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: $\{y_i\}_{i=1}^{n_d}$. Continuous object: $f(\vec{r})$. Hopeless?

All models are wrong but some models are useful.

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

$$f(\vec{r}) \approx \tilde{f}(\vec{r}) = \sum_{j=1}^{n_p} x_j b_j(\vec{r}) \leftarrow \text{ "basis functions"}$$

Forward projection:

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

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

- Projection integrals become finite summations.
- a_{ij} is contribution of jth basis function (*e.g.*, voxel) to ith detector unit.
- The units of a_{ij} and x_j depend on the user-selected units of $b_j(\vec{r})$.
- The $n_d \times n_p$ 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), ...

Considerations

- Represent $f(\vec{r})$ “well” with moderate n_p
- Orthogonality? (not essential)
- “Easy” to compute a_{ij}’s and/or $\boldsymbol{A}\boldsymbol{x}$
- Rotational symmetry
- 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

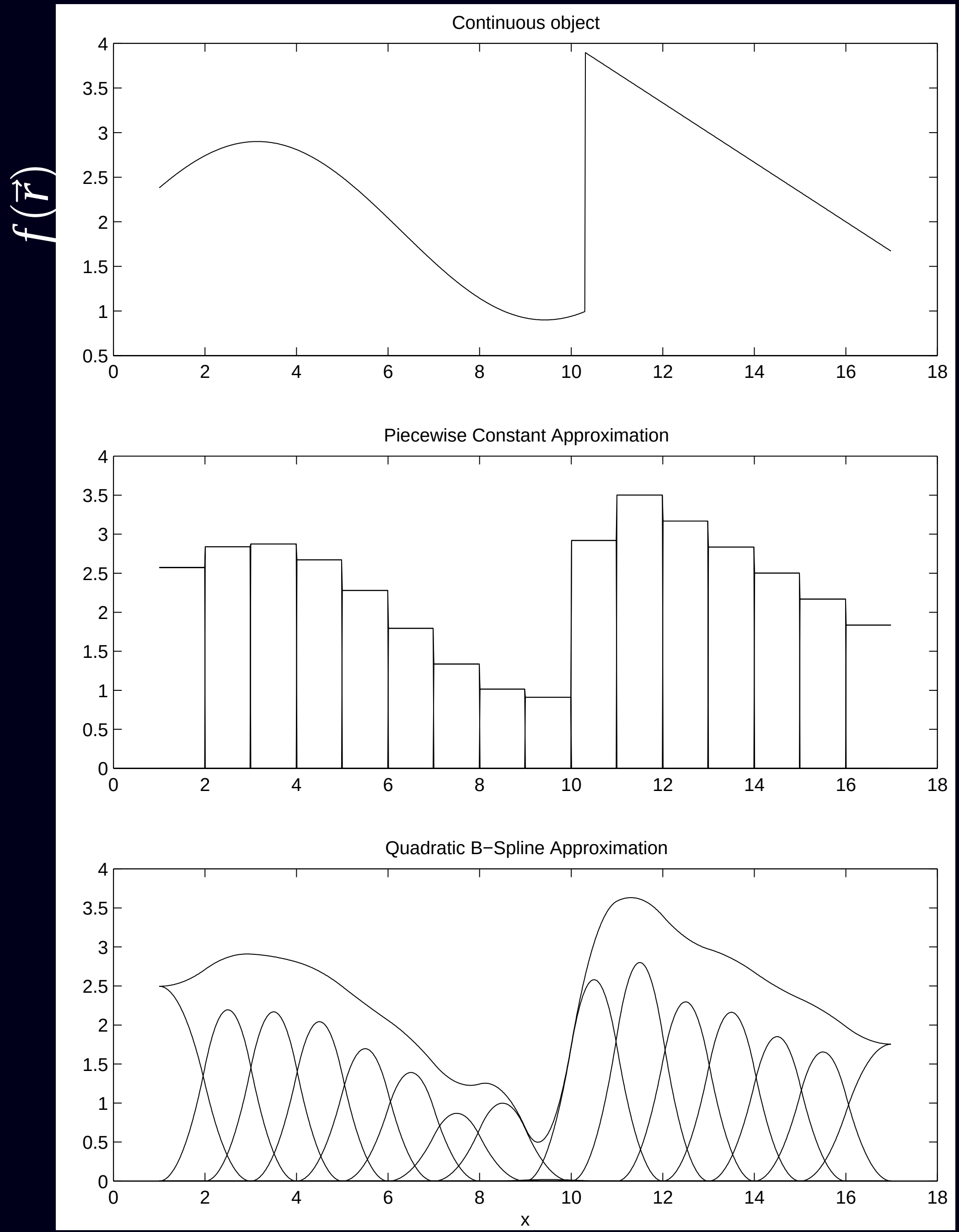
Estimation of intensity *and* shape (*e.g.*, location, radius, etc.)

Surface-based (homogeneous) models
- Circles / spheres
- Ellipses / ellipsoids
- Superquadrics
- Polygons
- Bi-quadratic triangular Bezier patches, ...

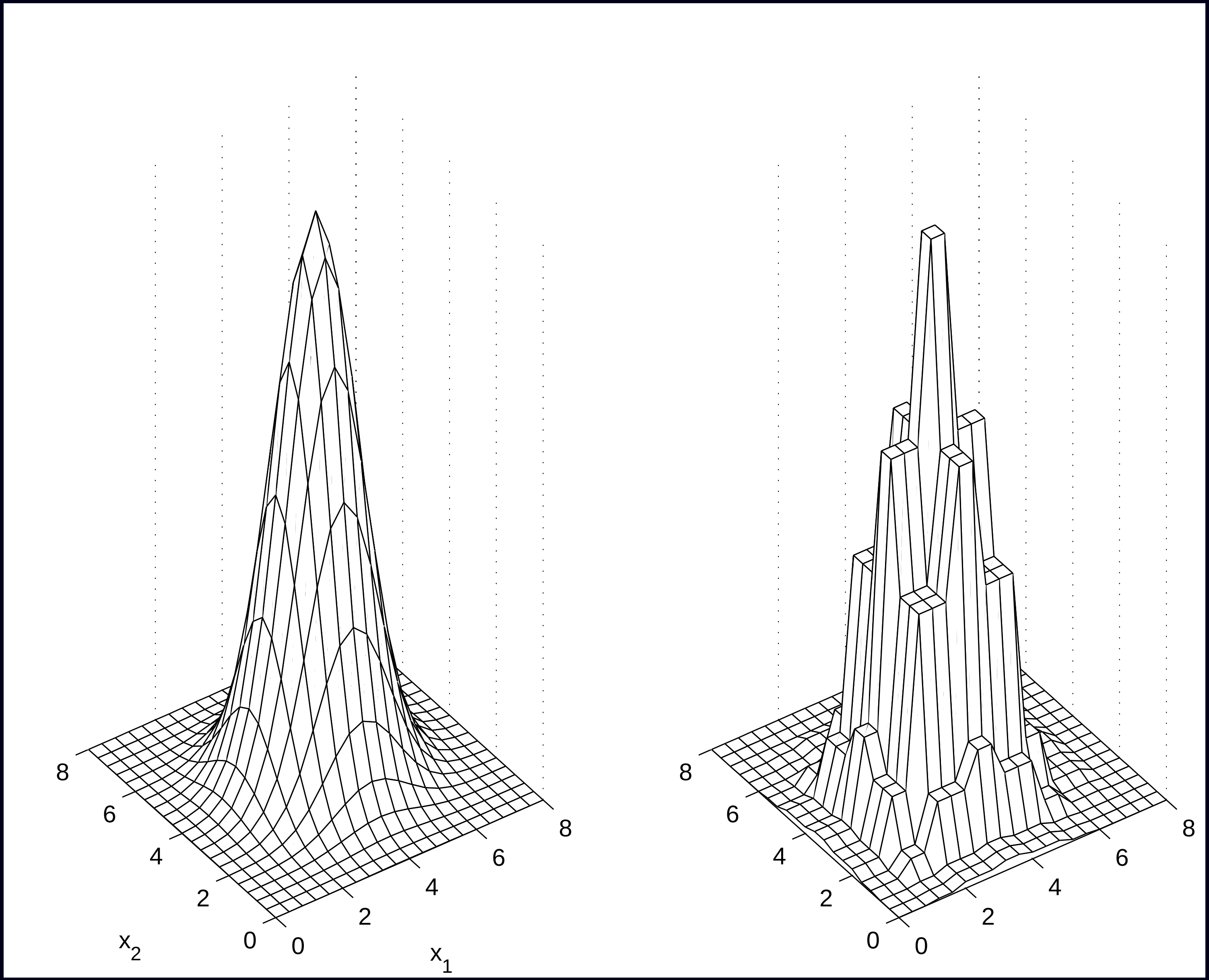
Other models
- Generalized series $f(\vec{r}) = \sum_j x_j b_j(\vec{r}, \boldsymbol{\theta})$
- Deformable templates $f(\vec{r}) = b(T_{\boldsymbol{\theta}}(\vec{r}))$
- ...

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



Pixel Basis Functions - 2D



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

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

Discrete Emission Reconstruction Problem

Having chosen a basis an parameterized the emission density...

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

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

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

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

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

Choice 2. System Model

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

- 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

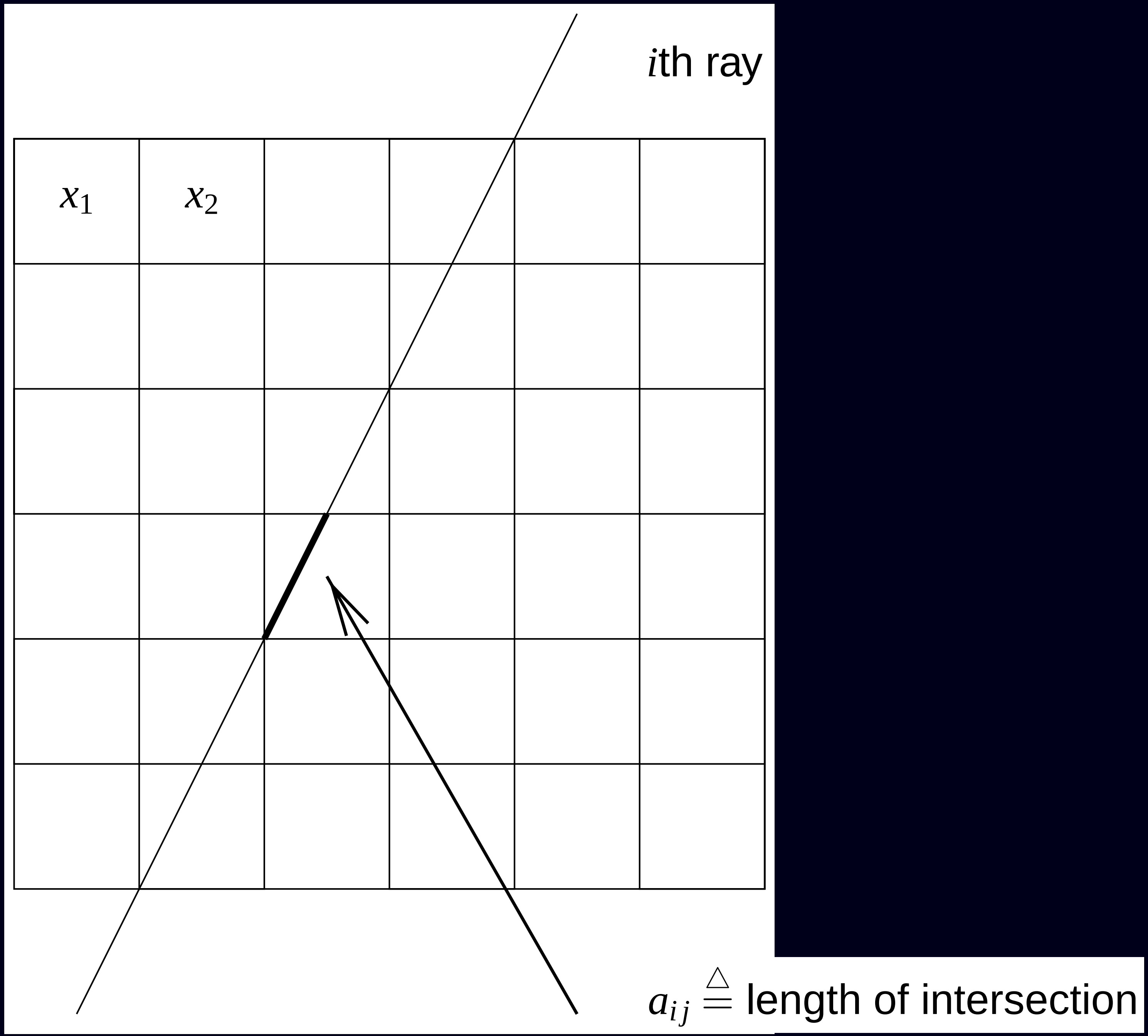
- Improving system model can improve
  - Quantitative accuracy
  - Spatial resolution
  - Contrast, SNR, detectability
- Computation time (and storage vs compute-on-fly)
- Model uncertainties
  (*e.g.*, calculated scatter probabilities based on noisy attenuation map)
- Artifacts due to over-simplifications

Measured System Model?

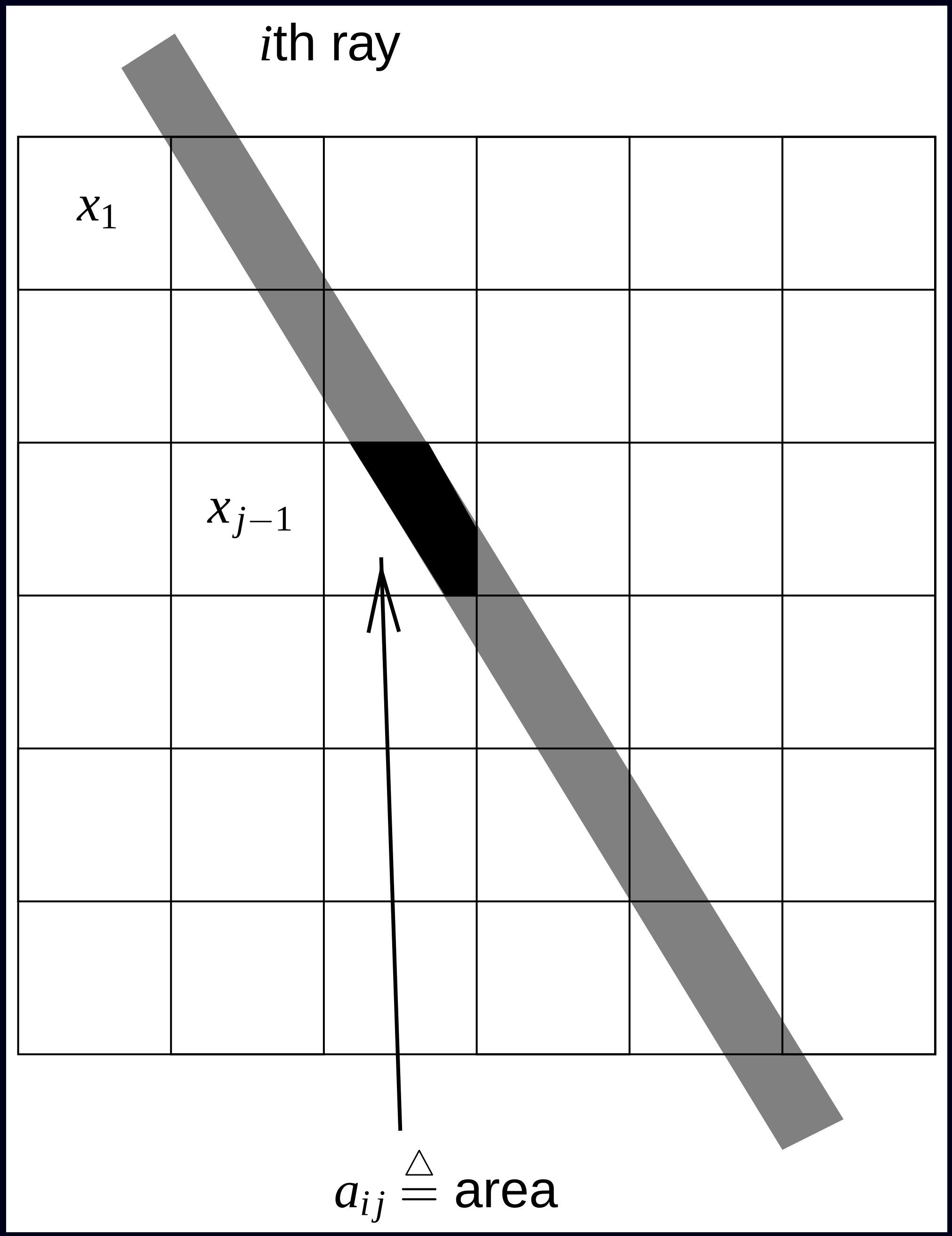
Determine a_{ij}'s by scanning a voxel-sized cube source over the imaging volume and recording counts in all detector units (separately for each voxel).

- Avoids mathematical model approximations
- Scatter / attenuation added later, approximately
- Small probabilities $\Rightarrow$ long scan times
- Storage
- Repeat for every voxel size of interest
- Repeat if detectors change

“Line Length” System Model

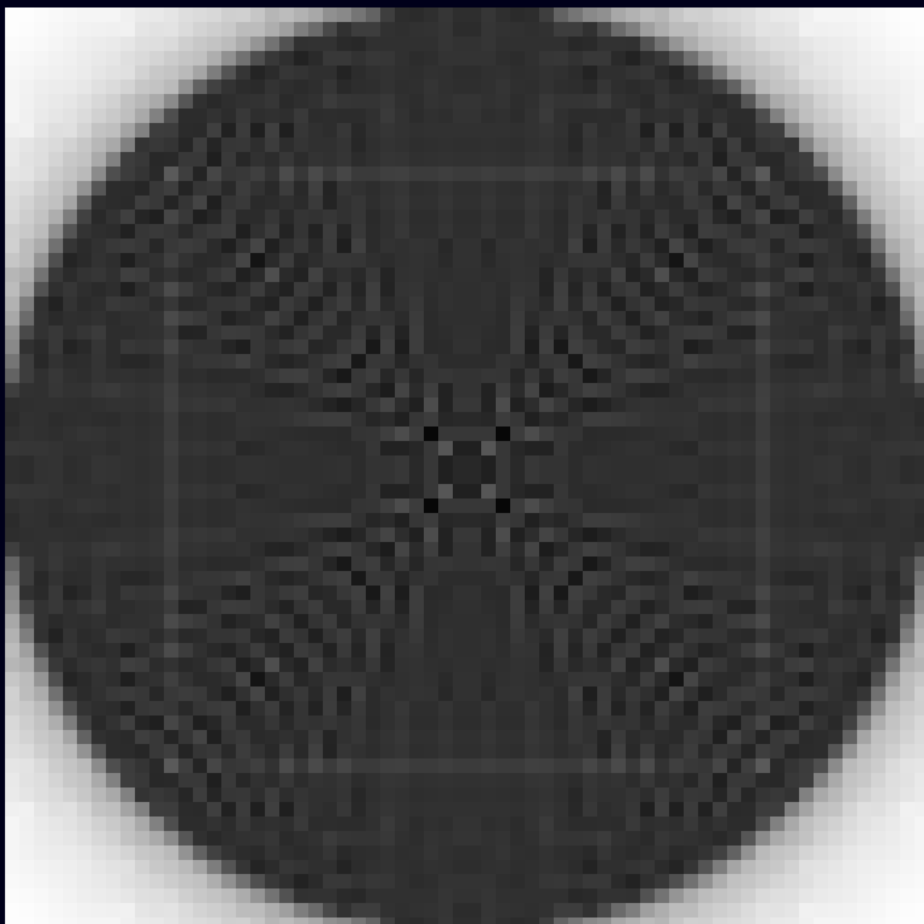


"Strip Area" System Model

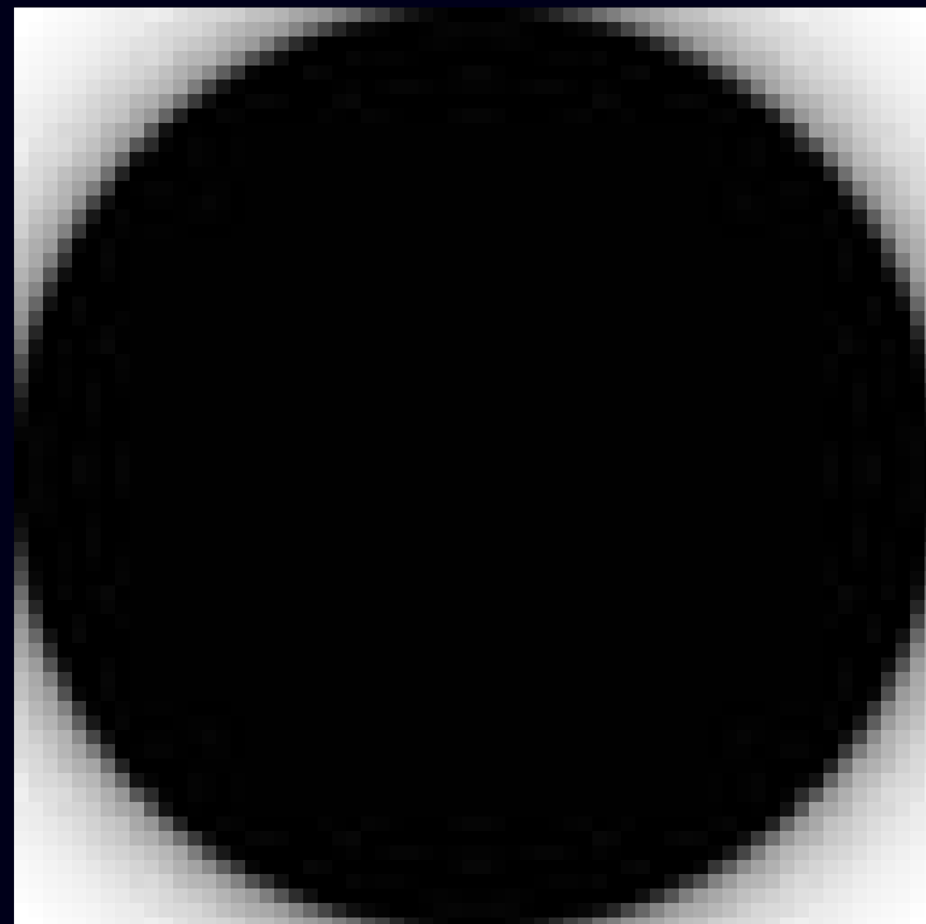


Sensitivity Patterns

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

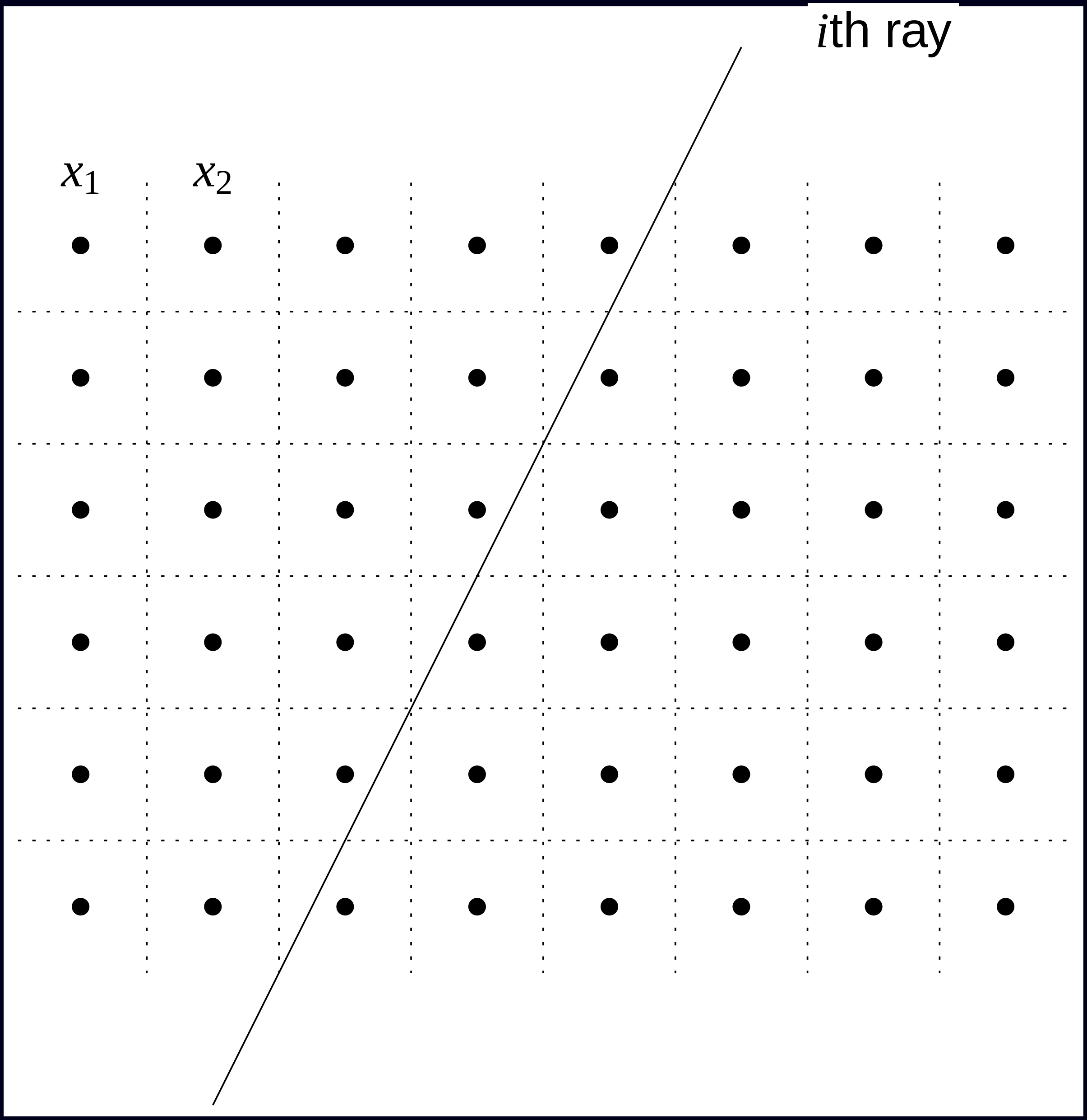


Line Length



Strip Area

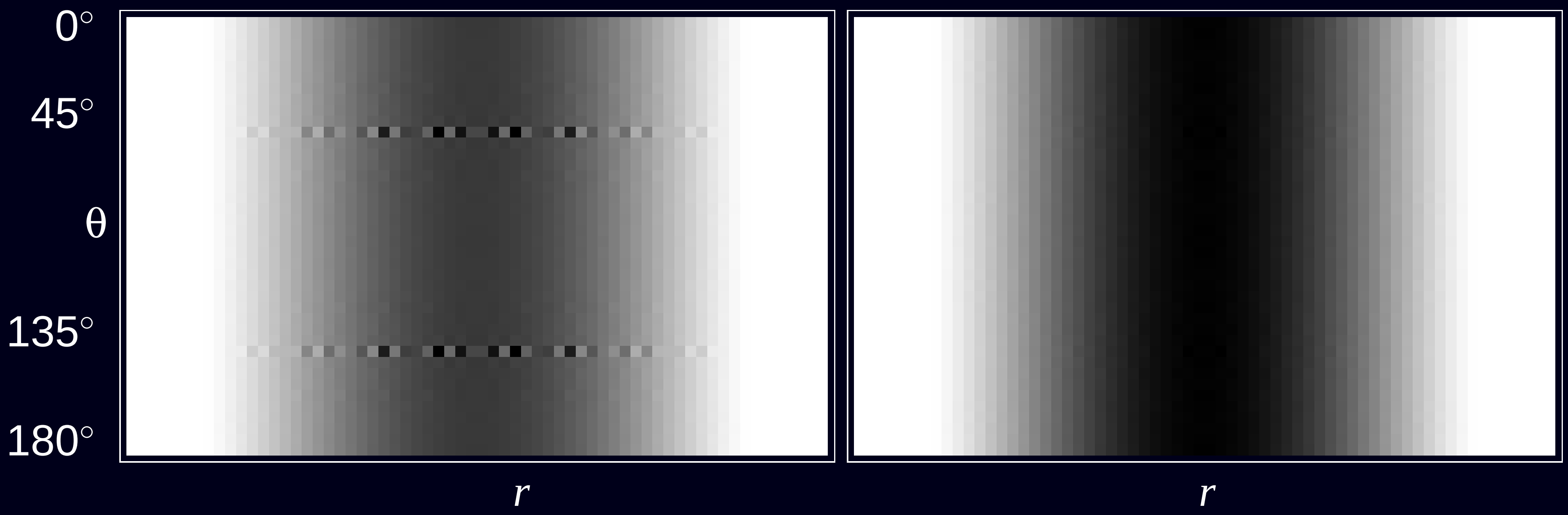
Point-Lattice Projector/Backprojector



a_{ij}'s determined by linear interpolation

Point-Lattice Artifacts

Projections (sinograms) of uniform disk object:



Point Lattice

Strip Area

Forward- / Back-projector "Pairs"

Forward projection (image domain to projection domain):

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

Backprojection (projection domain to image domain):

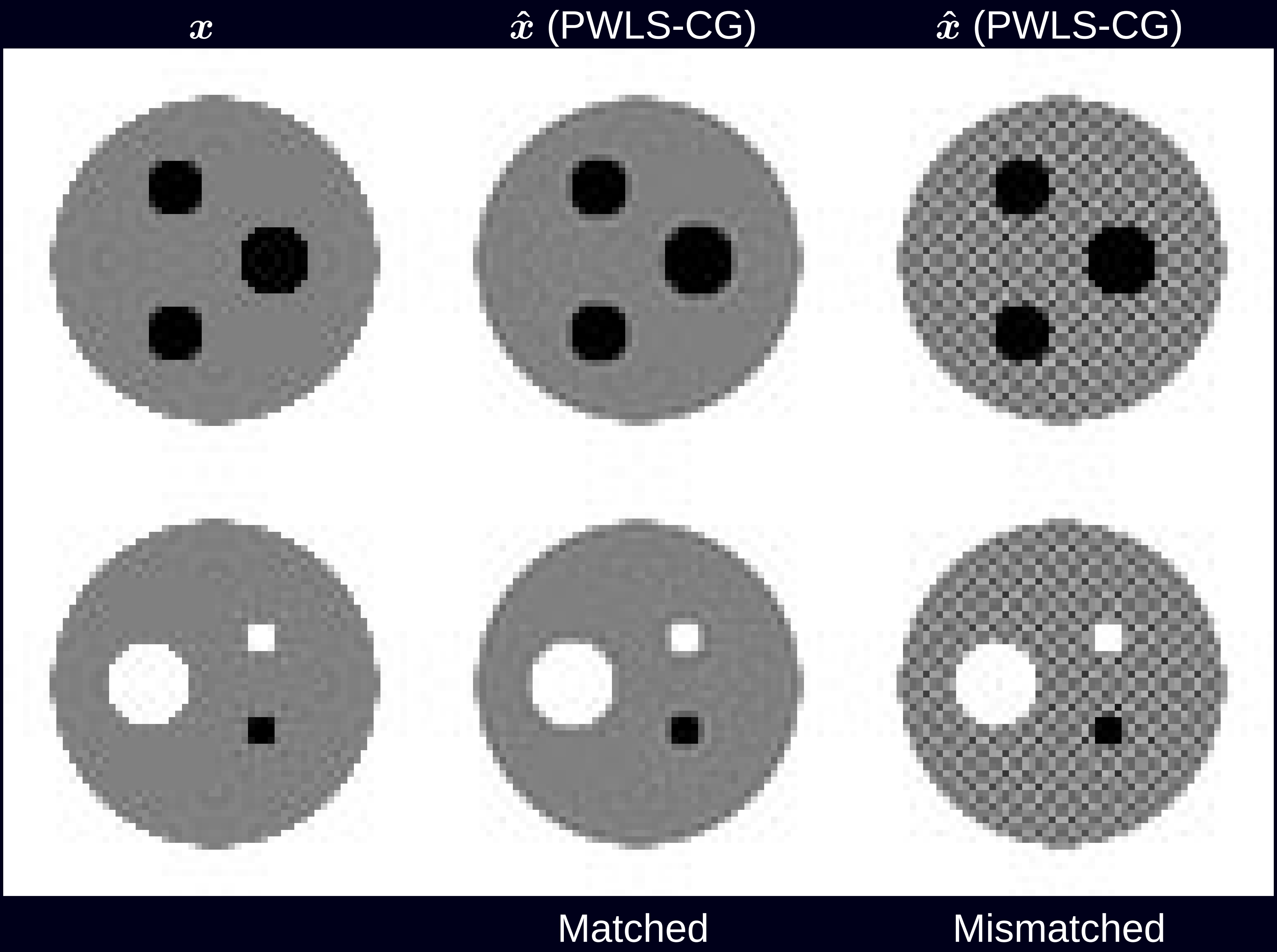
$$\boldsymbol{A}'\boldsymbol{y} = \left\{ \sum_{i=1}^{n_d} a_{ij} y_i \right\}_{j=1}^{n_p}$$

Often $\boldsymbol{A}'\boldsymbol{y}$ is implemented as $\boldsymbol{B}\boldsymbol{y}$ for some "backprojector" $\boldsymbol{B} \neq \boldsymbol{A}'$

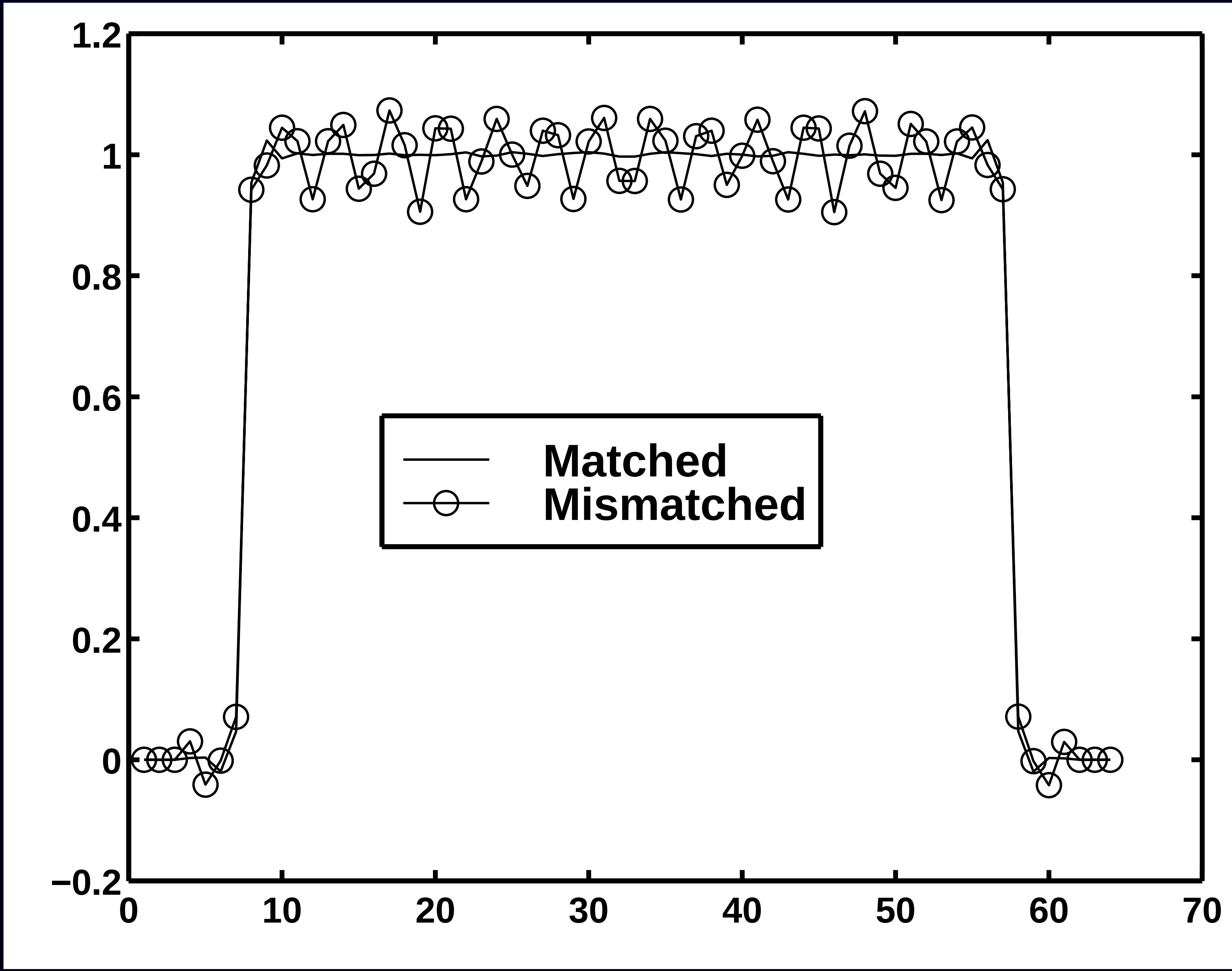
Least-squares solutions (for example):

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

Mismatched Backprojector $B \neq A'$



Horizontal Profiles



System Model Tricks

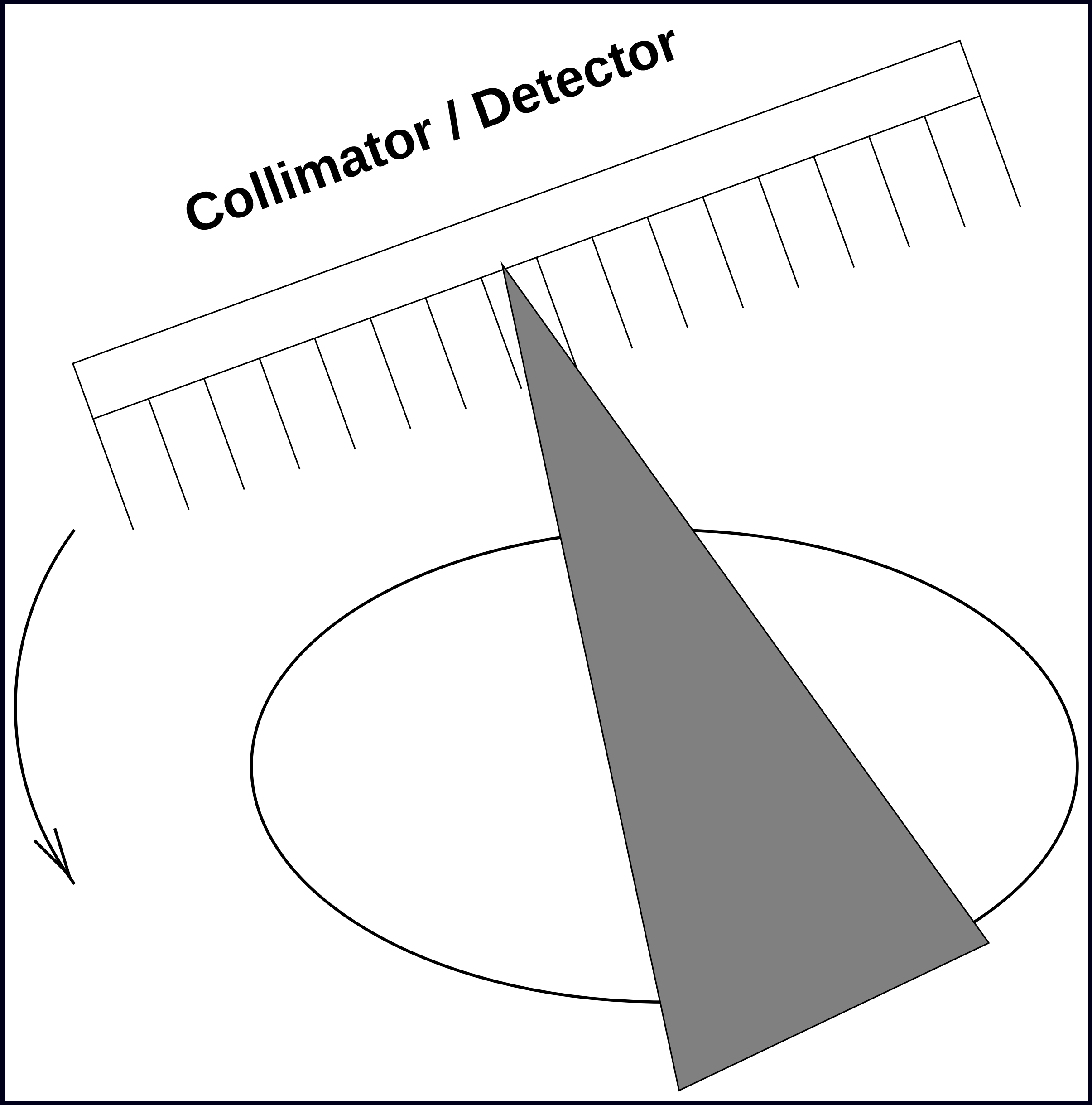
- Factorize (*e.g.*, PET Gaussian detector response)

  $$\boldsymbol{A} \approx \boldsymbol{S}\boldsymbol{G}$$

  (geometric projection followed by Gaussian smoothing)
- Symmetry
- Rotate and Sum
- Gaussian diffusion
  for SPECT Gaussian detector response
- Correlated Monte Carlo (Beekman *et al.*)

In all cases, consistency of backprojector with $\boldsymbol{A}'$ requires care.

SPECT System Model



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

Choice 3. Statistical Models

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

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

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 $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2$
- Non-white Gaussian noise. Weighted least squares: minimize

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

- 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 PET

Precorrected random coincidences: $y_i = y_i^{\text{prompt}} - y_i^{\text{delay}}$

$$\begin{aligned} y_i^{\text{prompt}} &\sim \text{Poisson}\{[\boldsymbol{A}\boldsymbol{x}]_i + r_i\} \\ y_i^{\text{delay}} &\sim \text{Poisson}\{r_i\} \\ E[y_i] &= [\boldsymbol{A}\boldsymbol{x}]_i \\ \text{Var}\{y_i\} &= [\boldsymbol{A}\boldsymbol{x}]_i + 2r_i \end{aligned}$$

Mean $\neq$ Variance $\Rightarrow$ not Poisson!

Statistical model choices

- Ordinary Poisson model: ignore randoms

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

Causes bias due to truncated negatives

- Data-weighted least-squares (Gaussian model):

$$y_i \sim N\left([\boldsymbol{A}\boldsymbol{x}]_i, \hat{\sigma}_i^2\right), \qquad \hat{\sigma}_i^2 = \max\left(y_i + 2\hat{r}_i, \sigma_{\min}^2\right)$$

Causes bias due to data-weighting

- Shifted Poisson model (matches 2 moments):

$$[y_i + 2\hat{r}_i]_+ \sim \text{Poisson}\{[\boldsymbol{A}\boldsymbol{x}]_i + 2\hat{r}_i\}$$

Insensitive to inaccuracies in $\hat{r}_i$.

Shifted-Poisson Model for X-ray CT

Model with both photon variability and readout noise:

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

Shifted Poisson approximation

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

or just use WLS...

Complications:

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

Choice 4. Cost Functions

Components:

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

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

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

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 $p(\boldsymbol{x})$ for image vectors $\boldsymbol{x}$, by Bayes' rule:

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

so

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

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

Maximum a posteriori (MAP) estimator:

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

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

Choice 4.1: Data-Mismatch Term

Options for PET:

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

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

- Ordinary (unweighted) least squares: $\sum_{i=1}^{n_d} \frac{1}{2}(y_i - \hat{r}_i - [\boldsymbol{A}\boldsymbol{x}]_i)^2$
- Data-weighted least squares: $\sum_{i=1}^{n_d} \frac{1}{2}(y_i - \hat{r}_i - [\boldsymbol{A}\boldsymbol{x}]_i)^2 / \hat{\sigma}_i^2, \; \hat{\sigma}_i^2 = \max\left(y_i + \hat{r}_i, \sigma_{\min}^2\right),$ (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!)

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

- 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 [112]

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, 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
- Predictability
- Resolution can be made object independent
- Controlled resolution (*e.g.*, spatially uniform, edge preserving)
- Start with decent image (*e.g.*, FBP) $\Rightarrow$ reach solution faster.

True Regularization Methods

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

- 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
  - Disadvantage: apparently subjective choice of penalty
  - Apparent difficulty in choosing penalty parameters
    (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 nonseparable
- quadratic vs nonquadratic
- convex vs nonconvex

Penalty Functions: Separable vs Nonseparable

Separable

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

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

Simple, but these do not explicitly enforce smoothness.

Penalty Functions: Separable vs Nonseparable

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

x_1	x_2	x_3
x_4	x_5	

Example

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

2	2	2
2	1	

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

3	3	1
2	2	

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

1	3	1
2	2	

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

Rougher images $\Rightarrow$ greater $R(\boldsymbol{x})$

Roughness Penalty Functions

First-order neighborhood and pairwise pixel differences:

$$R(\boldsymbol{x}) = \sum_{j=1}^{n_p} \frac{1}{2} \sum_{k \in N_j} \psi(x_j - x_k)$$

$N_j \triangleq$ *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!)

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

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

Penalty Functions: General Form

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

Example:

x_1	x_2	x_3
x_4	x_5	

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

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

Penalty Functions: Quadratic vs Nonquadratic

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

Quadratic ψ_k

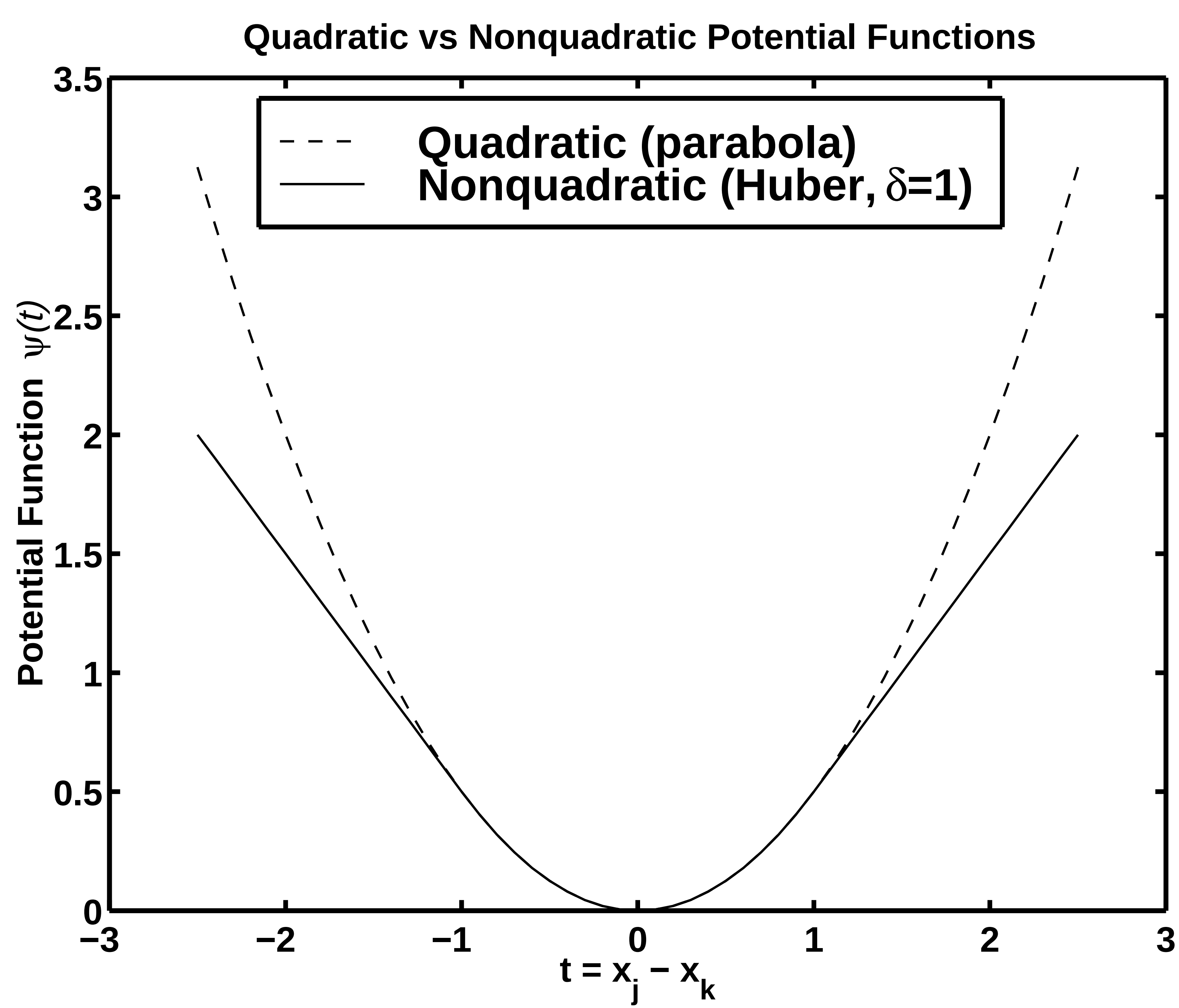
If $\psi_k(t) = t^2/2$, then $R(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}'\boldsymbol{C}'\boldsymbol{C}\boldsymbol{x}$, a quadratic form.

- Simpler optimization
- Global smoothing

Nonquadratic ψ_k

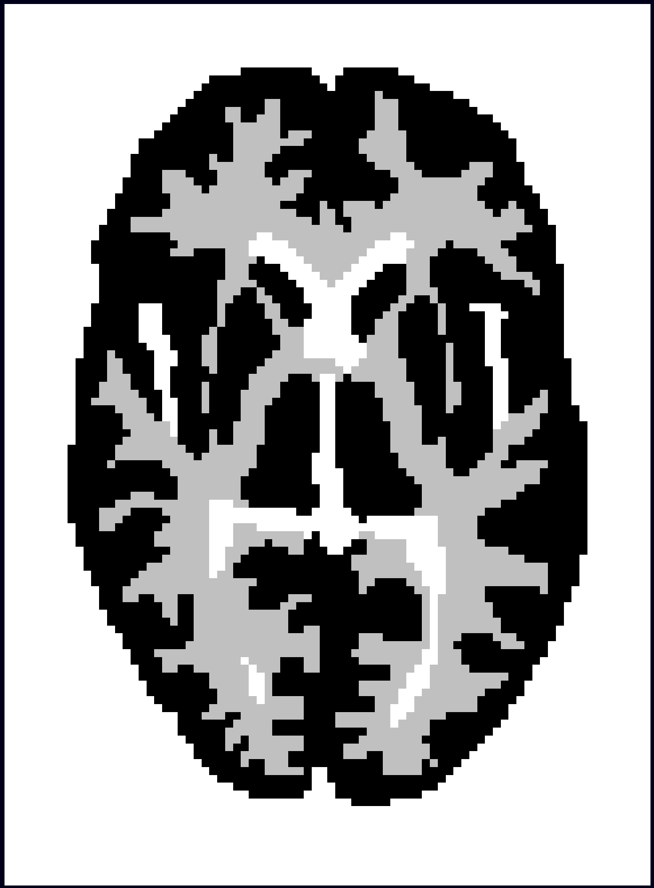
- Edge preserving
- More complicated optimization. (This is essentially solved in convex case.)
- Unusual noise properties
- Analysis/prediction of resolution and noise properties is difficult
- More adjustable parameters (*e.g.*, δ)

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

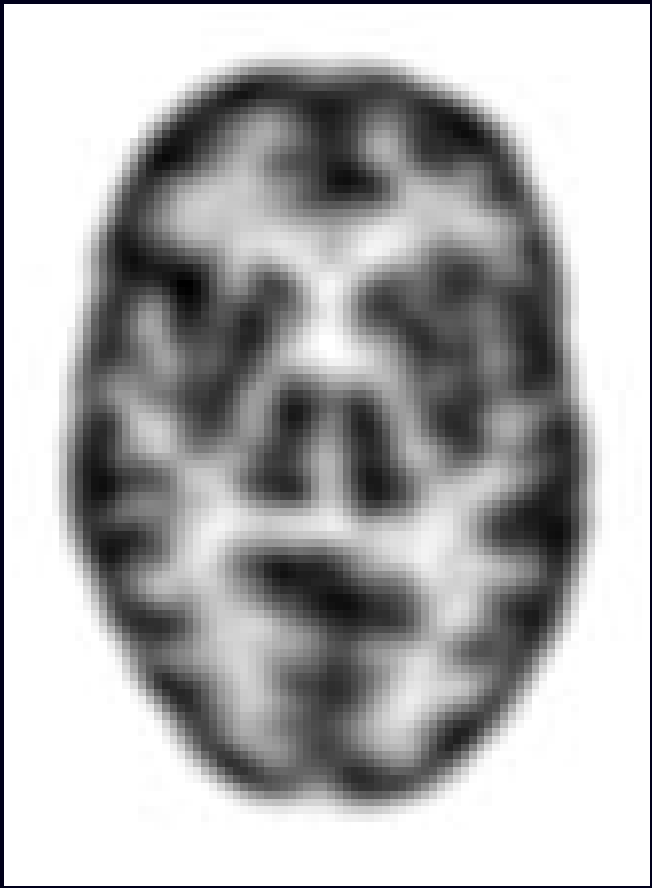


Lower cost for large differences ⇒ edge preservation

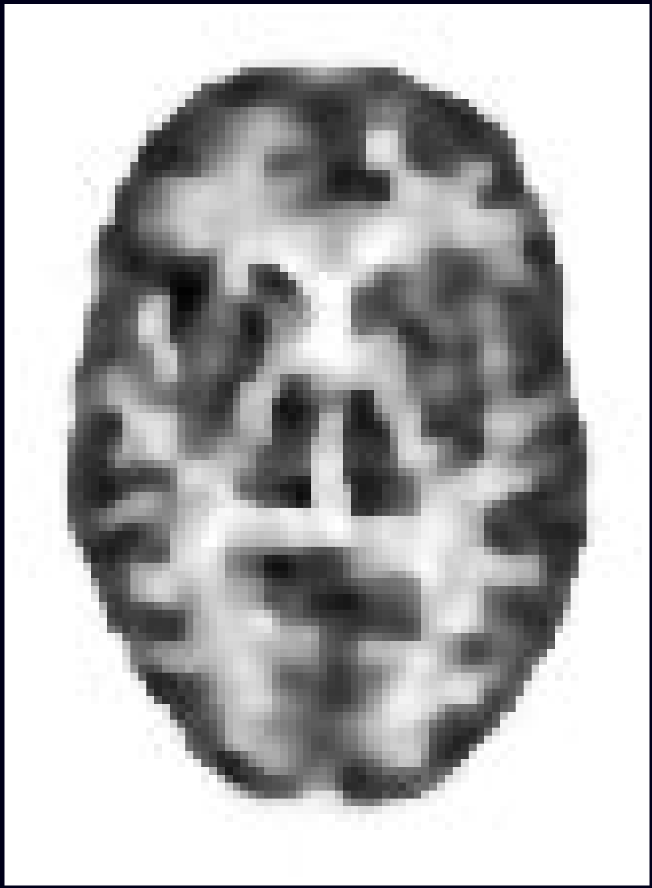
Edge-Preserving Reconstruction Example



Phantom



Quadratic Penalty



Huber Penalty

A transmission example would be preferable...

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

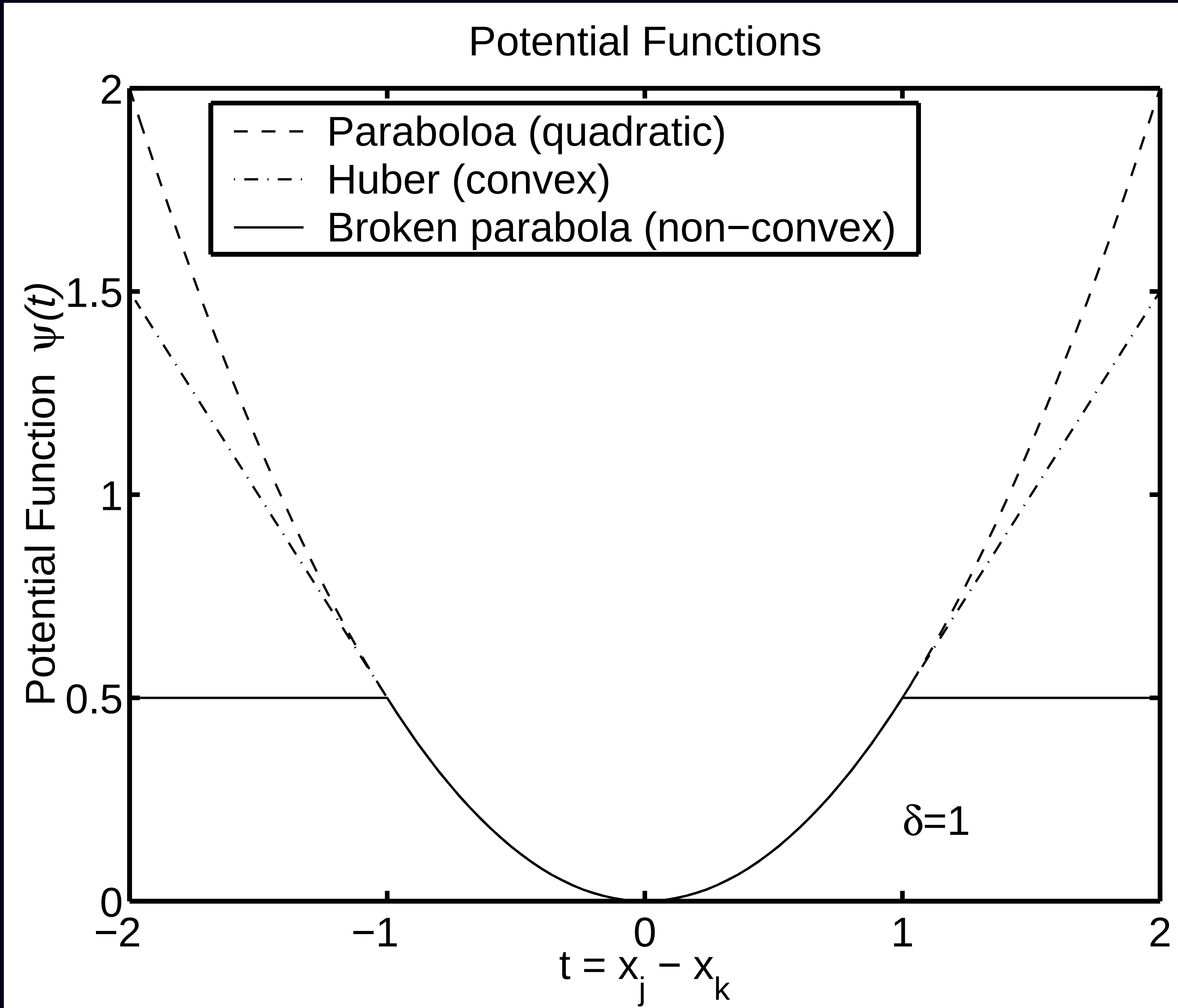
- "broken parabola"

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

- true median root prior:

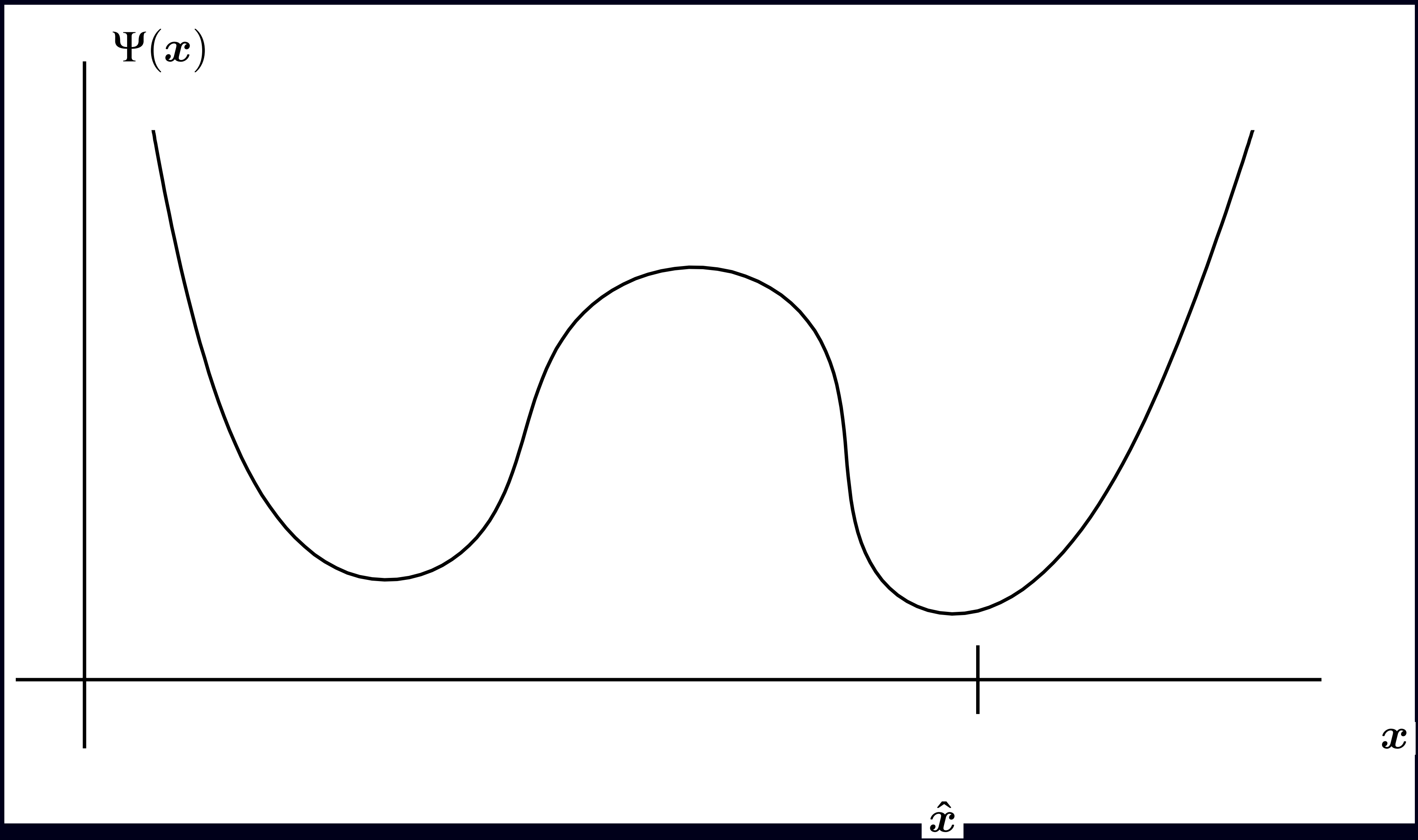
$$R(\boldsymbol{x}) = \sum_{j=1}^{n_p} \frac{(x_j - \operatorname{median}_j(\boldsymbol{x}))^2}{\operatorname{median}_j(\boldsymbol{x})} \quad \text{where} \quad \operatorname{median}_j(\boldsymbol{x}) \quad \text{is local median}$$

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



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

Local Extrema and Discontinuous Estimators



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

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:

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

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

$$R(\boldsymbol{x}|\boldsymbol{b}) = \sum_{j=1}^{n_p} \sum_{k \in N_j} (1 - b_{jk}) \frac{1}{2} (x_j - x_k)^2$$

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

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

- Can encourage local edge continuity
- Require annealing methods for minimization

Modified Penalty Functions

$$R(\boldsymbol{x}) = \sum_{j=1}^{n_p} \frac{1}{2} \sum_{k \in N_j} w_{jk} \psi(x_j - x_k)$$

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)
  - consider convex nonquadratic (*e.g.*, Huber) penalty functions
  - choose δ based on attenuation map units
  - 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

Modeling

- Noise in a_{ij}'s (system model errors)
- Noise in $\hat{r}_i$'s (estimates of scatter / randoms)
- Statistics of corrected measurements
- Statistics of measurements with deadtime losses

Cost functions

- Performance prediction for nonquadratic penalties
- Effect of nonquadratic penalties 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: $s_i(\boldsymbol{x})$
- 3. Measurement statistical model $Y_i \sim \boxed{?}$
- 4. Cost function: data-mismatch / regularization / constraints

Reconstruction Method = Cost Function + Algorithm

Naming convention:

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