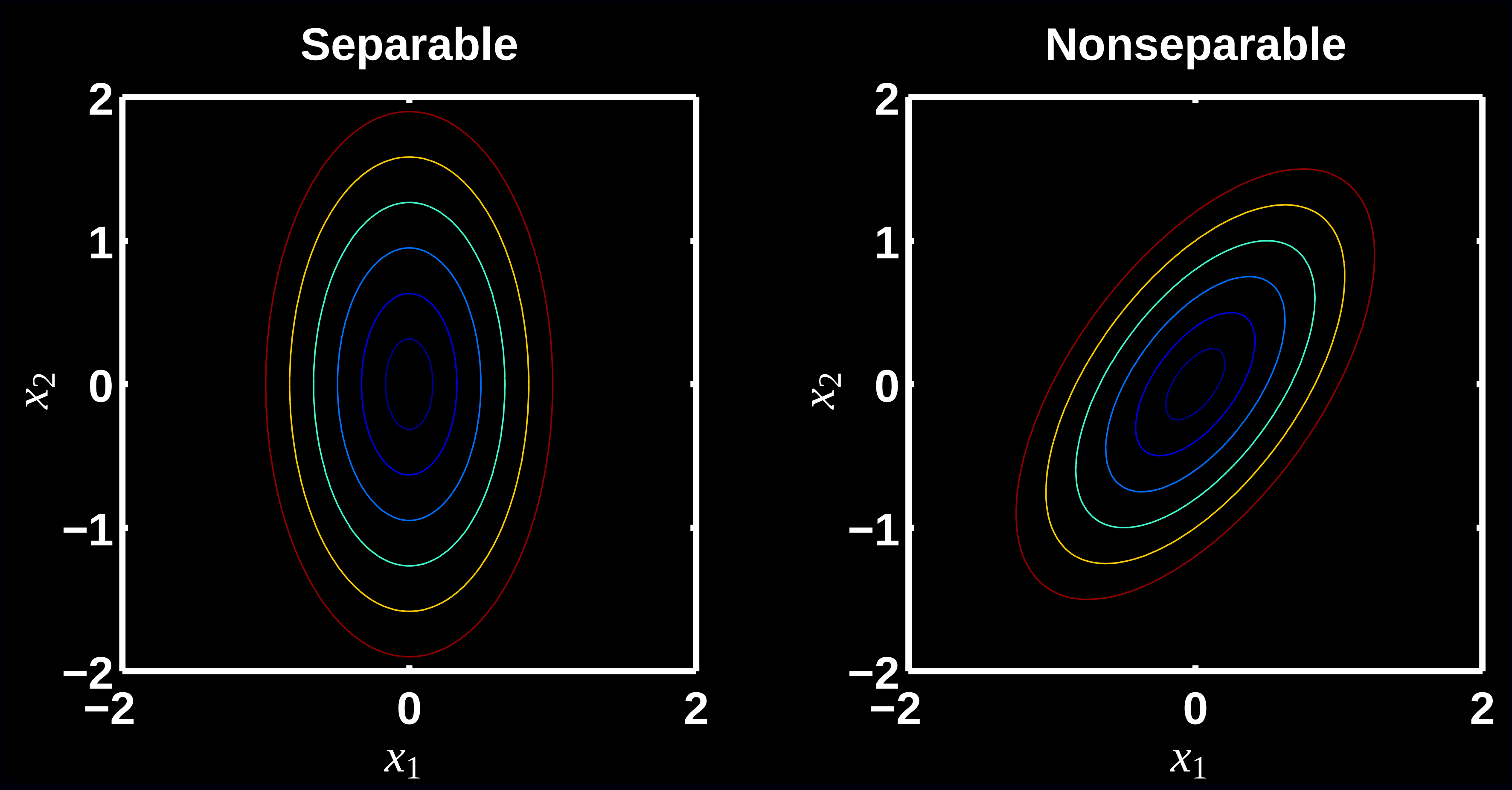
$$\begin{aligned}
x_j^{(n+1)} &= x_j^{(n)} \left[\sum_{i=1}^{n_\mathrm{d}} a_{ij} \frac{y_i}{\bar{y}_i^{(n)}} \right] \Big/ \left(\sum_{i=1}^{n_\mathrm{d}} a_{ij} \right) \\
&= x_j^{(n)} + x_j^{(n)} \left[\sum_{i=1}^{n_\mathrm{d}} a_{ij} \left(\frac{y_i}{\bar{y}_i^{(n)}} - 1 \right) \right] \Big/ \left(\sum_{i=1}^{n_\mathrm{d}} a_{ij} \right) \\
&= \boxed{x_j^{(n)} - \left(\frac{x_j^{(n)}}{\sum_{i=1}^{n_\mathrm{d}} a_{ij}} \right) \frac{\partial}{\partial x_j} \Psi\left(\boldsymbol{x}^{(n)}\right),} \qquad j = 1, \ldots, n_\mathrm{p}
\end{aligned}$$

$$\boldsymbol{x}^{(n+1)} = \boldsymbol{x}^{(n)} + \boldsymbol{D}(\boldsymbol{x}^{(n)}) \nabla \Psi\left(\boldsymbol{x}^{(n)}\right)$$

This particular diagonal scaling matrix remarkably

- ensures monotonicity,
- ensures nonnegativity.

Consideration: Separable vs Nonseparable



Contour plots: loci of equal function values.

Uncoupled vs coupled minimization.

Separable Surrogate Functions (Easy M-step)

The preceding EM derivation structure applies to *any* cost function of the form

$$\Psi(\boldsymbol{x}) = \sum_{i=1}^{n_\mathrm{d}} \mathsf{h}_i([\boldsymbol{A}\boldsymbol{x}]_i)\,.$$

cf ISRA (for nonnegative LS), “convex algorithm” for transmission reconstruction

Derivation yields a separable surrogate function

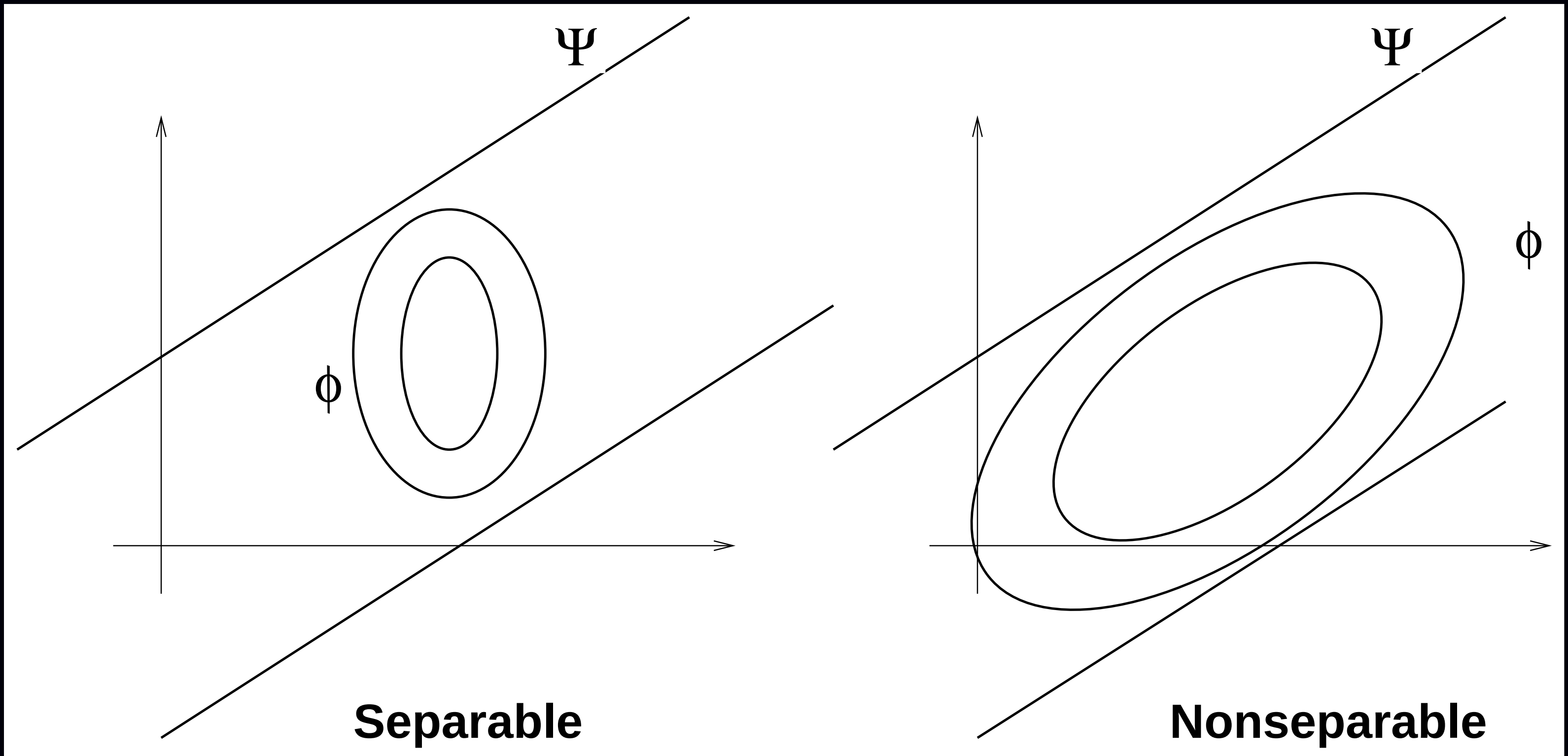
$$\Psi(\boldsymbol{x}) \leq \phi\left(\boldsymbol{x};\boldsymbol{x}^{(n)}\right), \quad \text{where} \quad \phi\left(\boldsymbol{x};\boldsymbol{x}^{(n)}\right) = \sum_{j=1}^{n_\mathrm{p}} \phi_j\left(x_j;\boldsymbol{x}^{(n)}\right)$$

M-step separates into 1D minimization problems (fully parallelizable):

$$\boldsymbol{x}^{(n+1)} = \arg\min_{\boldsymbol{x}\geq\boldsymbol{0}} \phi\left(\boldsymbol{x};\boldsymbol{x}^{(n)}\right) \Longrightarrow x_j^{(n+1)} = \arg\min_{x_j\geq 0} \phi_j\left(x_j;\boldsymbol{x}^{(n)}\right), \qquad j = 1,\ldots,n_\mathrm{p}$$

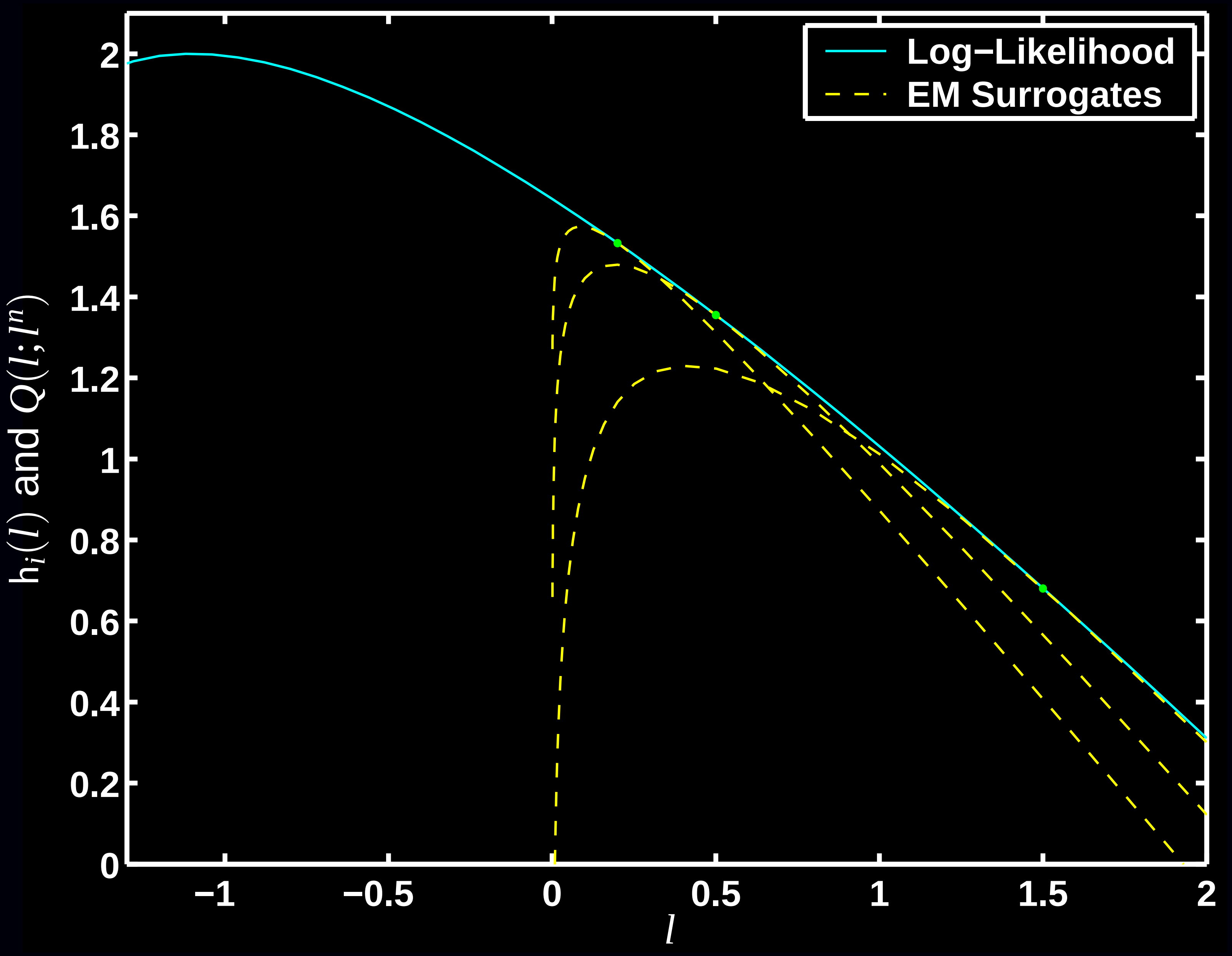
Why do EM / ISRA / convex-algorithm / etc. converge so slowly?

Separable vs Nonseparable



Separable surrogates (*e.g.*, EM) have high curvature ∴ slow convergence.
Nonseparable surrogates can have lower curvature ∴ faster convergence.
Harder to minimize? Use paraboloids (quadratic surrogates).

High Curvature of EM Surrogate



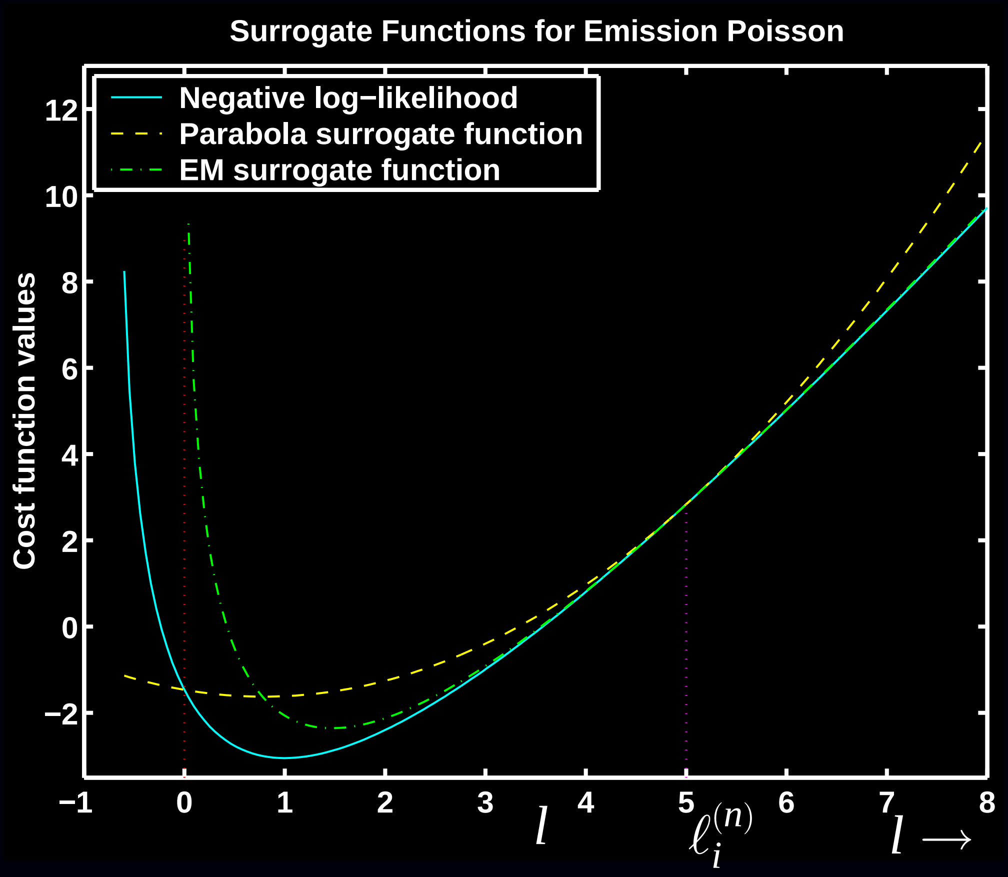
1D Parabola Surrogate Function

Find parabola $q_i^{(n)}(l)$ of the form:

$$q_i^{(n)}(l) = \mathsf{h}_i\left(\ell_i^{(n)}\right) + \dot{\mathsf{h}}_i\left(\ell_i^{(n)}\right)(l - \ell_i^{(n)}) + c_i^{(n)} \frac{1}{2}(l - \ell_i^{(n)})^2, \quad \text{where } \ell_i^{(n)} \triangleq [\boldsymbol{A}\boldsymbol{x}^{(n)}]_i$$

Satisfies tangent condition. Choose curvature to ensure “lies above” condition:

$$c_i^{(n)} \triangleq \min\left\{c \geq 0 : q_i^{(n)}(l) \geq \mathsf{h}_i(l), \quad \forall l \geq 0\right\}.$$



Lower curvature!

Paraboloidal Surrogate

Combining 1D parabola surrogates yields *paraboloidal surrogate*:

$$\Psi(\boldsymbol{x}) = \sum_{i=1}^{n_{\mathrm{d}}} \mathsf{h}_i([\boldsymbol{A}\boldsymbol{x}]_i) \leq \phi\left(\boldsymbol{x};\boldsymbol{x}^{(n)}\right) = \sum_{i=1}^{n_{\mathrm{d}}} q_i^{(n)}([\boldsymbol{A}\boldsymbol{x}]_i)$$

Rewriting: $\phi\left(\boldsymbol{\delta}+\boldsymbol{x}^{(n)};\boldsymbol{x}^{(n)}\right) = \Psi\left(\boldsymbol{x}^{(n)}\right) + \nabla\Psi\left(\boldsymbol{x}^{(n)}\right)\boldsymbol{\delta} + \frac{1}{2}\boldsymbol{\delta}'\boldsymbol{A}'\,\mathrm{diag}\left\{c_i^{(n)}\right\}\boldsymbol{A}\boldsymbol{\delta}$

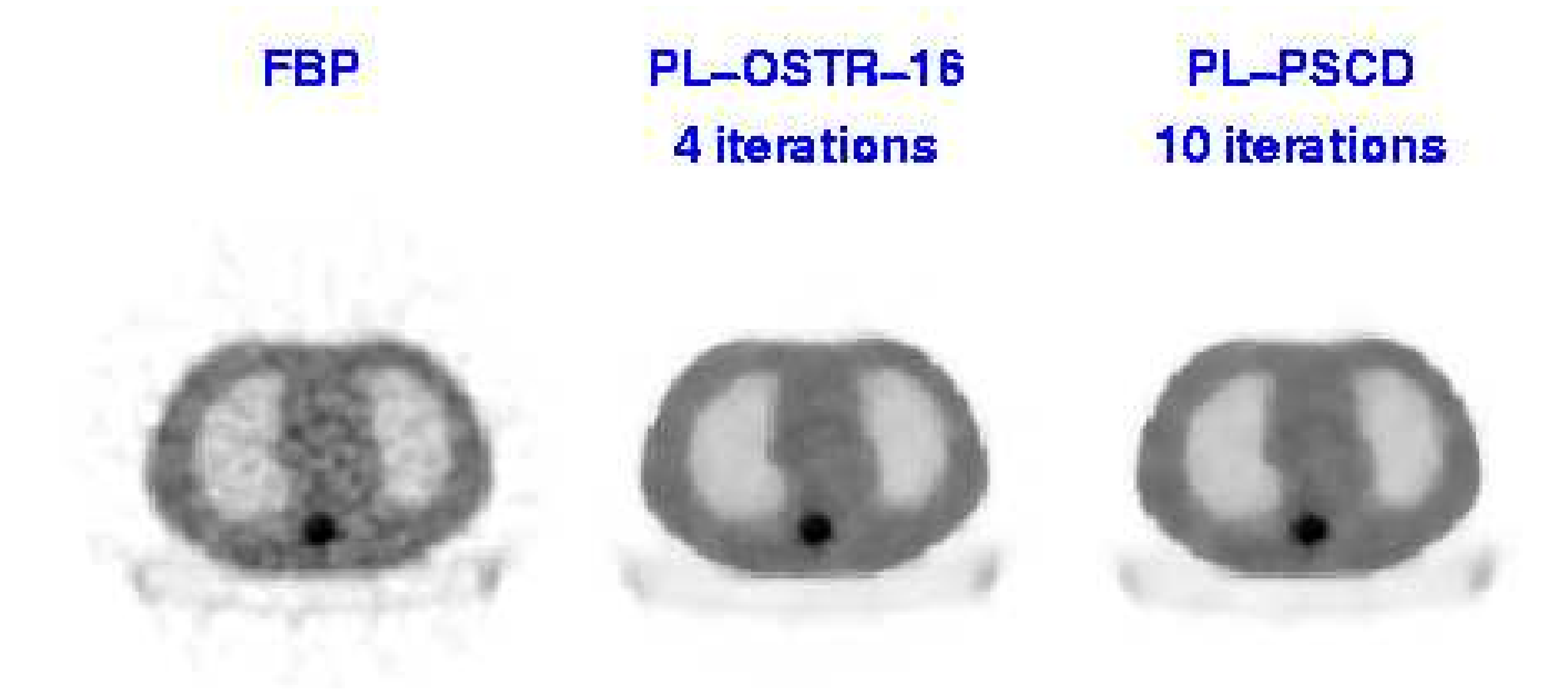
Advantages

- Surrogate $\phi(\boldsymbol{x};\boldsymbol{x}^{(n)})$ is *quadratic*, unlike Poisson log-likelihood
  $\Longrightarrow$ easier to minimize
- Not separable (unlike EM surrogate)
- Not self-similar (unlike EM surrogate)
- Small curvatures $\Longrightarrow$ fast convergence
- Intrinsically monotone global convergence
- Fairly simple to derive / implement

Quadratic minimization

- Coordinate descent
  - + fast converging
  - + Nonnegativity easy
  - - precomputed column-stored system matrix
- Gradient-based quadratic minimization methods
  - - Nonnegativity inconvenient

Example: PSCD for PET Transmission Scans



- square-pixel basis
- strip-integral system model
- shifted-Poisson statistical model
- edge-preserving convex regularization (Huber)
- nonnegativity constraint
- inscribed circle support constraint
- paraboloidal surrogate coordinate descent (PSCD) algorithm

Separable Paraboloidal Surrogate

To derive a parallelizable algorithm apply another De Pierro trick:

$$[\boldsymbol{A}\boldsymbol{x}]_i = \sum_{j=1}^{n_\mathrm{p}} \pi_{ij} \left[\frac{a_{ij}}{\pi_{ij}} (x_j - x_j^{(n)}) + \ell_i^{(n)} \right], \qquad \ell_i^{(n)} = [\boldsymbol{A}\boldsymbol{x}^{(n)}]_i.$$

Provided $\pi_{ij} \geq 0$ and $\sum_{j=1}^{n_\mathrm{p}} \pi_{ij} = 1$, since parabola q_i is convex:

$$q_i^{(n)}([\boldsymbol{A}\boldsymbol{x}]_i) = q_i^{(n)} \left(\sum_{j=1}^{n_\mathrm{p}} \pi_{ij} \left[\frac{a_{ij}}{\pi_{ij}} (x_j - x_j^{(n)}) + \ell_i^{(n)} \right] \right) \leq \sum_{j=1}^{n_\mathrm{p}} \pi_{ij}\, q_i^{(n)} \left(\frac{a_{ij}}{\pi_{ij}} (x_j - x_j^{(n)}) + \ell_i^{(n)} \right)$$

$$\therefore \phi\left(\boldsymbol{x}; \boldsymbol{x}^{(n)}\right) = \sum_{i=1}^{n_\mathrm{d}} q_i^{(n)}([\boldsymbol{A}\boldsymbol{x}]_i) \leq \tilde{\phi}\left(\boldsymbol{x}; \boldsymbol{x}^{(n)}\right) \triangleq \sum_{i=1}^{n_\mathrm{d}} \sum_{j=1}^{n_\mathrm{p}} \pi_{ij}\, q_i^{(n)} \left(\frac{a_{ij}}{\pi_{ij}} (x_j - x_j^{(n)}) + \ell_i^{(n)} \right)$$

Separable Paraboloidal Surrogate:

$$\tilde{\phi}\left(\boldsymbol{x}; \boldsymbol{x}^{(n)}\right) = \sum_{j=1}^{n_\mathrm{p}} \phi_j\left(x_j; \boldsymbol{x}^{(n)}\right), \qquad \phi_j\left(x_j; \boldsymbol{x}^{(n)}\right) \triangleq \sum_{i=1}^{n_\mathrm{d}} \pi_{ij}\, q_i^{(n)} \left(\frac{a_{ij}}{\pi_{ij}} (x_j - x_j^{(n)}) + \ell_i^{(n)} \right)$$

Parallelizable M-step (cf gradient descent!):

$$x_j^{(n+1)} = \operatorname*{arg\,min}_{x_j \geq 0} \phi_j\left(x_j; \boldsymbol{x}^{(n)}\right) = \left[x_j^{(n)} - \frac{1}{d_j^{(n)}} \frac{\partial}{\partial x_j} \Psi\left(\boldsymbol{x}^{(n)}\right) \right]_+, \qquad d_j^{(n)} = \sum_{i=1}^{n_\mathrm{d}} \frac{a_{ij}^2}{\pi_{ij}} c_i^{(n)}$$

Natural choice is $\pi_{ij} = |a_{ij}| / |a|_i$, $|a|_i = \sum_{j=1}^{n_\mathrm{p}} |a_{ij}|$

Example: Poisson ML Transmission Problem

Transmission negative log-likelihood (for ith ray):

$$\mathsf{h}_i(l) = (b_i e^{-l} + r_i) - y_i \log\left(b_i e^{-l} + r_i\right).$$

Optimal (smallest) parabola surrogate curvature (Erdoğan, T-MI, Sep. 1999):

$$c_i^{(n)} = c(\ell_i^{(n)}, \mathsf{h}_i), \qquad c(l,h) = \begin{cases} \left[2\dfrac{h(0) - h(l) + \dot{h}(l)l}{l^2}\right]_+, & l > 0 \\ \left[\ddot{h}(l)\right]_+, & l = 0. \end{cases}$$

Separable Paraboloidal Surrogate (SPS) Algorithm:

Precompute $|a|_i = \sum_{j=1}^{n_\mathrm{p}} a_{ij}, \qquad i = 1, \ldots, n_\mathrm{d}$

$$\begin{aligned} \ell_i^{(n)} &= [\boldsymbol{A}\boldsymbol{x}^{(n)}]_i, && \text{(forward projection)} \\ \bar{y}_i^{(n)} &= b_i \mathrm{e}^{-\ell_i^{(n)}} + r_i && \text{(predicted means)} \\ \dot{\mathsf{h}}_i^{(n)} &= 1 - y_i / \bar{y}_i^{(n)} && \text{(slopes)} \\ c_i^{(n)} &= c(\ell_i^{(n)}, \mathsf{h}_i) && \text{(curvatures)} \end{aligned}$$

$$x_j^{(n+1)} = \left[x_j^{(n)} - \frac{1}{d_j^{(n)}} \frac{\partial}{\partial x_j} \Psi\left(\boldsymbol{x}^{(n)}\right)\right]_+ = \left[x_j^{(n)} - \frac{\sum_{i=1}^{n_\mathrm{d}} a_{ij} \dot{\mathsf{h}}_i^{(n)}}{\sum_{i=1}^{n_\mathrm{d}} a_{ij} |a|_i c_i^{(n)}}\right]_+, \qquad j = 1, \ldots, n_\mathrm{p}$$

Monotonically decreases cost function each iteration. No logarithm!

The MAP-EM M-step “Problem”

Add a penalty function to our surrogate for the negative log-likelihood:

$$\begin{aligned} \Psi(\boldsymbol{x}) &= Ł(\boldsymbol{x}) + \beta \mathsf{R}(\boldsymbol{x}) \\ \phi\left(\boldsymbol{x}; \boldsymbol{x}^{(n)}\right) &= \sum_{j=1}^{n_\mathrm{p}} \phi_j\left(x_j; \boldsymbol{x}^{(n)}\right) + \beta \mathsf{R}(\boldsymbol{x}) \end{aligned}$$

M-step: $\boldsymbol{x}^{(n+1)} = \arg\min_{\boldsymbol{x} \geq \boldsymbol{0}} \phi\left(\boldsymbol{x}; \boldsymbol{x}^{(n)}\right) = \arg\min_{\boldsymbol{x} \geq \boldsymbol{0}} \sum_{j=1}^{n_\mathrm{p}} \phi_j\left(x_j; \boldsymbol{x}^{(n)}\right) + \beta \mathsf{R}(\boldsymbol{x}) = ?$

For nonseparable penalty functions, the M-step is coupled ∴ difficult.

Suboptimal solutions

- Generalized EM (GEM) algorithm (coordinate descent on ϕ)
  Monotonic, but inherits slow convergence of EM.
- One-step late (OSL) algorithm (use outdated gradients) (Green, T-MI, 1990)

$$\frac{\partial}{\partial x_j} \phi(\boldsymbol{x}; \boldsymbol{x}^{(n)}) = \frac{\partial}{\partial x_j} \phi_j(x_j; \boldsymbol{x}^{(n)}) + \beta \frac{\partial}{\partial x_j} R(\boldsymbol{x}) \overset{?}{\approx} \frac{\partial}{\partial x_j} \phi_j(x_j; \boldsymbol{x}^{(n)}) + \beta \frac{\partial}{\partial x_j} R(\boldsymbol{x}^{(n)})$$

  Nonmonotonic. Known to diverge, depending on β.
  Temptingly simple, but *avoid!*

Contemporary solution

- Use separable surrogate for penalty function too (De Pierro, T-MI, Dec. 1995)
  Ensures monotonicity. Obviates all reasons for using OSL!

De Pierro's MAP-EM Algorithm

Apply separable paraboloidal surrogates to penalty function:

$$R(\boldsymbol{x}) \leq R_{\mathrm{SPS}}(\boldsymbol{x};\boldsymbol{x}^{(n)}) = \sum_{j=1}^{n_{\mathrm{p}}} R_j(x_j;\boldsymbol{x}^{(n)})$$

Overall separable surrogate: $\phi\left(\boldsymbol{x};\boldsymbol{x}^{(n)}\right) = \sum_{j=1}^{n_{\mathrm{p}}} \phi_j\left(x_j;\boldsymbol{x}^{(n)}\right) + \beta \sum_{j=1}^{n_{\mathrm{p}}} R_j(x_j;\boldsymbol{x}^{(n)})$

The M-step becomes fully parallelizable:

$$x_j^{(n+1)} = \operatorname*{arg\,min}_{x_j \geq 0} \phi_j\left(x_j;\boldsymbol{x}^{(n)}\right) - \beta R_j(x_j;\boldsymbol{x}^{(n)}), \qquad j = 1,\ldots,n_{\mathrm{p}}.$$

Consider quadratic penalty $\mathrm{R}(\boldsymbol{x}) = \sum_k \psi([\boldsymbol{C}\boldsymbol{x}]_k)$, where $\psi(t) = t^2/2$.
If $\gamma_{kj} \geq 0$ and $\sum_{j=1}^{n_{\mathrm{p}}} \gamma_{kj} = 1$ then

$$[\boldsymbol{C}\boldsymbol{x}]_k = \sum_{j=1}^{n_{\mathrm{p}}} \gamma_{kj} \left[\frac{c_{kj}}{\gamma_{kj}}(x_j - x_j^{(n)}) + [\boldsymbol{C}\boldsymbol{x}^{(n)}]_k\right].$$

Since ψ is convex:

$$\begin{aligned} \psi([\boldsymbol{C}\boldsymbol{x}]_k) &= \psi\left(\sum_{j=1}^{n_{\mathrm{p}}} \gamma_{kj} \left[\frac{c_{kj}}{\gamma_{kj}}(x_j - x_j^{(n)}) + [\boldsymbol{C}\boldsymbol{x}^{(n)}]_k\right]\right) \\ &\leq \sum_{j=1}^{n_{\mathrm{p}}} \gamma_{kj} \psi\left(\frac{c_{kj}}{\gamma_{kj}}(x_j - x_j^{(n)}) + [\boldsymbol{C}\boldsymbol{x}^{(n)}]_k\right) \end{aligned}$$

De Pierro’s Algorithm Continued

So $\mathrm{R}(\boldsymbol{x}) \leq R(\boldsymbol{x};\boldsymbol{x}^{(n)}) \triangleq \sum_{j=1}^{n_\mathrm{p}} R_j(x_j;\boldsymbol{x}^{(n)})$ where

$$R_j(x_j;\boldsymbol{x}^{(n)}) \triangleq \sum_k \gamma_{kj}\, \psi\left(\frac{c_{kj}}{\gamma_{kj}} (x_j - x_j^{(n)}) + [\boldsymbol{C}\boldsymbol{x}^{(n)}]_k \right)$$

M-step: Minimizing $\phi_j(x_j;\boldsymbol{x}^{(n)}) + \beta R_j(x_j;\boldsymbol{x}^{(n)})$ yields the iteration:

$$x_j^{(n+1)} = \frac{x_j^{(n)} \sum_{i=1}^{n_\mathrm{d}} a_{ij} y_i / \bar{y}_i^{(n)}}{B_j + \sqrt{B_j^2 + \left(x_j^{(n)} \sum_{i=1}^{n_\mathrm{d}} a_{ij} y_i / \bar{y}_i^{(n)} \right) \left(\beta \sum_k c_{kj}^2 / \gamma_{kj} \right)}}$$

where $B_j \triangleq \dfrac{1}{2} \left[\sum_{i=1}^{n_\mathrm{d}} a_{ij} + \beta \sum_k \left(c_{kj} [\boldsymbol{C}\boldsymbol{x}^{(n)}]_k - \frac{c_{kj}^2}{\gamma_{kj}} x_j^{(n)} \right) \right], \qquad j = 1, \ldots, n_\mathrm{p}$

and $\bar{y}_i^{(n)} = [\boldsymbol{A}\boldsymbol{x}^{(n)}]_i + r_i$.

Advantages: Intrinsically monotone, nonnegativity, fully parallelizable.
Requires only a couple % more computation per iteration than ML-EM

Disadvantages: Slow convergence (like EM) due to separable surrogate

Ordered Subsets Algorithms

aka *block iterative* or *incremental gradient* algorithms

The gradient appears in essentially every algorithm:

$$\text{Ł}(\boldsymbol{x}) = \sum_{i=1}^{n_{\mathrm{d}}} \mathsf{h}_i([\boldsymbol{A}\boldsymbol{x}]_i) \Longrightarrow \frac{\partial}{\partial x_j} \text{Ł}(\boldsymbol{x}) = \sum_{i=1}^{n_{\mathrm{d}}} a_{ij}\, \dot{\mathsf{h}}_i([\boldsymbol{A}\boldsymbol{x}]_i)\,.$$

This is a *backprojection* of a sinogram of the derivatives $\left\{\dot{\mathsf{h}}_i([\boldsymbol{A}\boldsymbol{x}]_i)\right\}$.

Intuition: with half the angular sampling, this backprojection would be fairly similar

$$\frac{1}{n_{\mathrm{d}}} \sum_{i=1}^{n_{\mathrm{d}}} a_{ij}\, \dot{\mathsf{h}}_i(\cdot) \approx \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} a_{ij}\, \dot{\mathsf{h}}_i(\cdot),$$

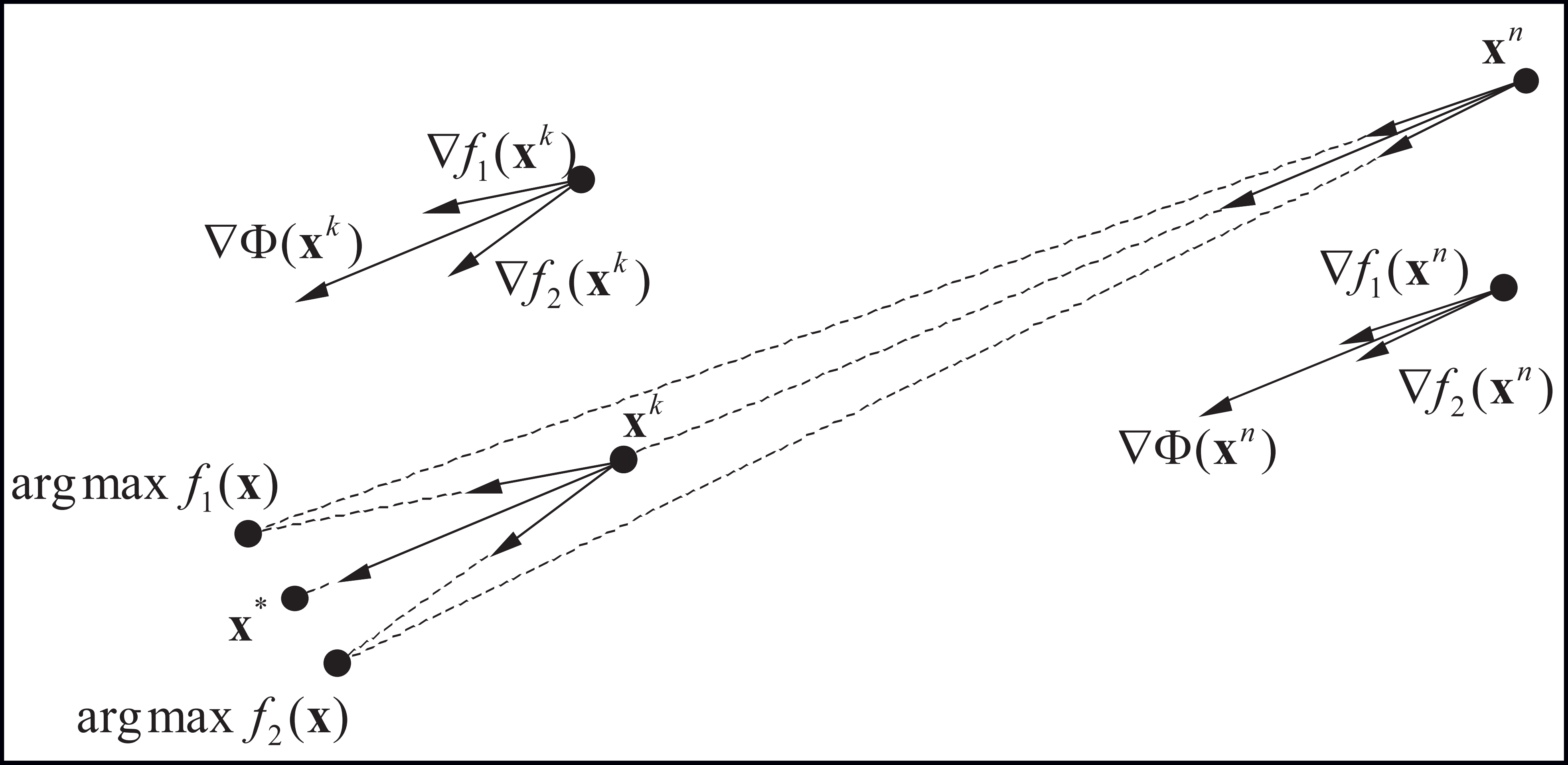
where $\mathcal{S}$ is a subset of the rays.

To “OS-ize” an algorithm, replace all backprojections with partial sums.

Recall typical iteration:

$$\boldsymbol{x}^{(n+1)} = \boldsymbol{x}^{(n)} - \boldsymbol{D}(\boldsymbol{x}^{(n)})\, \nabla\Psi\left(\boldsymbol{x}^{(n)}\right).$$

Geometric View of Ordered Subsets

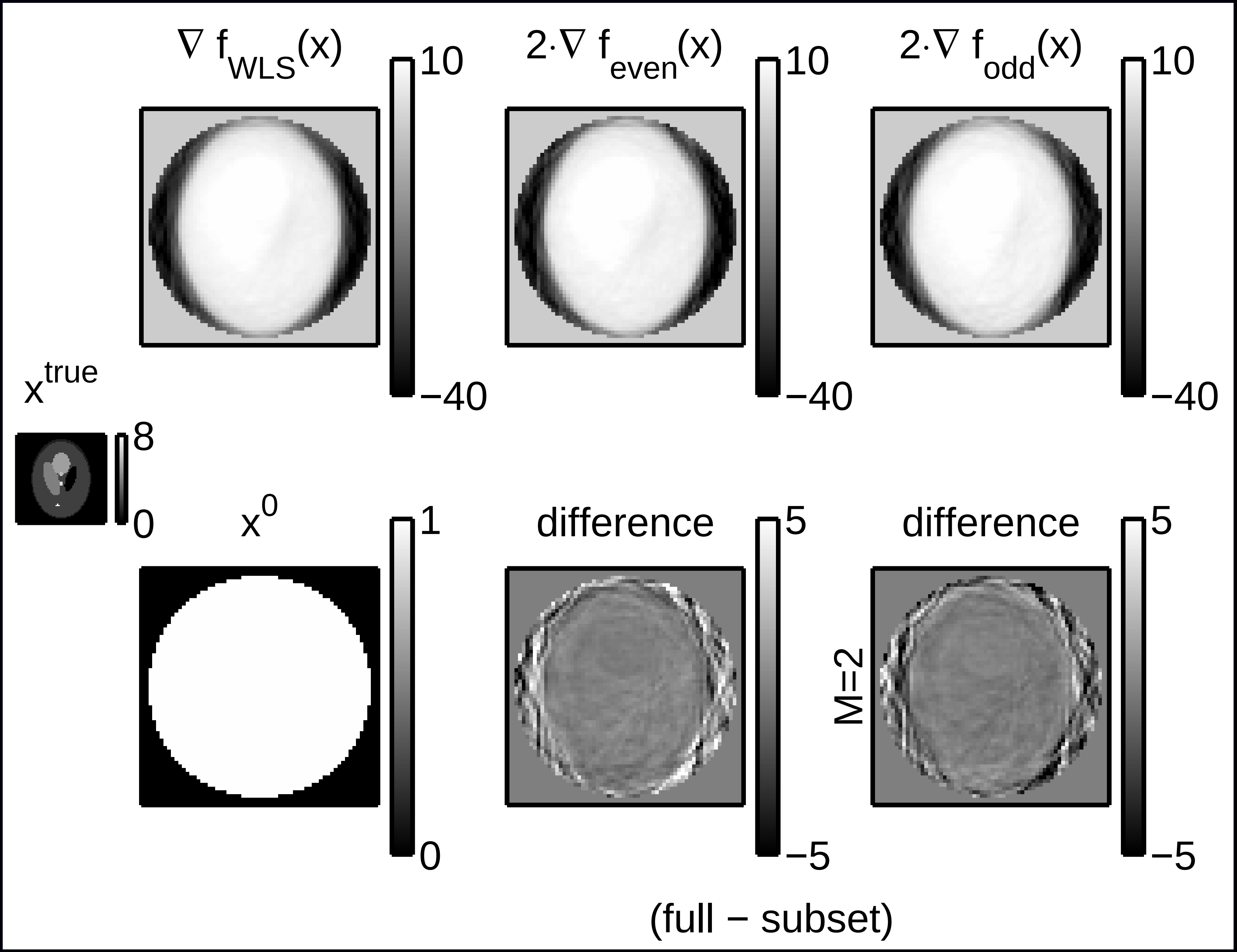


$\mathbf{x}^n$
$\nabla f_1(\mathbf{x}^k)$
$\nabla \Phi(\mathbf{x}^k)$
$\nabla f_2(\mathbf{x}^k)$
$\nabla f_1(\mathbf{x}^n)$
$\nabla f_2(\mathbf{x}^n)$
$\nabla \Phi(\mathbf{x}^n)$
$\mathbf{x}^k$
$\arg\max f_1(\mathbf{x})$
$\mathbf{x}^*$
$\arg\max f_2(\mathbf{x})$

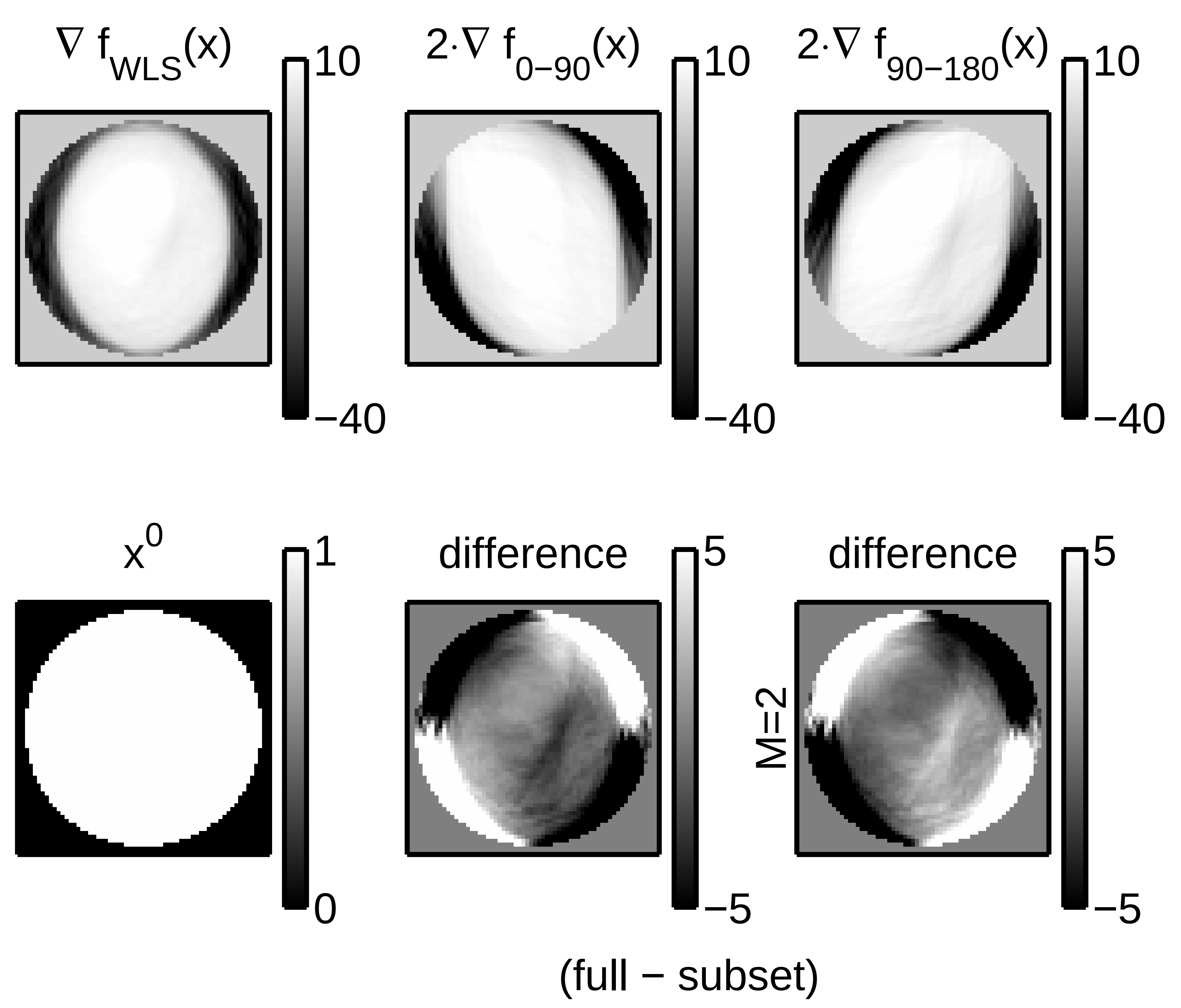
Two subset case: $\Psi(\boldsymbol{x}) = f_1(\boldsymbol{x}) + f_2(\boldsymbol{x})$ (*e.g.*, odd and even projection views).

For $\boldsymbol{x}^{(n)}$ far from $\boldsymbol{x}^\star$, even partial gradients should point roughly towards $\boldsymbol{x}^\star$.
For $\boldsymbol{x}^{(n)}$ near $\boldsymbol{x}^\star$, however, $\nabla \Psi(\boldsymbol{x}) \approx \mathbf{0}$, so $\nabla f_1(\boldsymbol{x}) \approx -\nabla f_2(\boldsymbol{x}) \Longrightarrow$ cycles!
Issues. “Subset gradient balance”: $\nabla \Psi(\boldsymbol{x}) \approx M \nabla f_k(\boldsymbol{x})$. Choice of ordering.

Incremental Gradients (WLS, 2 Subsets)


$\nabla f_{\mathrm{WLS}}(x)$
10
−40
$2 \cdot \nabla f_{\mathrm{even}}(x)$
10
−40
$2 \cdot \nabla f_{\mathrm{odd}}(x)$
10
−40
x^{true}
8
0
x^0
1
0
difference
5
−5
M=2
difference
5
−5
(full − subset)

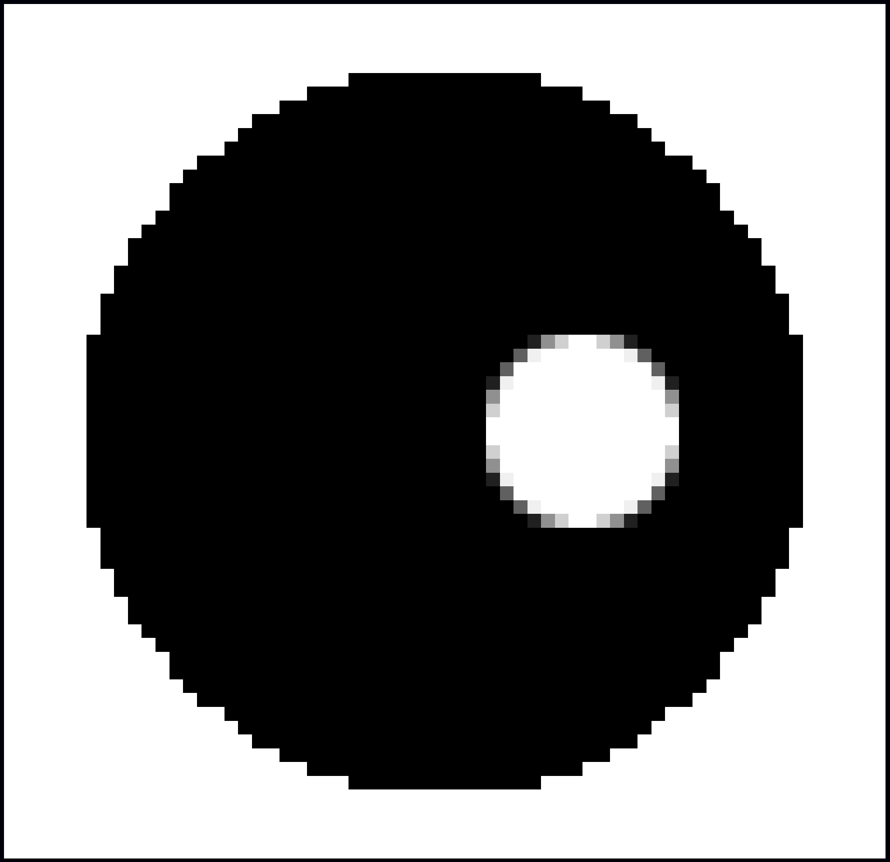
Subset Gradient Imbalance



Problems with OS-EM

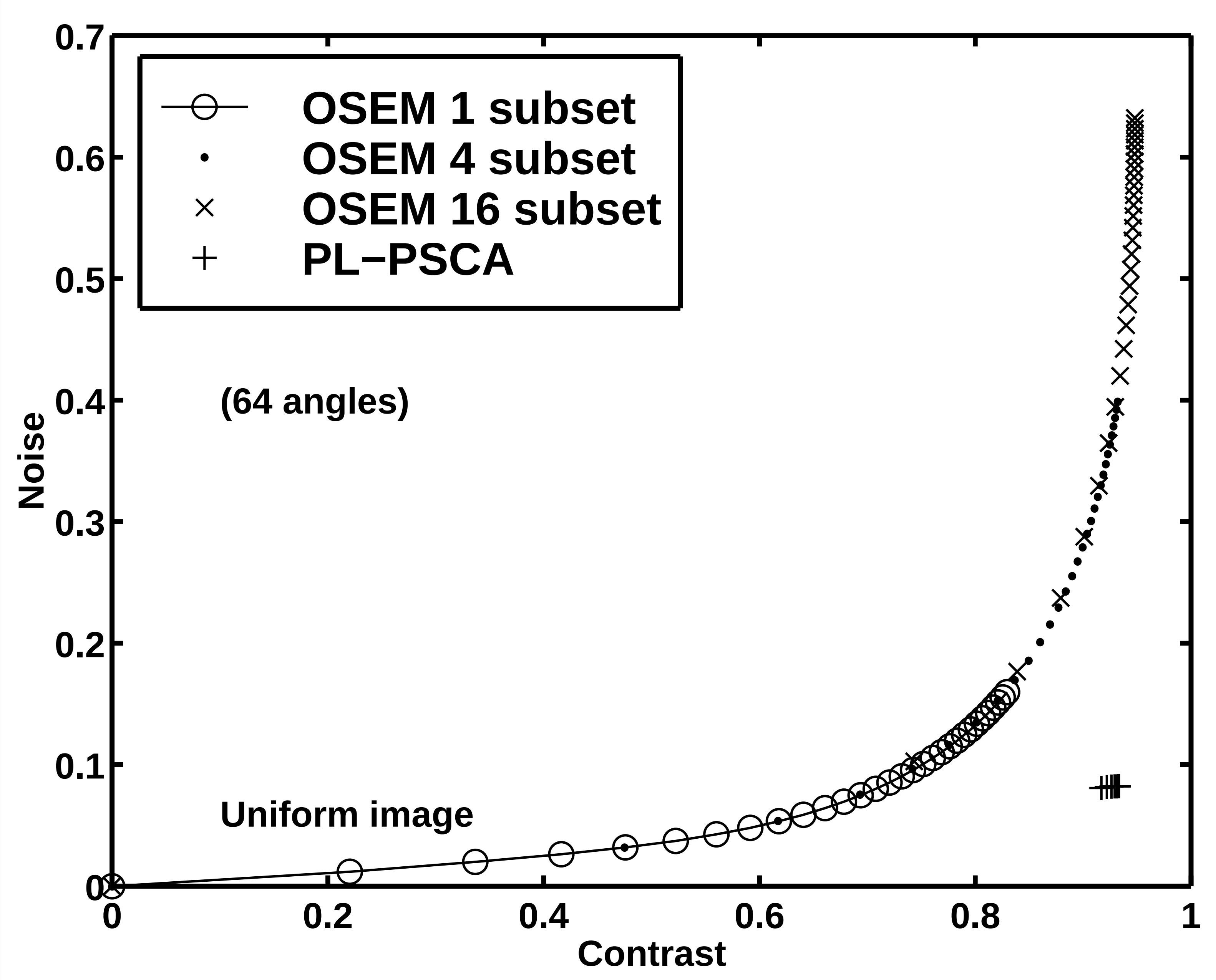
- Non-monotone
- Does not converge (may cycle)
- Byrne's "rescaled block iterative" (RBI) approach converges only for consistent (noiseless) data
- $\therefore$ unpredictable
  - What resolution after n iterations?
    Object-dependent, spatially nonuniform
  - What variance after n iterations?
  - ROI variance? (*e.g.*, for Huesman's WLS kinetics)

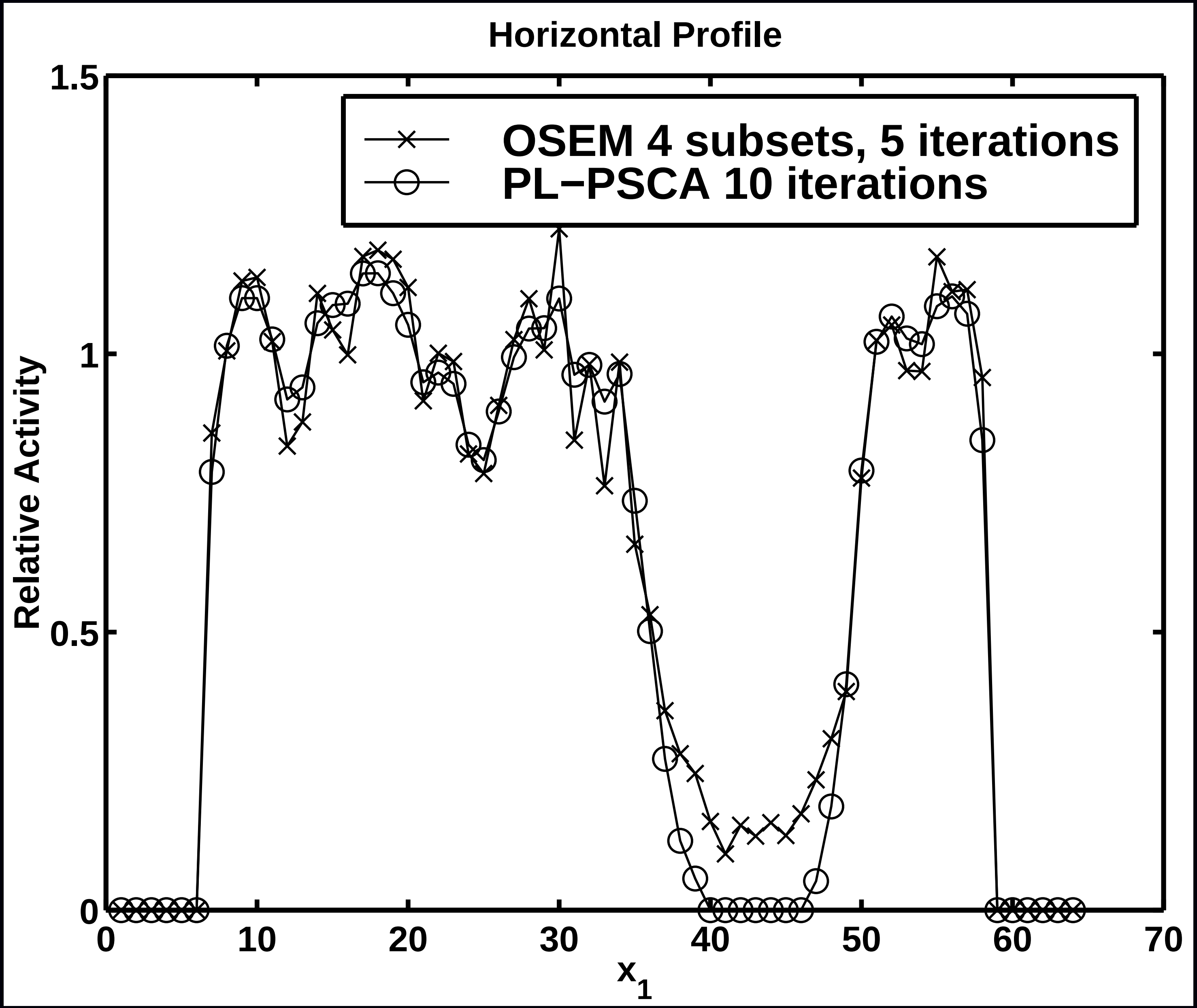
OSEM vs Penalized Likelihood



- 64×62 image
- 66×60 sinogram
- 10^6 counts
- 15% randoms/scatter
- uniform attenuation
- contrast in cold region
- within-region σ opposite side

Contrast-Noise Results




Horizontal Profile
Relative Activity
x₁
0
0.5
1
1.5
0
10
20
30
40
50
60
70
OSEM 4 subsets, 5 iterations
PL−PSCA 10 iterations

Making OS Methods Converge

- Relaxation
- Incrementalism

Relaxed block-iterative methods

$$\Psi(\boldsymbol{x}) = \sum_{m=1}^{M} \Psi_m(\boldsymbol{x})$$

$$\boldsymbol{x}^{(n+(m+1)/M)} = \boldsymbol{x}^{(n+m/M)} - \alpha_n D(\boldsymbol{x}^{(n+m/M)}) \nabla \Psi_m \left(\boldsymbol{x}^{(n+m/M)}\right), \qquad m = 0, \ldots, M-1$$

Relaxation of step sizes:

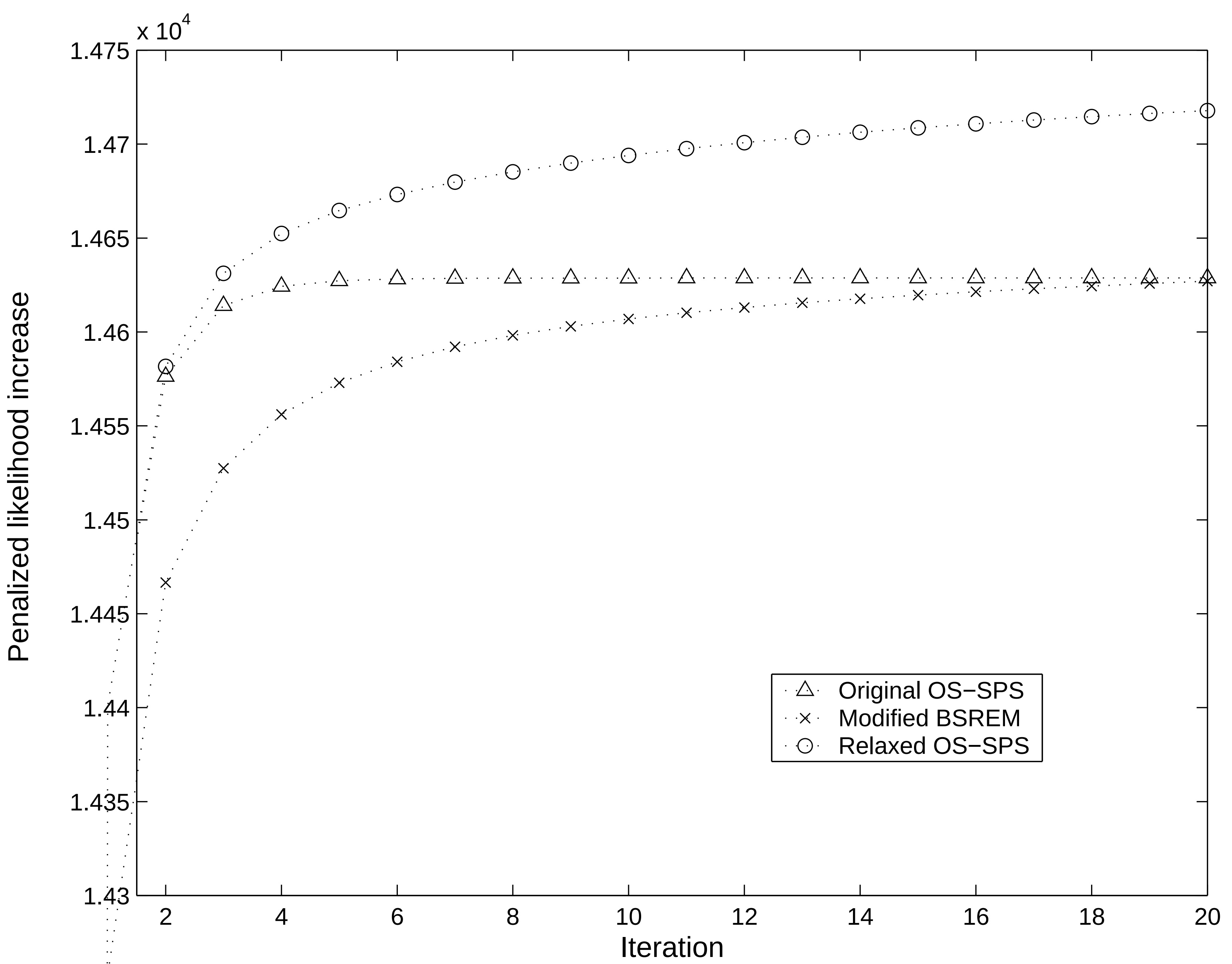
$$\alpha_n \to 0 \text{ as } n \to \infty, \qquad \sum_n \alpha_n = \infty, \qquad \sum_n \alpha_n^2 < \infty$$

- ART
- RAMLA, BSREM (De Pierro, T-MI, 1997, 2001)
- Ahn and Fessler, NSS/MIC 2001, T-MI 2003

Considerations

- Proper relaxation can induce convergence, *but* still lacks monotonicity.
- Choice of relaxation schedule requires experimentation.
- $\Psi_m(\boldsymbol{x}) = \text{Ł}_m(\boldsymbol{x}) + \frac{1}{M}\mathrm{R}(\boldsymbol{x})$, so each Ψ_m includes part of the likelihood yet all of R

Relaxed OS-SPS



Incremental Methods

Incremental EM applied to emission tomography by Hsiao *et al.* as C-OSEM

Incremental optimization transfer (Ahn & Fessler, MIC 2004)

Find majorizing surrogate for each sub-objective function:

$$\begin{aligned} \phi_m(\boldsymbol{x};\boldsymbol{x}) &= \Psi_m(\boldsymbol{x}), && \forall \boldsymbol{x} \\ \phi_m(\boldsymbol{x};\bar{\boldsymbol{x}}) &\geq \Psi_m(\boldsymbol{x}), && \forall \boldsymbol{x}, \bar{\boldsymbol{x}} \end{aligned}$$

Define the following augmented cost function: $F(\boldsymbol{x};\bar{\boldsymbol{x}}_1,\ldots,\bar{\boldsymbol{x}}_M) = \sum_{m=1}^M \phi_m(\boldsymbol{x};\bar{\boldsymbol{x}}_m)$.
Fact: by construction $\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \Psi(\boldsymbol{x}) = \arg\min_{\boldsymbol{x}} \min_{\bar{\boldsymbol{x}}_1,\ldots,\bar{\boldsymbol{x}}_M} F(\boldsymbol{x};\bar{\boldsymbol{x}}_1,\ldots,\bar{\boldsymbol{x}}_M)$.

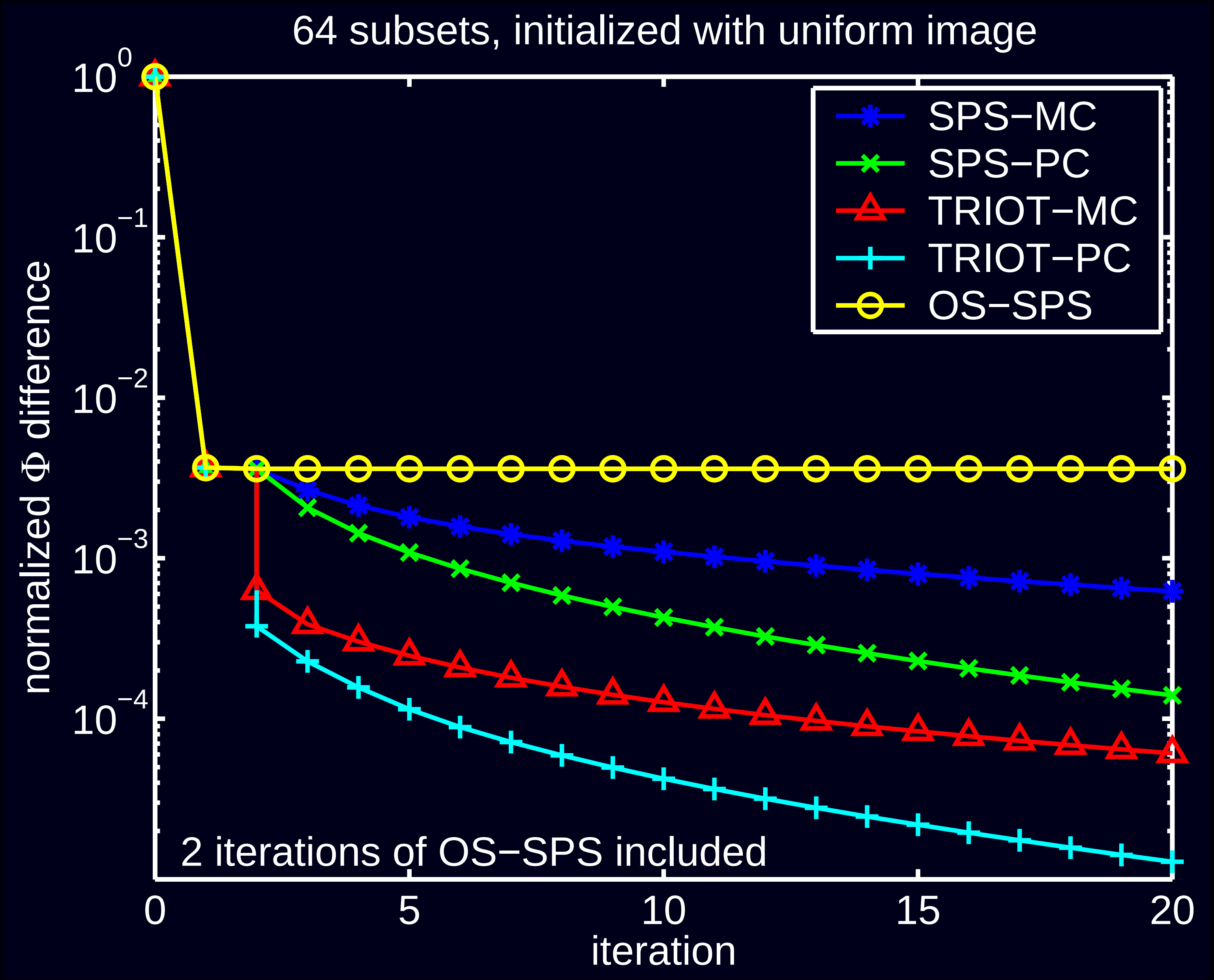
Alternating minimization: `for` $m = 1,\ldots,M$:

$$\begin{aligned} \boldsymbol{x}^{\text{new}} &= \arg\min_{\boldsymbol{x}} F\left(\boldsymbol{x};\bar{\boldsymbol{x}}_1^{(n+1)},\ldots,\bar{\boldsymbol{x}}_{m-1}^{(n+1)},\bar{\boldsymbol{x}}_m^{(n)},\bar{\boldsymbol{x}}_{m+1}^{(n)},\ldots\bar{\boldsymbol{x}}_M^{(n)}\right) \\ \bar{\boldsymbol{x}}_m^{(n+1)} &= \arg\min_{\bar{\boldsymbol{x}}_m} F\left(\boldsymbol{x}^{\text{new}};\bar{\boldsymbol{x}}_1^{(n+1)},\ldots,\bar{\boldsymbol{x}}_{m-1}^{(n+1)},\bar{\boldsymbol{x}}_m,\bar{\boldsymbol{x}}_{m+1}^{(n)},\ldots\bar{\boldsymbol{x}}_M^{(n)}\right) = \boldsymbol{x}^{\text{new}}. \end{aligned}$$

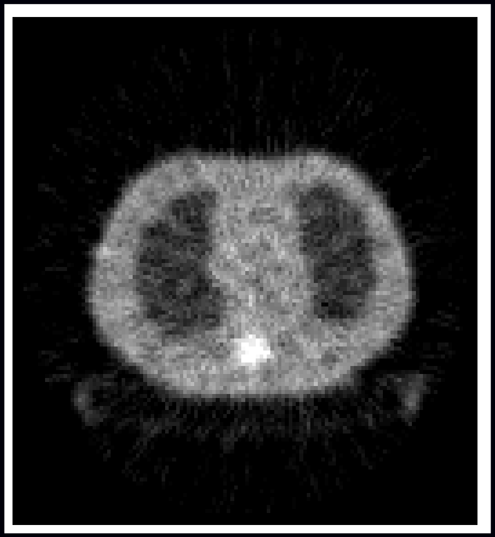
- *Use all current information, but increment the surrogate for only one subset.*
- Monotone in F, converges under reasonable assumptions on Ψ
- In constrast, OS-EM uses the information from *only* one subset at a time

TRIOT Example: Convergence Rate

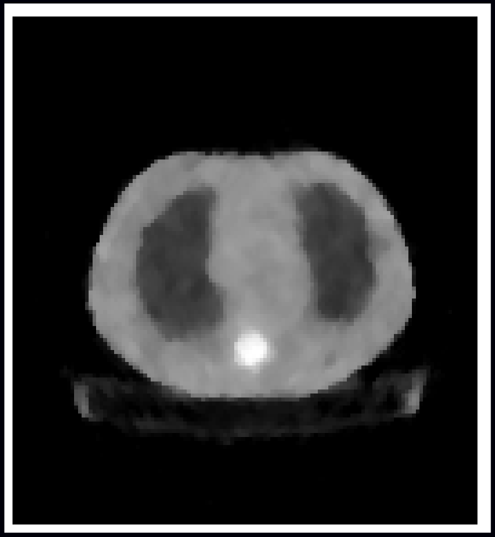
Transmission incremental optimization transfer (TRIOT)



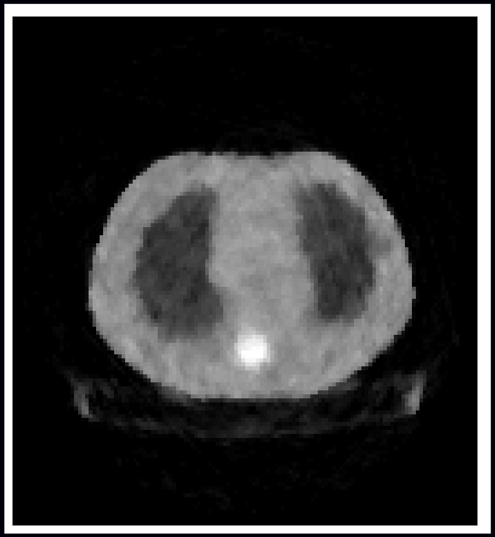
TRIOT Example: Attenuation Map Images



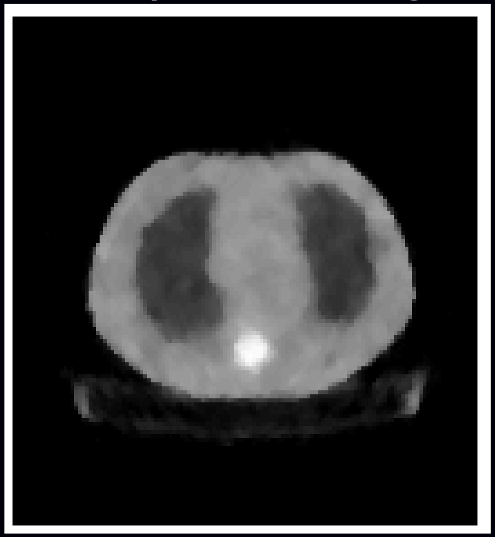
FBP



PL optimal image



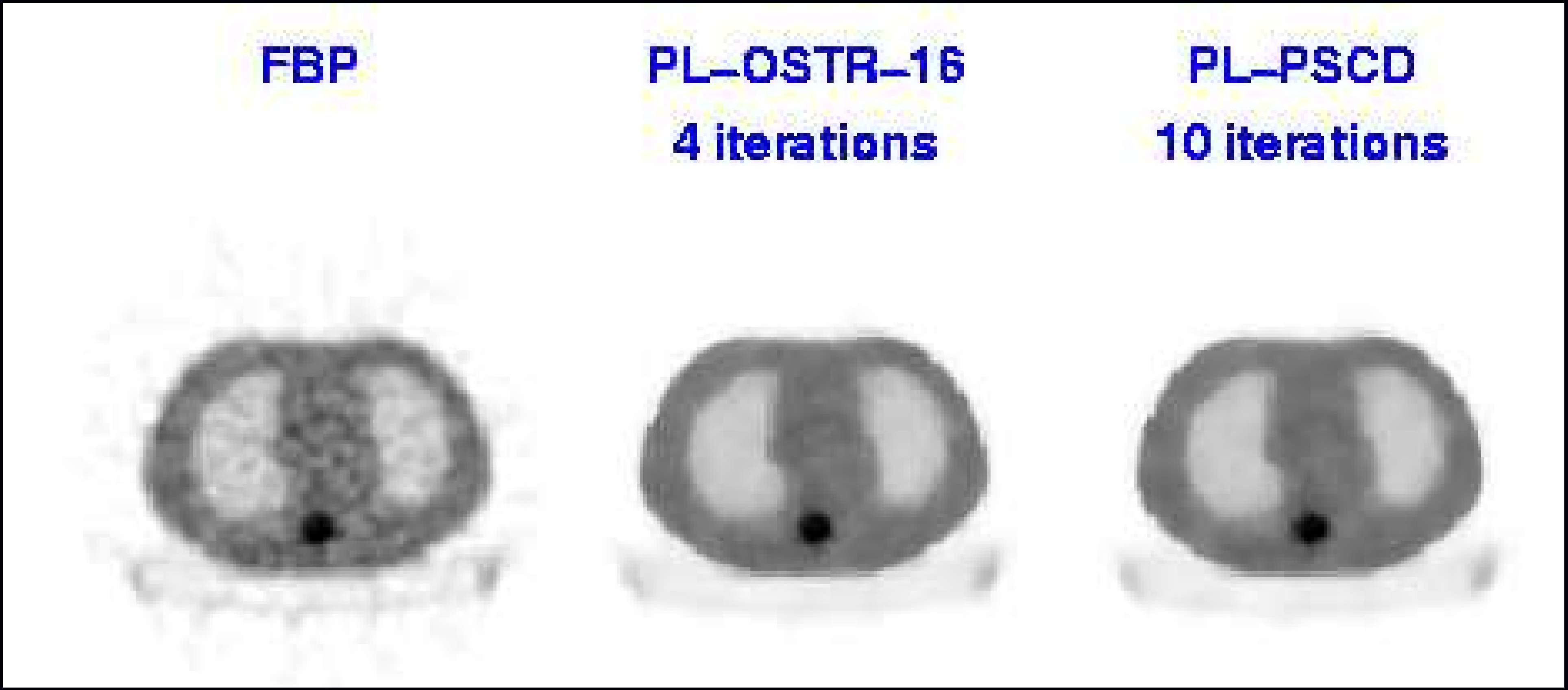
OS-SPS



TRIOT-PC

OS-SPS: 64 subsets, 20 iterations, one point of the limit cycle
TRIOT-PC: 64 subsets, 20 iterations, after 2 iterations of OS-SPS)

OSTR aka Transmission OS-SPS



Ordered subsets version of separable paraboloidal surrogates
for PET transmission problem with nonquadratic convex *regularization*

Matlab m-file `http://www.eecs.umich.edu/~fessler/irt/irt/transmission/tpl_os_sps.m`

Precomputed curvatures for OS-SPS

Separable Paraboloidal Surrogate (SPS) Algorithm:

$$x_j^{(n+1)} = \left[x_j^{(n)} - \frac{\sum_{i=1}^{n_\mathrm{d}} a_{ij} \dot{\mathsf{h}}_i([\boldsymbol{A}\boldsymbol{x}^{(n)}]_i)}{\sum_{i=1}^{n_\mathrm{d}} a_{ij} |a|_i c_i^{(n)}} \right]_+, \qquad j = 1, \ldots, n_\mathrm{p}$$

Ordered-subsets abandons monotonicity, so why use optimal curvatures $c_i^{(n)}$?

Precomputed curvature:

$$c_i = \ddot{\mathsf{h}}_i\left(\hat{l}_i\right), \qquad \hat{l}_i = \arg\min_l \mathsf{h}_i(l)$$

Precomputed denominator (saves one backprojection each iteration!):

$$d_j = \sum_{i=1}^{n_\mathrm{d}} a_{ij} |a|_i c_i, \qquad j = 1, \ldots, n_\mathrm{p}.$$

OS-SPS algorithm with M subsets:

$$x_j^{(n+1)} = \left[x_j^{(n)} - \frac{\sum_{i \in \mathcal{S}^{(n)}} a_{ij} \dot{\mathsf{h}}_i([\boldsymbol{A}\boldsymbol{x}^{(n)}]_i)}{d_j / M} \right]_+, \qquad j = 1, \ldots, n_\mathrm{p}$$

Summary of Algorithms

- General-purpose optimization algorithms
- Optimization transfer for image reconstruction algorithms
- Separable surrogates $\Longrightarrow$ high curvatures $\Longrightarrow$ slow convergence
- Ordered subsets accelerate *initial* convergence
  require relaxation or incrementalism for true convergence
- Principles apply to emission and transmission reconstruction
- Still work to be done...

Matlab/Freemat "image reconstruction toolbox" online:
`http://www.eecs.umich.edu/~fessler/code`

An Open Problem

Still no algorithm with all of the following properties:

- Nonnegativity easy
- Fast converging
- Intrinsically monotone global convergence
- Accepts any type of system matrix
- Parallelizable

Part 4. Performance Characteristics

Easy case: MRI with quadratic regularization

- Spatial resolution properties
- Noise properties

General case

- Spatial resolution properties
- Noise properties
- Detection properties

Regularized Least-Squares Estimation

Estimate object by minimizing a *regularized* cost function:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x} \in \mathbb{C}^{n_p}} \Psi(\boldsymbol{x}), \qquad \Psi(\boldsymbol{x}) = \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2 + \alpha \mathsf{R}(\boldsymbol{x})$$

- data fit term $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2$
  corresponds to negative log-likelihood of Gaussian distribution
- regularizing term $\mathsf{R}(\boldsymbol{x})$ controls noise by penalizing roughness,

$$\textit{e.g.}: \quad \mathsf{R}(\boldsymbol{x}) \approx \int \|\nabla f\|^2 \, d\vec{r}$$

- regularization parameter $\alpha > 0$
  controls tradeoff between spatial resolution and noise
- Equivalent to Bayesian MAP estimation with prior $\propto e^{-\alpha \mathsf{R}(\boldsymbol{x})}$

Complexities:

- choosing $\mathsf{R}(f)$
- choosing α
- computing minimizer rapidly.

Quadratic regularization

1D example: squared differences between neighboring pixel values:

$$\mathsf{R}(f) = \sum_{j=2}^{n_\mathrm{p}} \frac{1}{2} |f_j - f_{j-1}|^2 .$$

In matrix-vector notation, $\mathsf{R}(\boldsymbol{x}) = \frac{1}{2} \|\boldsymbol{C}\boldsymbol{x}\|^2$ where

$$\boldsymbol{C} = \begin{bmatrix} -1 & 1 & 0 & 0 & \dots & 0 \\ 0 & -1 & 1 & 0 & \dots & 0 \\ & & & \ddots & \ddots & \\ 0 & \dots & 0 & 0 & -1 & 1 \end{bmatrix}, \text{ so } \boldsymbol{C}\boldsymbol{x} = \begin{bmatrix} x_2 - x_1 \\ \vdots \\ x_N - x_{N-1} \end{bmatrix}.$$

For 2D and higher-order differences, modify differencing matrix $\boldsymbol{C}$.

Leads to closed-form solution:

$$\begin{aligned} \hat{\boldsymbol{x}} &= \arg\min_{\boldsymbol{x}} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2 + \alpha \|\boldsymbol{C}\boldsymbol{x}\|^2 \\ &= \left[\boldsymbol{A}'\boldsymbol{A} + \alpha \boldsymbol{C}'\boldsymbol{C}\right]^{-1} \boldsymbol{A}'\boldsymbol{y}. \end{aligned}$$

(a formula of limited practical use for computing $\hat{\boldsymbol{x}}$)

Choosing the Regularization Parameter

Spatial resolution analysis (Fessler & Rogers, IEEE T-IP, 1996):

$$\hat{\boldsymbol{x}} = \left[\boldsymbol{A}'\boldsymbol{A} + \alpha \boldsymbol{C}'\boldsymbol{C}\right]^{-1} \boldsymbol{A}'\boldsymbol{y}$$

$$\mathsf{E}[\hat{\boldsymbol{x}}] = \left[\boldsymbol{A}'\boldsymbol{A} + \alpha \boldsymbol{C}'\boldsymbol{C}\right]^{-1} \boldsymbol{A}'\,\mathsf{E}[\boldsymbol{y}]$$

$$\mathsf{E}[\hat{\boldsymbol{x}}] = \underbrace{\left[\boldsymbol{A}'\boldsymbol{A} + \alpha \boldsymbol{C}'\boldsymbol{C}\right]^{-1} \boldsymbol{A}'\boldsymbol{A}}_{\text{blur}}\boldsymbol{x}$$

$\boldsymbol{A}'\boldsymbol{A}$ and $\boldsymbol{C}'\boldsymbol{C}$ are Toeplitz $\Longrightarrow$ blur is approximately shift-invariant.

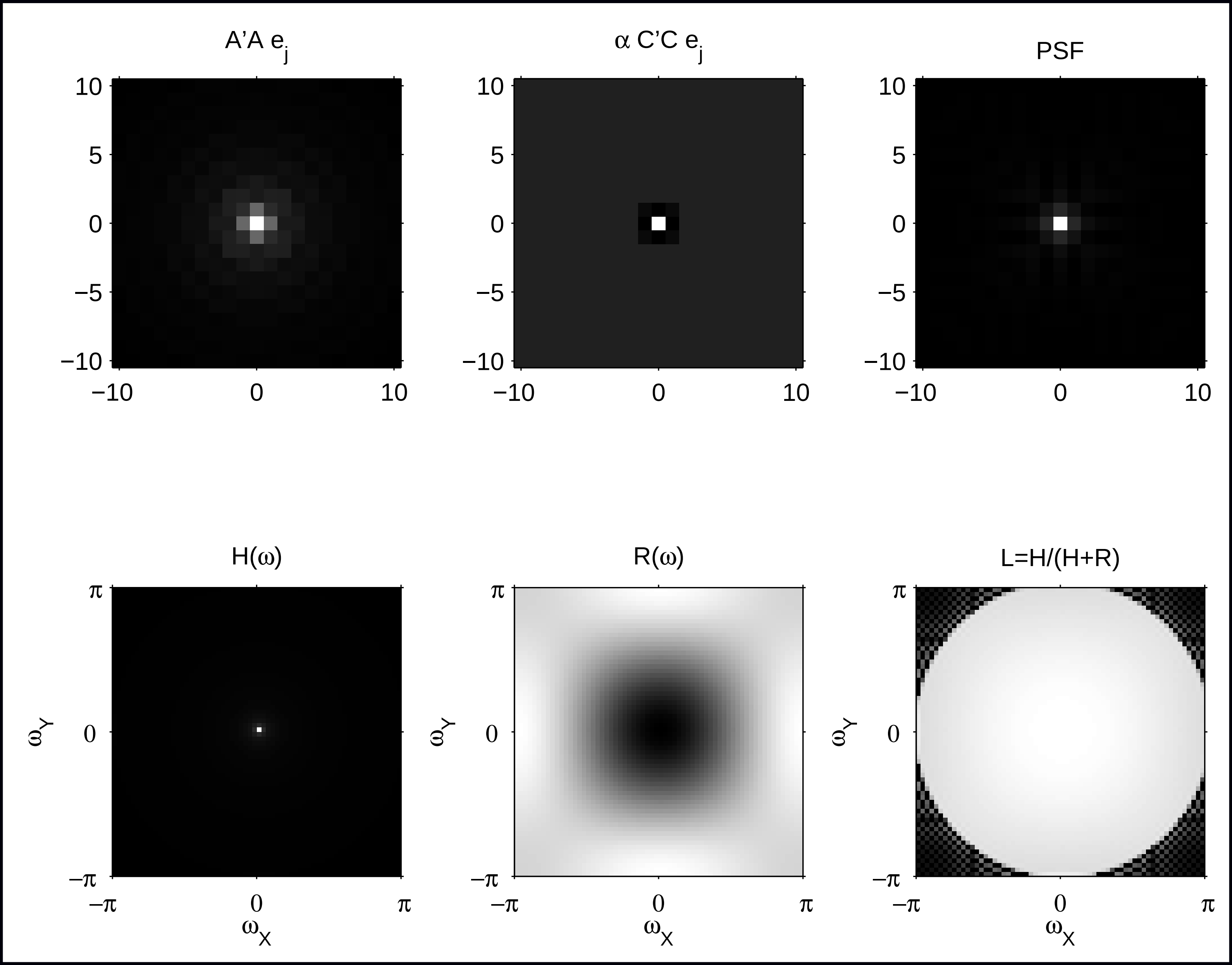
Frequency response of blur:

$$L(\omega) = \frac{H(\omega)}{H(\omega) + \alpha R(\omega)}$$

- $H(\omega_k) = \mathrm{FFT}(\boldsymbol{A}'\boldsymbol{A}\,e_j)$ (lowpass)
- $R(\omega_k) = \mathrm{FFT}(\boldsymbol{C}'\boldsymbol{C}\,e_j)$ (highpass)

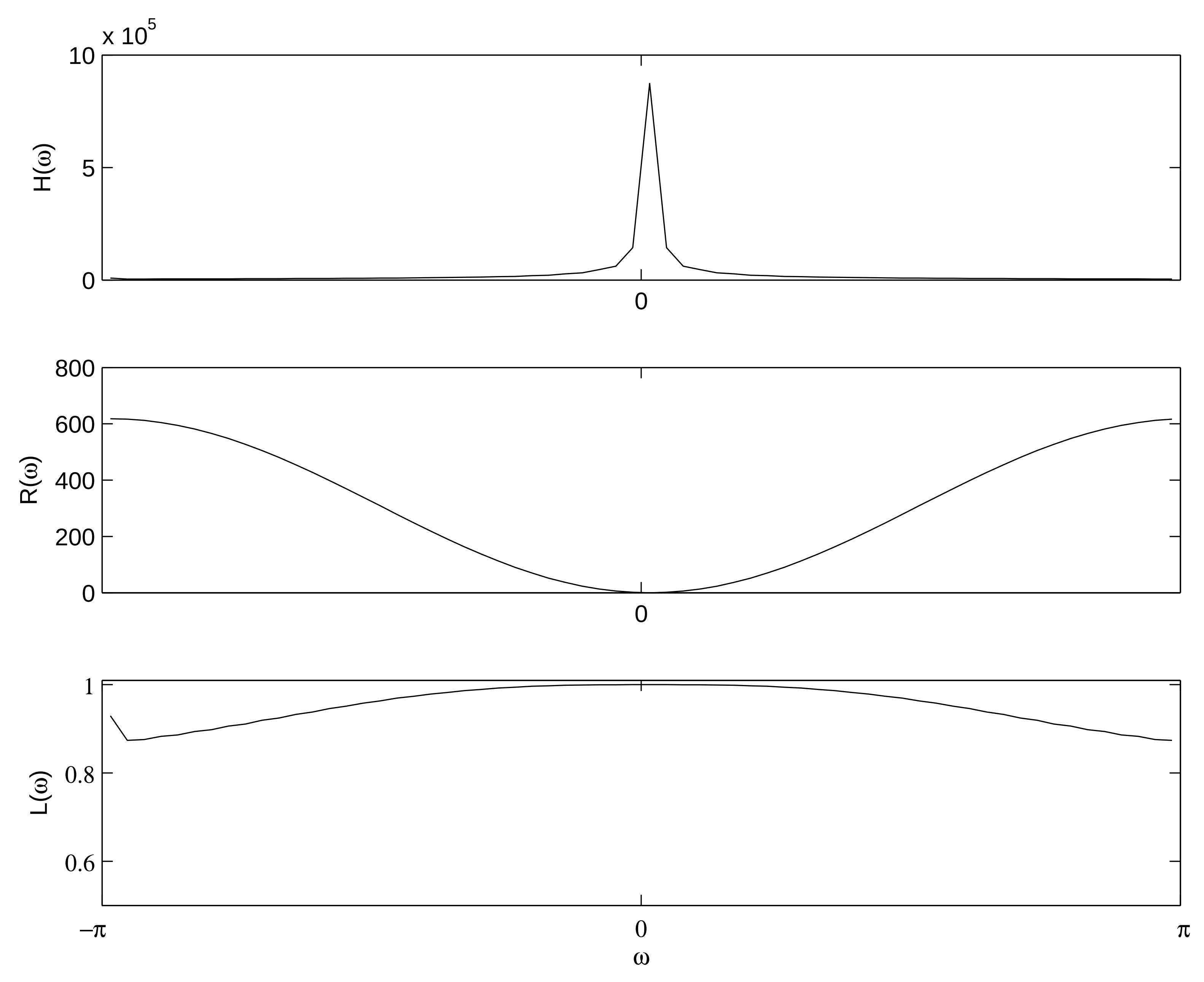
Adjust α to achieve desired spatial resolution.

Spatial Resolution Example

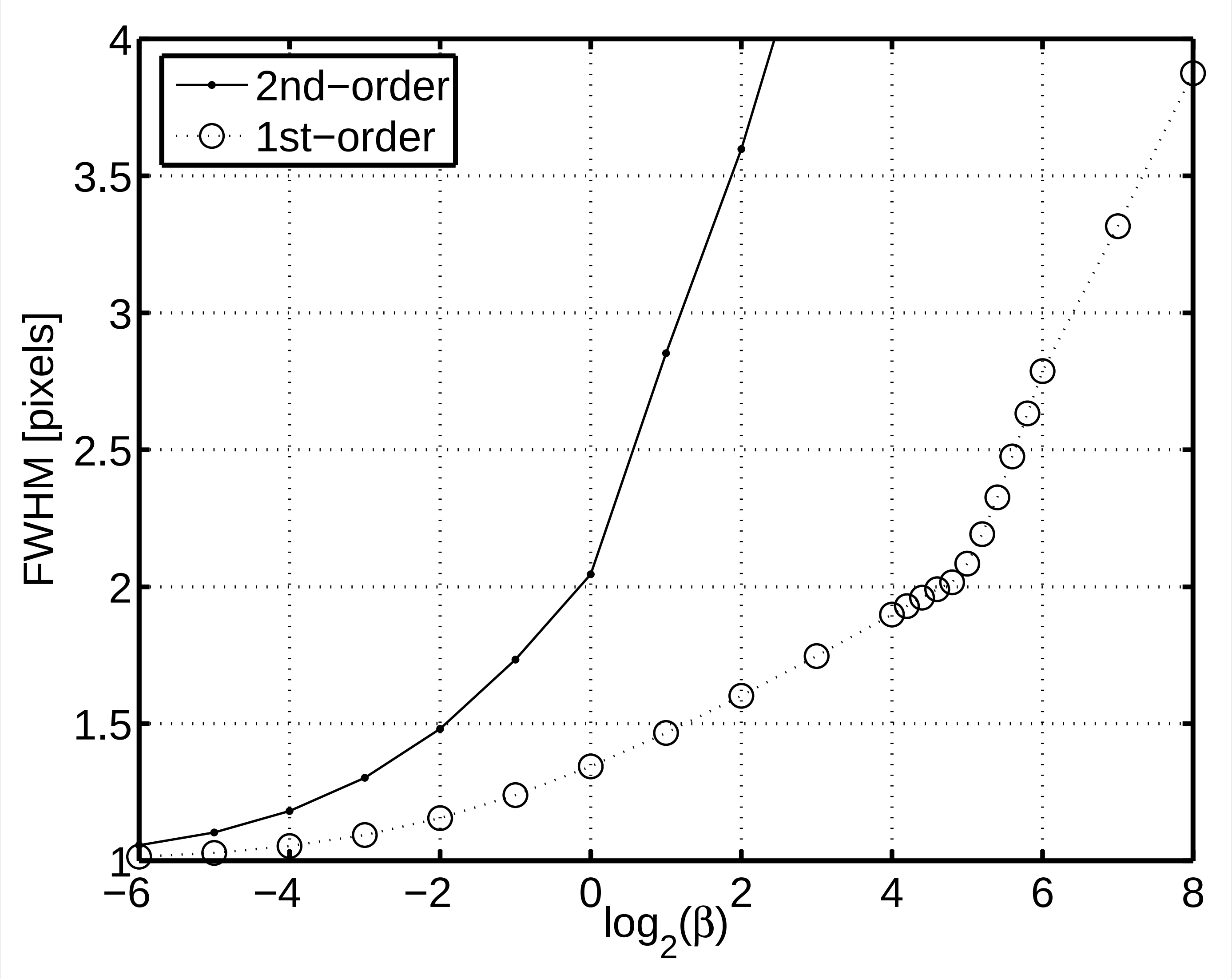


Radial k-space trajectory, FWHM of PSF is 1.2 pixels

Spatial Resolution Example: Profiles



Tabulating Spatial Resolution vs Regularization



Trajectory specific, but easily computed using a few FFTs
Works only for quadratic regularization

Resolution/noise tradeoffs

Noise analysis:

$$\mathsf{Cov}\{\hat{\boldsymbol{x}}\} = \left[\boldsymbol{A}'\boldsymbol{A} + \alpha \boldsymbol{C}'\boldsymbol{C}\right]^{-1} \boldsymbol{A}' \, \mathsf{Cov}\{\boldsymbol{y}\} \, \boldsymbol{A} \left[\boldsymbol{A}'\boldsymbol{A} + \alpha \boldsymbol{C}'\boldsymbol{C}\right]^{-1}$$

Using circulant approximations to $\boldsymbol{A}'\boldsymbol{A}$ and $\boldsymbol{C}'\boldsymbol{C}$ yields:

$$\mathsf{Var}\{\hat{x}_j\} \approx \sigma_\varepsilon^2 \sum_k \frac{H(\omega_k)}{(H(\omega_k) + \alpha R(\omega_k))^2}$$

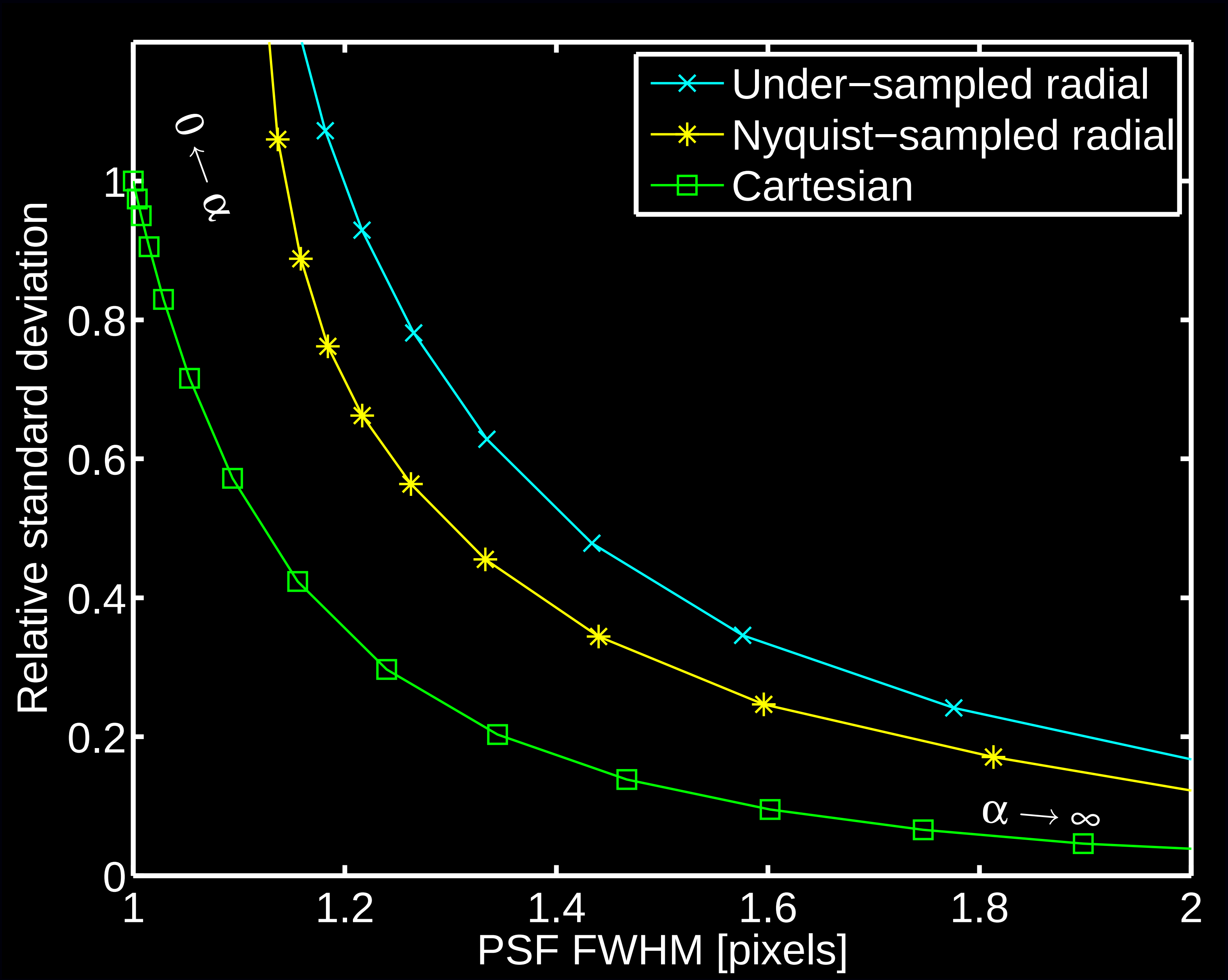
- $H(\omega_k) = \mathrm{FFT}(\boldsymbol{A}'\boldsymbol{A}\, e_j)$ (lowpass)
- $R(\omega_k) = \mathrm{FFT}(\boldsymbol{C}'\boldsymbol{C}\, e_j)$ (highpass)

$\Longrightarrow$ Predicting reconstructed image noise requires just 2 FFTs.

(*cf.* gridding approach?)

Adjust α to achieve desired spatial resolution / noise tradeoff.

Resolution/Noise Tradeoff Example



In short: one can choose α rapidly and predictably for quadratic regularization.

Part 4. Performance Characteristics

(General case)

- Spatial resolution properties
- Noise properties
- Detection properties

Spatial Resolution Properties

Choosing β can be painful, so ...

For true minimization methods:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \Psi(\boldsymbol{x})$$

the *local impulse response* is approximately (Fessler and Rogers, T-MI, 1996):

$$\boldsymbol{l}_j(\boldsymbol{x}) = \lim_{\delta \to 0} \frac{\mathsf{E}[\hat{\boldsymbol{x}} | \boldsymbol{x} + \delta \boldsymbol{e}_j] - \mathsf{E}[\hat{\boldsymbol{x}} | \boldsymbol{x}]}{\delta} \approx \left[-\nabla^{20}\Psi\right]^{-1} \nabla^{11}\Psi \frac{\partial}{\partial x_j} \bar{\boldsymbol{y}}(\boldsymbol{x}).$$

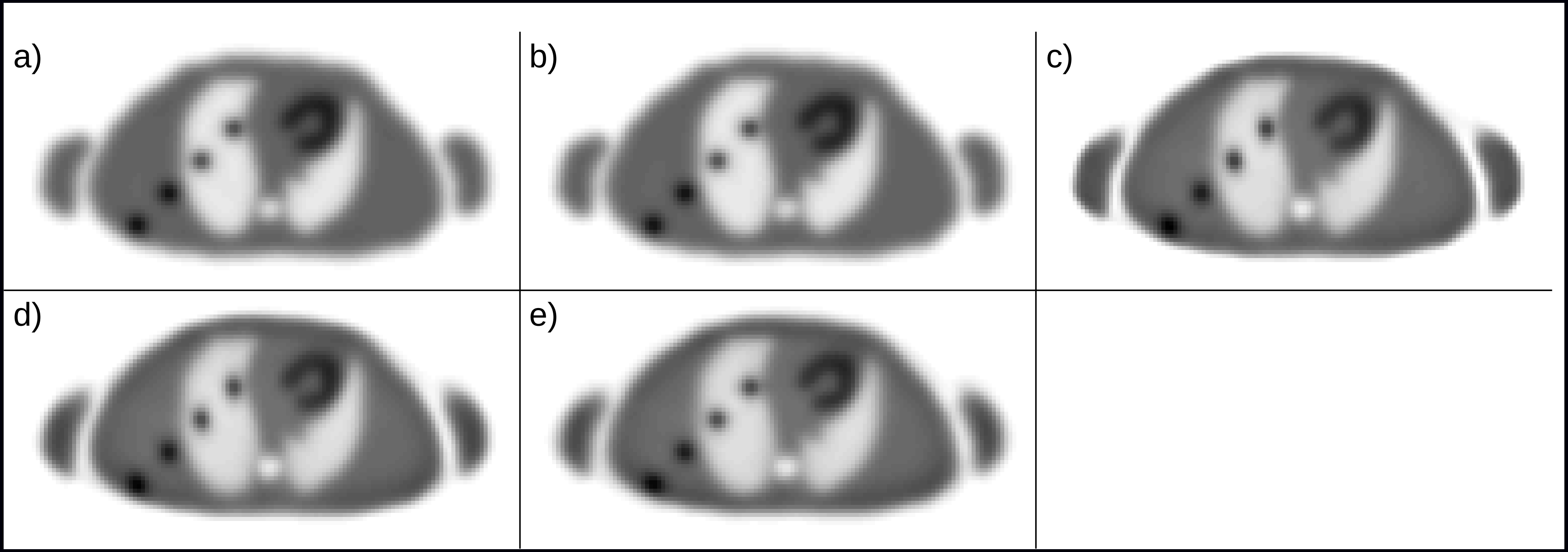
Depends only on chosen cost function and statistical model.
Independent of optimization algorithm (if iterated “to convergence”).

- Enables prediction of resolution properties (provided Ψ is minimized)
- Useful for designing regularization penalty functions with desired resolution properties. For penalized likelihood:

$$\boldsymbol{l}_j(\boldsymbol{x}) \approx [\boldsymbol{A}'\boldsymbol{W}\boldsymbol{A} + \beta \boldsymbol{R}]^{-1} \boldsymbol{A}'\boldsymbol{W}\boldsymbol{A}\boldsymbol{x}^{\text{true}}.$$

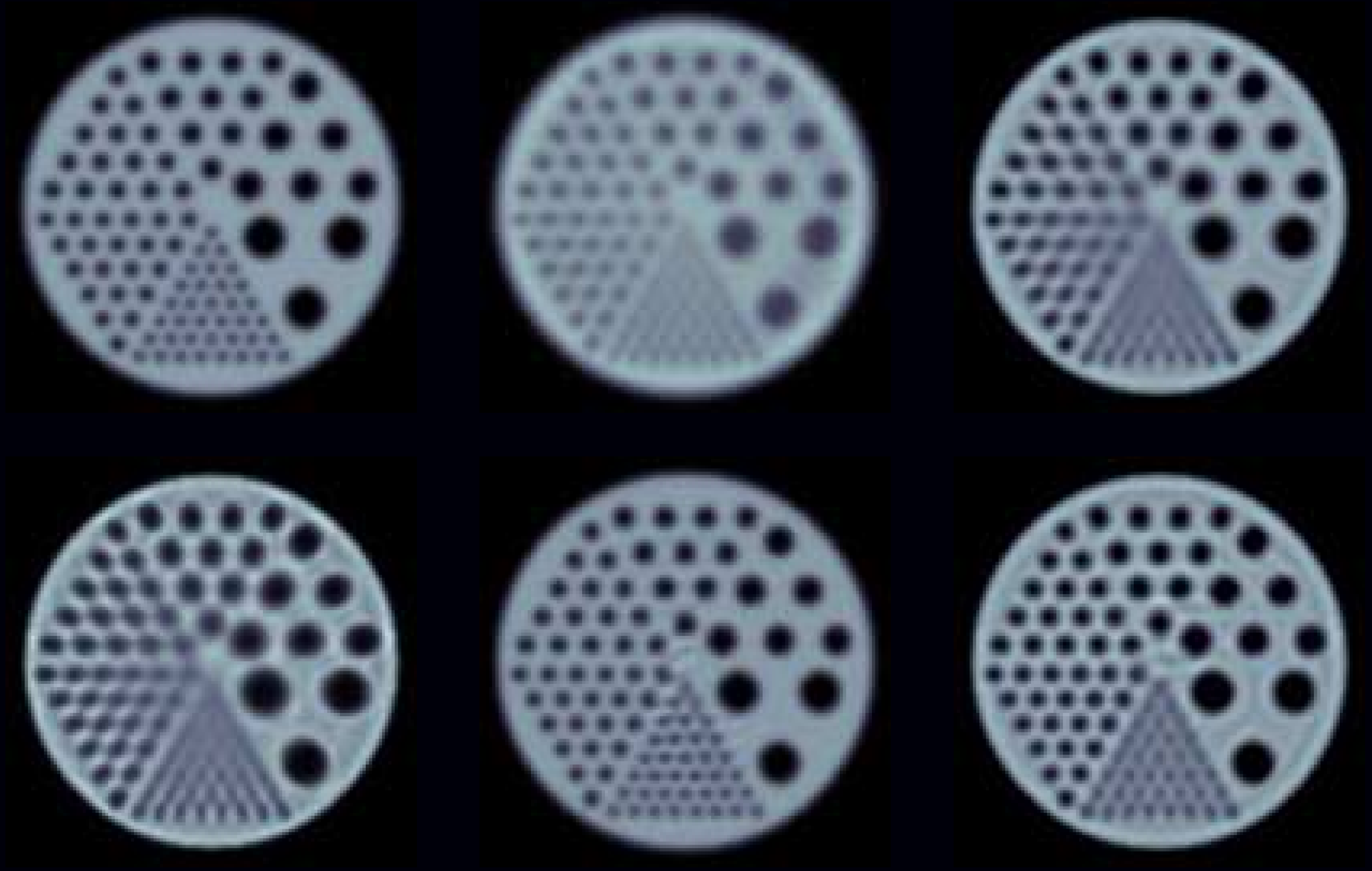
- Helps choose β for desired spatial resolution

Modified Penalty Example, PET



a) filtered backprojection
b) Penalized unweighted least-squares
c) PWLS with conventional regularization
d) PWLS with certainty-based penalty (Fessler & Rogers, 1996, T-MI)
e) PWLS with modified penalty (Stayman & Fessler, 2000, T-MI)

Modified Penalty Example, SPECT - Noiseless



Target filtered object

FBP

Conventional PWLS

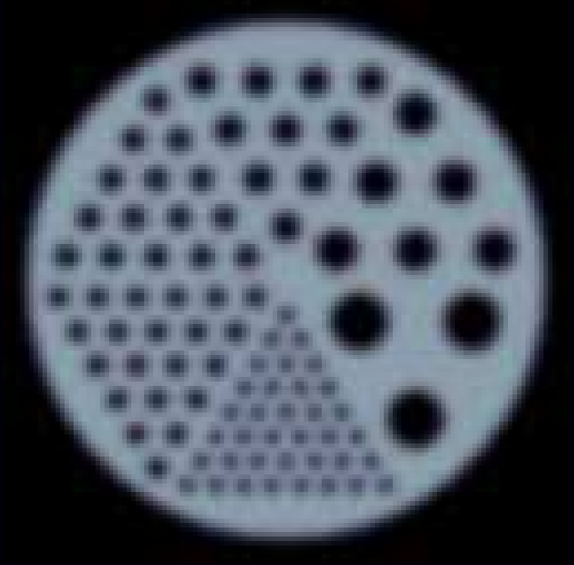
Truncated EM

Post-filtered EM

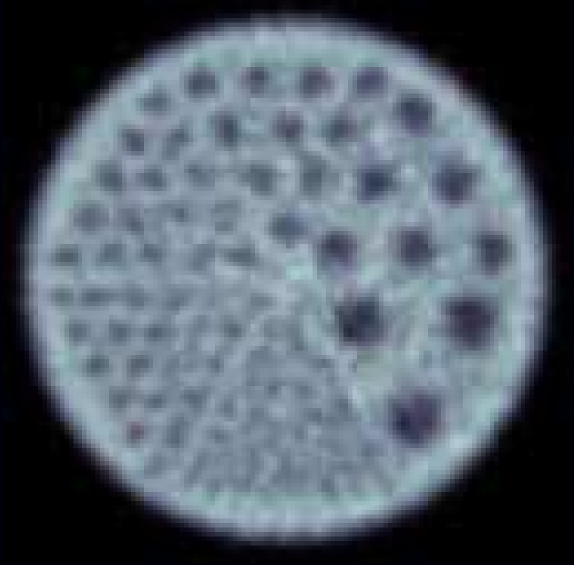
Modified Regularization

Modified Penalty Example, SPECT - Noisy

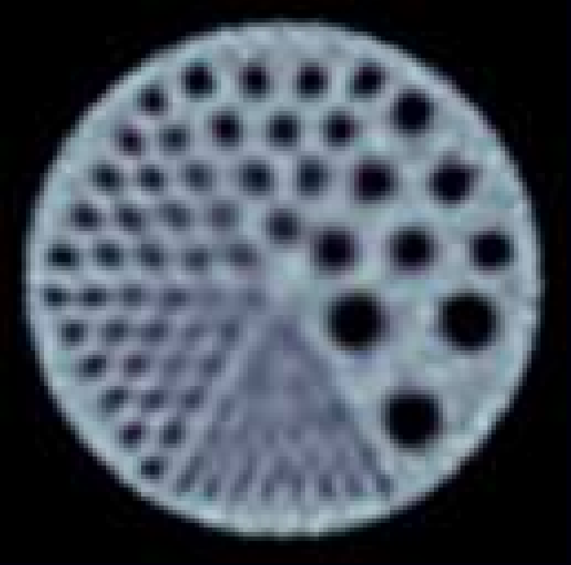
Target filtered object

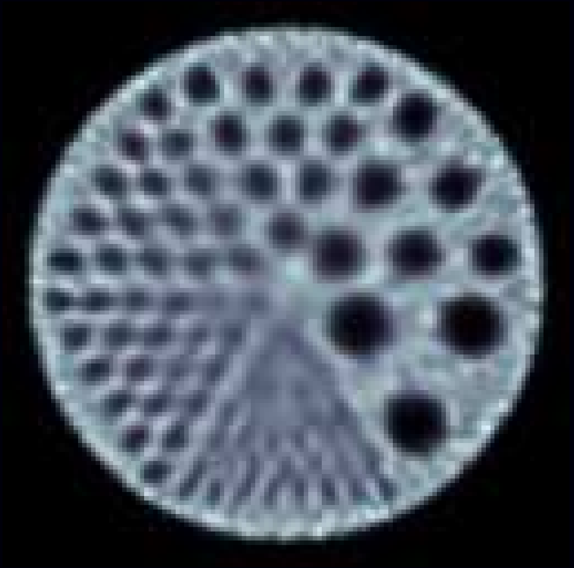


FBP

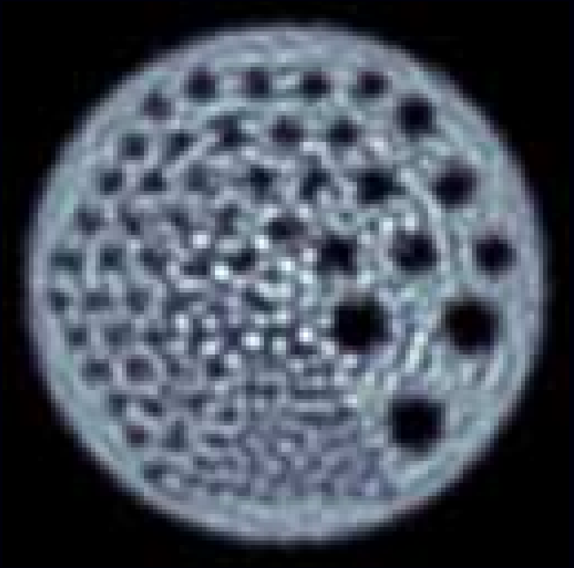


Conventional PWLS

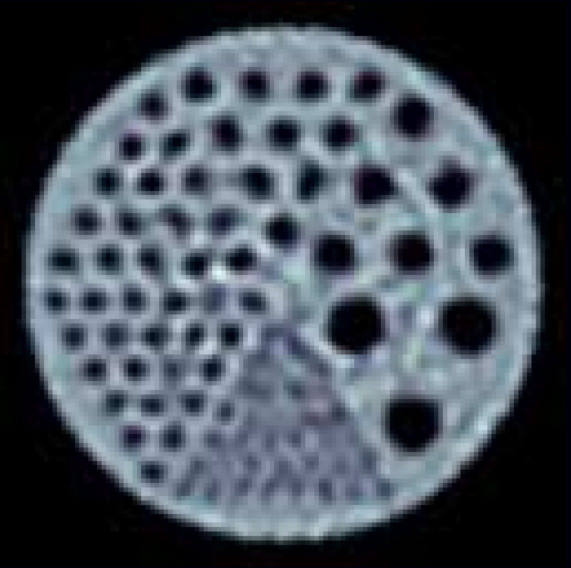




Truncated EM

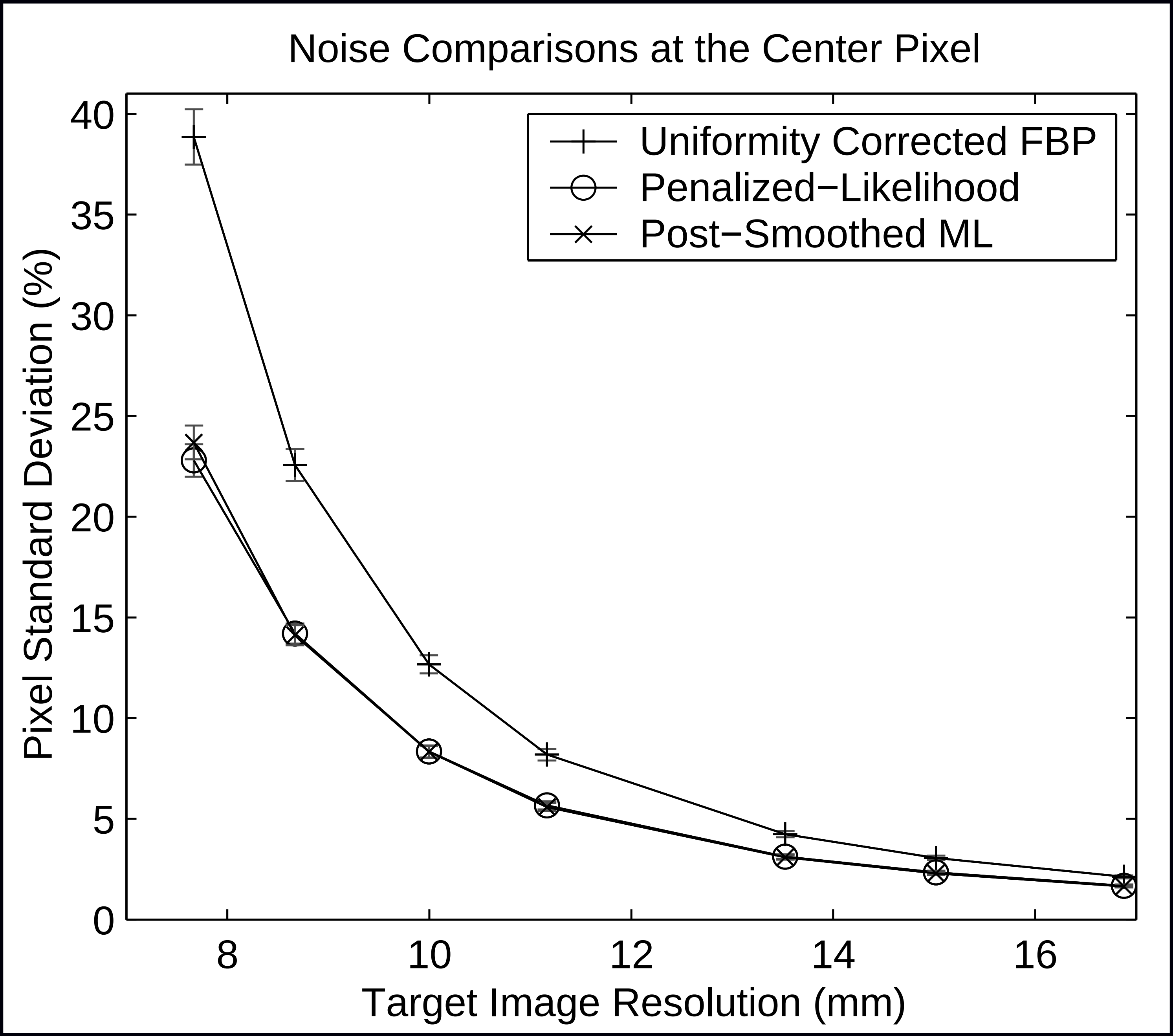


Post-filtered EM



Modified Regularization

Regularized vs Post-filtered, with Matched PSF



Reconstruction Noise Properties

For unconstrained (converged) minimization methods, the estimator is *implicit*:

$$\hat{\boldsymbol{x}} = \hat{\boldsymbol{x}}(\boldsymbol{y}) = \arg\min_{\boldsymbol{x}} \Psi(\boldsymbol{x},\boldsymbol{y}) .$$

What is $\mathsf{Cov}\{\hat{\boldsymbol{x}}\}$? New simpler derivation.

Denote the column gradient by $g(\boldsymbol{x},\boldsymbol{y}) \triangleq \nabla_{\boldsymbol{x}} \Psi(\boldsymbol{x},\boldsymbol{y})$.
Ignoring constraints, the gradient is zero at the minimizer: $g(\hat{\boldsymbol{x}}(\boldsymbol{y}),\boldsymbol{y}) = \boldsymbol{0}$.
First-order Taylor series expansion:

$$\begin{aligned} g(\hat{\boldsymbol{x}},\boldsymbol{y}) &\approx g(\boldsymbol{x}^{\text{true}},\boldsymbol{y}) + \nabla_{\boldsymbol{x}} g(\boldsymbol{x}^{\text{true}},\boldsymbol{y})(\hat{\boldsymbol{x}} - \boldsymbol{x}^{\text{true}}) \\ &= g(\boldsymbol{x}^{\text{true}},\boldsymbol{y}) + \nabla^2_{\boldsymbol{x}} \Psi\left(\boldsymbol{x}^{\text{true}},\boldsymbol{y}\right)(\hat{\boldsymbol{x}} - \boldsymbol{x}^{\text{true}}). \end{aligned}$$

Equating to zero:

$$\hat{\boldsymbol{x}} \approx \boldsymbol{x}^{\text{true}} - \left[\nabla^2_{\boldsymbol{x}} \Psi\left(\boldsymbol{x}^{\text{true}},\boldsymbol{y}\right)\right]^{-1} \nabla_{\boldsymbol{x}} \Psi\left(\boldsymbol{x}^{\text{true}},\boldsymbol{y}\right) .$$

If the Hessian $\nabla^2 \Psi$ is weakly dependent on $\boldsymbol{y}$, then

$$\boxed{\mathsf{Cov}\{\hat{\boldsymbol{x}}\} \approx \left[\nabla^2_{\boldsymbol{x}} \Psi\left(\boldsymbol{x}^{\text{true}},\bar{\boldsymbol{y}}\right)\right]^{-1} \mathsf{Cov}\left\{\nabla_{\boldsymbol{x}} \Psi\left(\boldsymbol{x}^{\text{true}},\boldsymbol{y}\right)\right\} \left[\nabla^2_{\boldsymbol{x}} \Psi\left(\boldsymbol{x}^{\text{true}},\bar{\boldsymbol{y}}\right)\right]^{-1} .}$$

If we further linearize w.r.t. the data: $g(\boldsymbol{x},\boldsymbol{y}) \approx g(\boldsymbol{x},\bar{\boldsymbol{y}}) + \nabla_{\boldsymbol{y}} g(\boldsymbol{x},\bar{\boldsymbol{y}})(\boldsymbol{y} - \bar{\boldsymbol{y}})$, then

$$\mathsf{Cov}\{\hat{\boldsymbol{x}}\} \approx \left[\nabla^2_{\boldsymbol{x}} \Psi\right]^{-1} (\nabla_{\boldsymbol{x}} \nabla_{\boldsymbol{y}} \Psi)\ \mathsf{Cov}\{\boldsymbol{y}\}\ (\nabla_{\boldsymbol{x}} \nabla_{\boldsymbol{y}} \Psi)'\ \left[\nabla^2_{\boldsymbol{x}} \Psi\right]^{-1} .$$

Covariance Continued

Covariance approximation:

$$\mathsf{Cov}\{\hat{\boldsymbol{x}}\} \approx \left[\nabla_{\boldsymbol{x}}^2 \Psi\left(\boldsymbol{x}^{\text{true}}, \bar{\boldsymbol{y}}\right)\right]^{-1} \mathsf{Cov}\left\{\nabla_{\boldsymbol{x}} \Psi\left(\boldsymbol{x}^{\text{true}}, \boldsymbol{y}\right)\right\} \left[\nabla_{\boldsymbol{x}}^2 \Psi\left(\boldsymbol{x}^{\text{true}}, \bar{\boldsymbol{y}}\right)\right]^{-1}$$

Depends only on chosen cost function and statistical model.
Independent of optimization algorithm.

- Enables prediction of noise properties
- Can make variance images
- Useful for computing ROI variance (*e.g.*, for weighted kinetic fitting)
- Good variance prediction for quadratic regularization in nonzero regions
- Inaccurate for nonquadratic penalties, or in nearly-zero regions

Detection Analysis

Qi and Huesman (IEEE T-MI, Aug. 2001) showed analytically:

SNR of MAP reconstruction $>$ SNR of FBP reconstruction

quadratic regularization
SKE/BKE task
prewhitened observer
non-prewhitened observer

Open issues

Choice of regularizer to optimize detectability?
Active work in several groups.

Choosing β: Unknown location

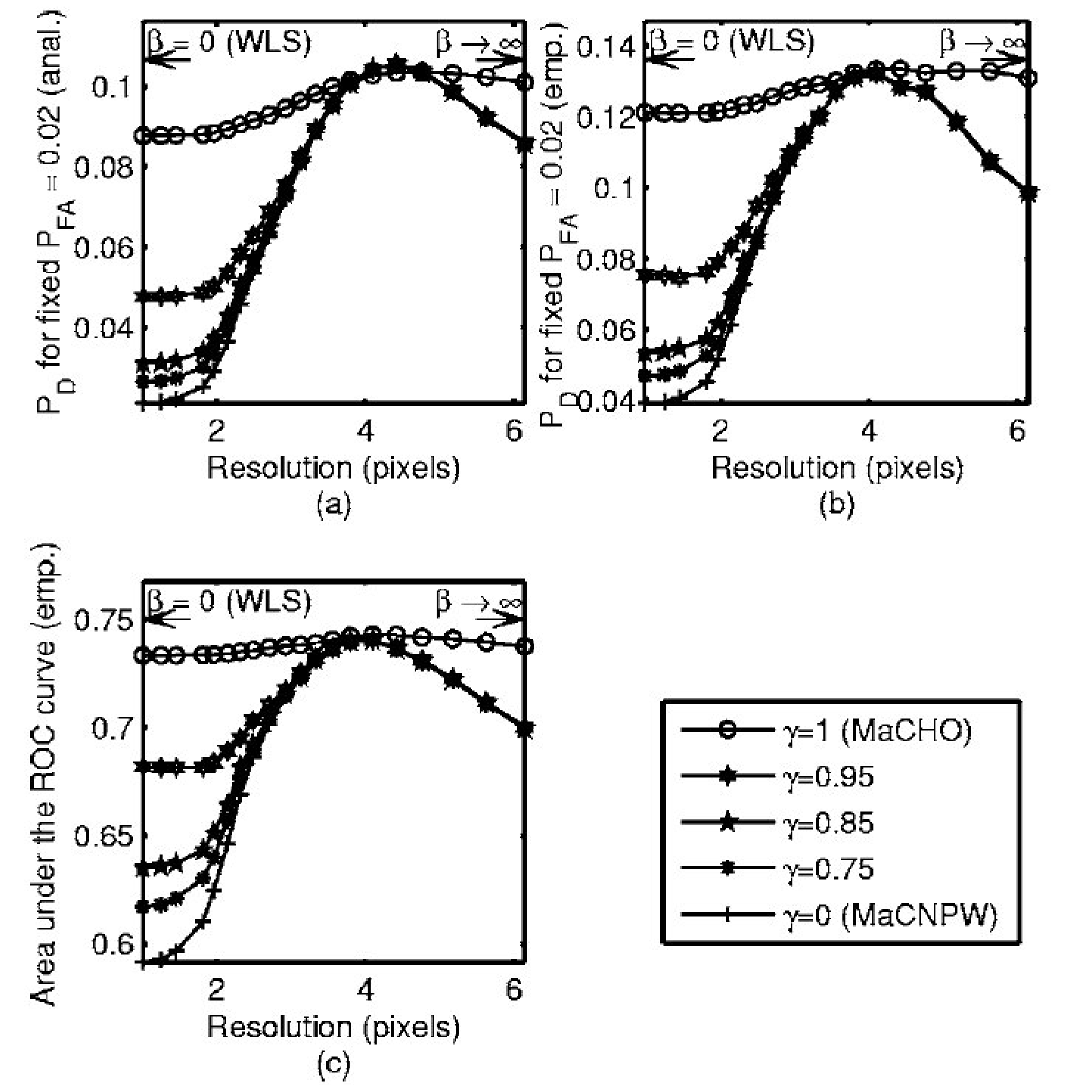


Fig. 2. Detection performance of MaCPPW observers versus QPWLS reconstruction resolution: P_D obtained analytically (a), P_D obtained empirically (b), and AUC obtained empirically (c). Results are shown for five different degrees of prewhitening accuracy. The search area is a disk with a diameter of 9 pixels.

AUC for signal detection with unknown location task.

Yendiki & Fessler, JOSA-A 24(12):B199, Dec. 2007

Summary of Performance Analysis

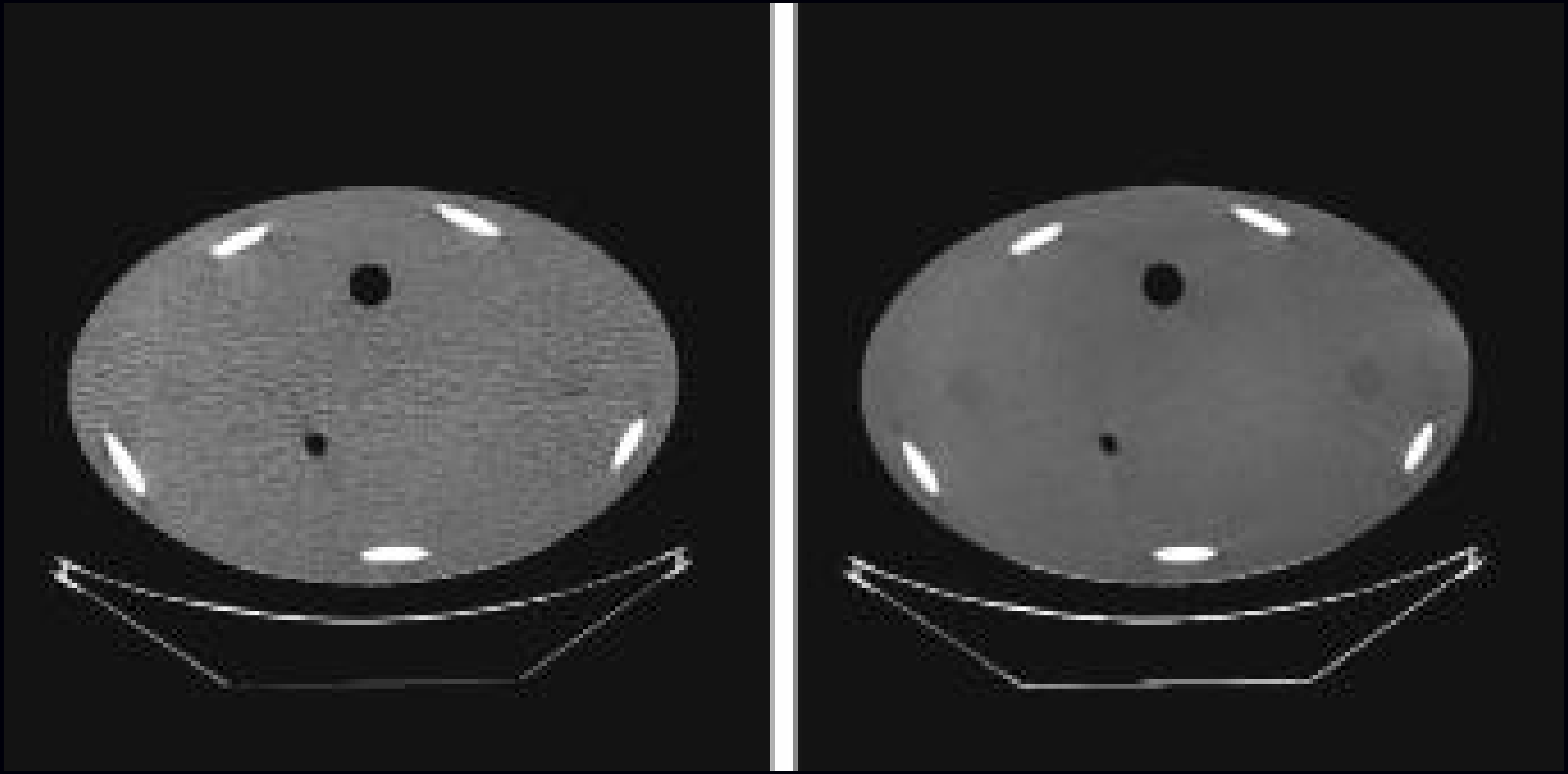
Spatial resolution / noise variance and covariance / AUC for signal detection:
all (somewhat) predictable based on properties of *cost function* Ψ.
(Provided an iterative algorithm is run "to convergence" to find minimizer of Ψ.)

This predictability also motivates regularized cost functions.
(*cf.* unregularized cost function with a stopping rule.)

Part: Application examples

- X-ray CT
- MRI

Example: X-ray Helical CT

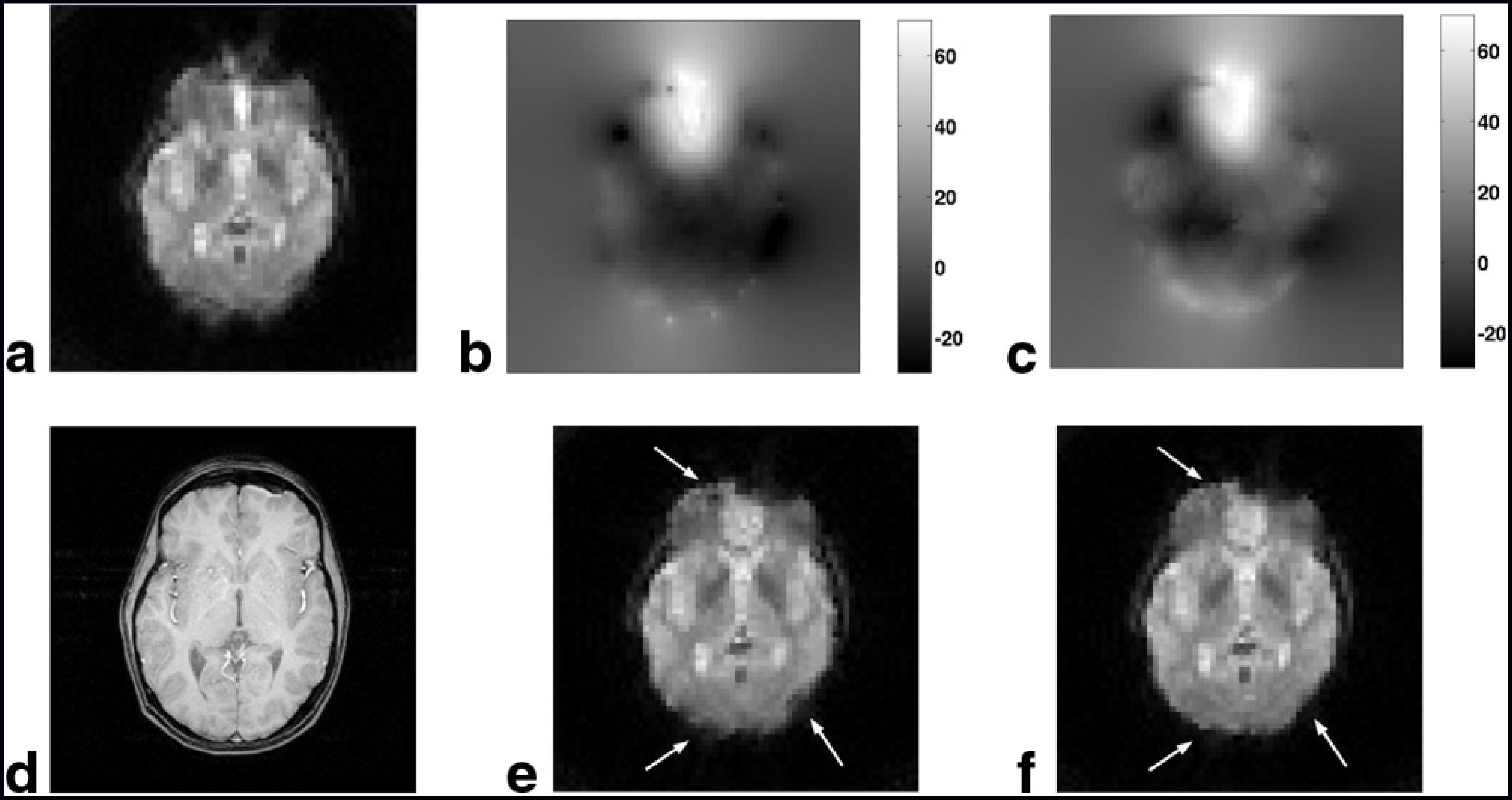


Left: FBP

Right: PWLS-ICD, edge-preserving

Thibault *et al.*, Med Phys. 34(11):4526, Nov. 2007

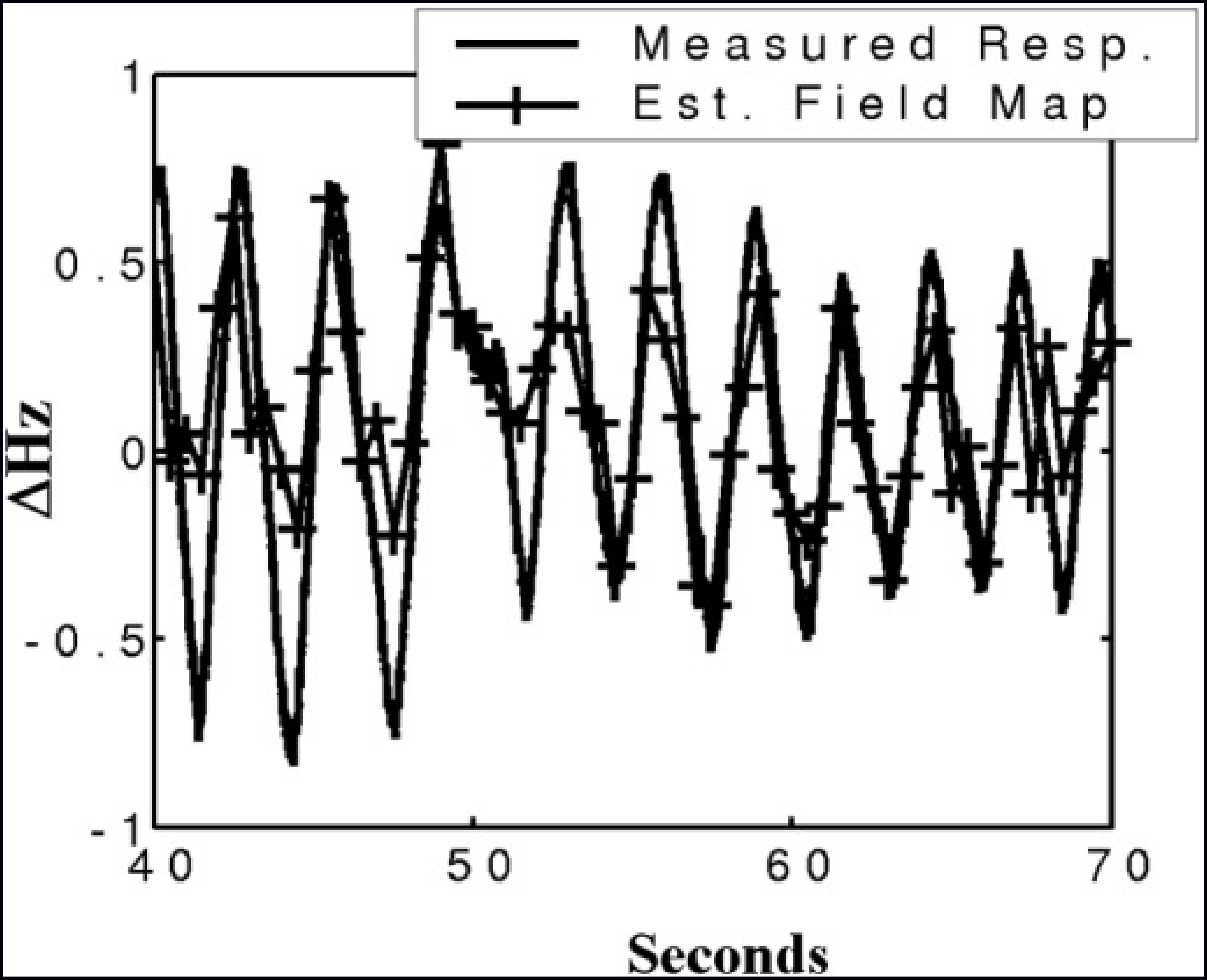
Example: fMRI with Joint Estimation of Fieldmap



(a) uncorr., (b) std. map, (c) joint map, (d) T1 ref, (e) using std, (f) using joint.

PWLS-CG with quadratic regularization. β chosen by PSF analysis.
Sutton *et al.*, MRM 51(6):1194, Jun. 2004

Tracking Respiration-Induced Field Changes



Other Topics

- Dynamic image sequence reconstruction / 4D regularization
- Motion and/or dynamic contrast changes

Summary

- Iterative reconstruction has had clinical impact in PET and SPECT
- MRI and X-ray CT may be next?
- todo: Still work to be done...

References

The literature on image reconstruction is enormous and growing. Many valuable publications are not included in this list, which is not intended to be comprehensive.

Slides and lecture notes available from:

`http://www.eecs.umich.edu/∼fessler`