Fast Temporal Reconstruction for EIT

Andy Adler[1], William R.B. Lionheart[2]

[1]Systems and Computer Engineering, Carleton University, Ottawa, Canada

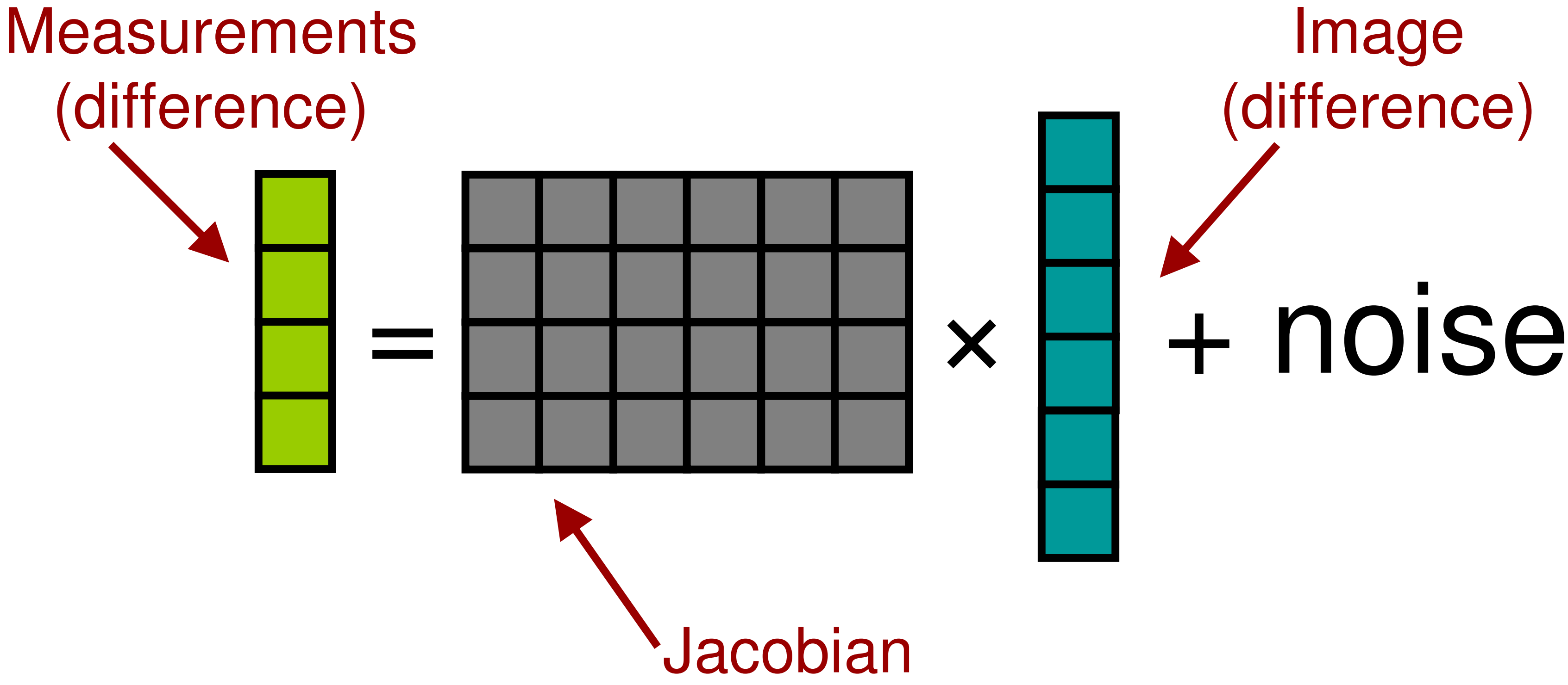
[2]School of Mathematics, University of Manchester, U.K.

Motivation

- One good thing about EIT is that the data collection is fast
- Frame rates up to 1000 fps
- Traditionally, EIT uses frame by frame image reconstruction
- It should be possible to use the *temporal* information to improve the images

Image Reconstruction

- Forward Model (linearized)



System is underdetermined

Image Reconstruction

- Inverse Model (linearized)

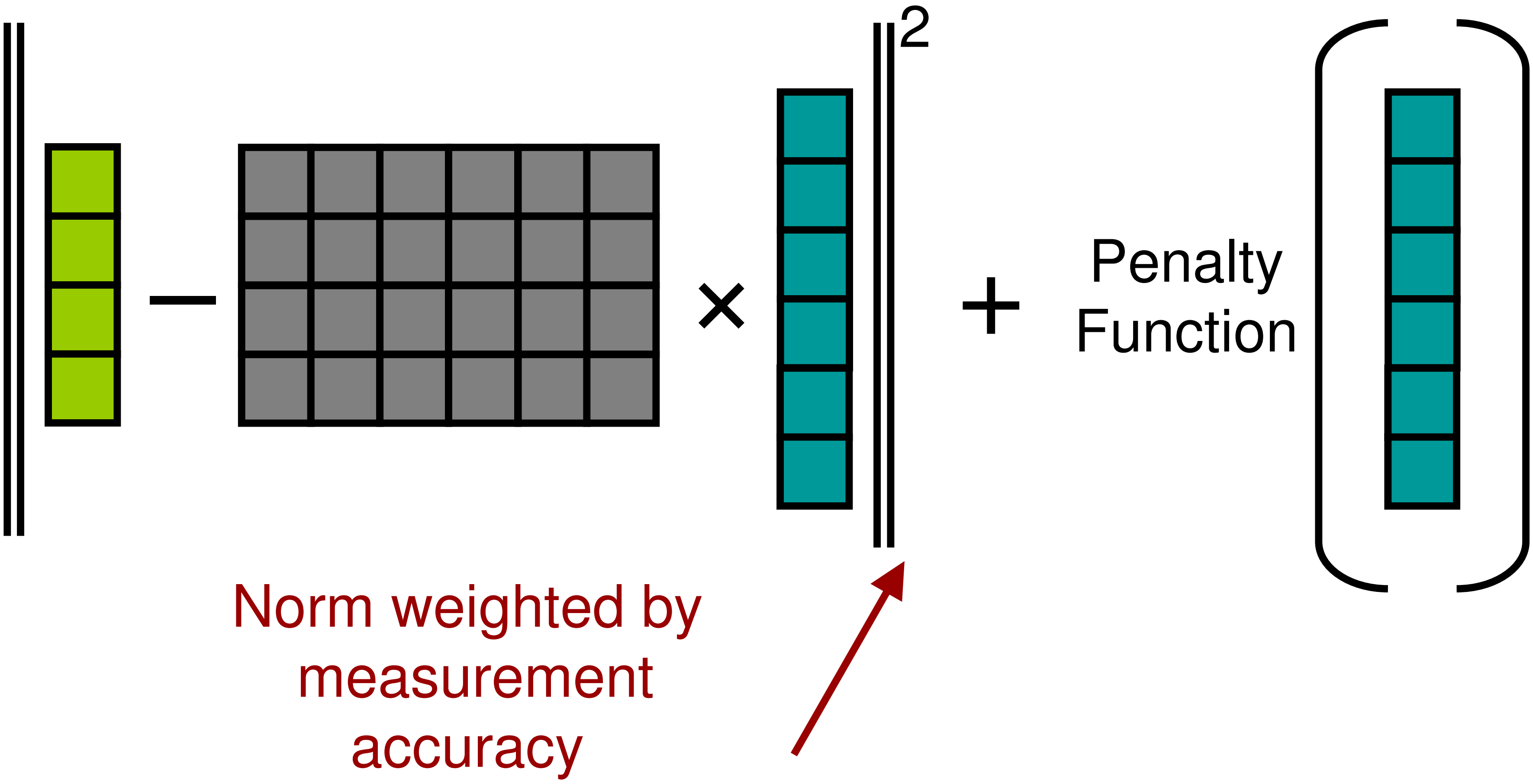


Image Reconstruction

- Penalty Functions

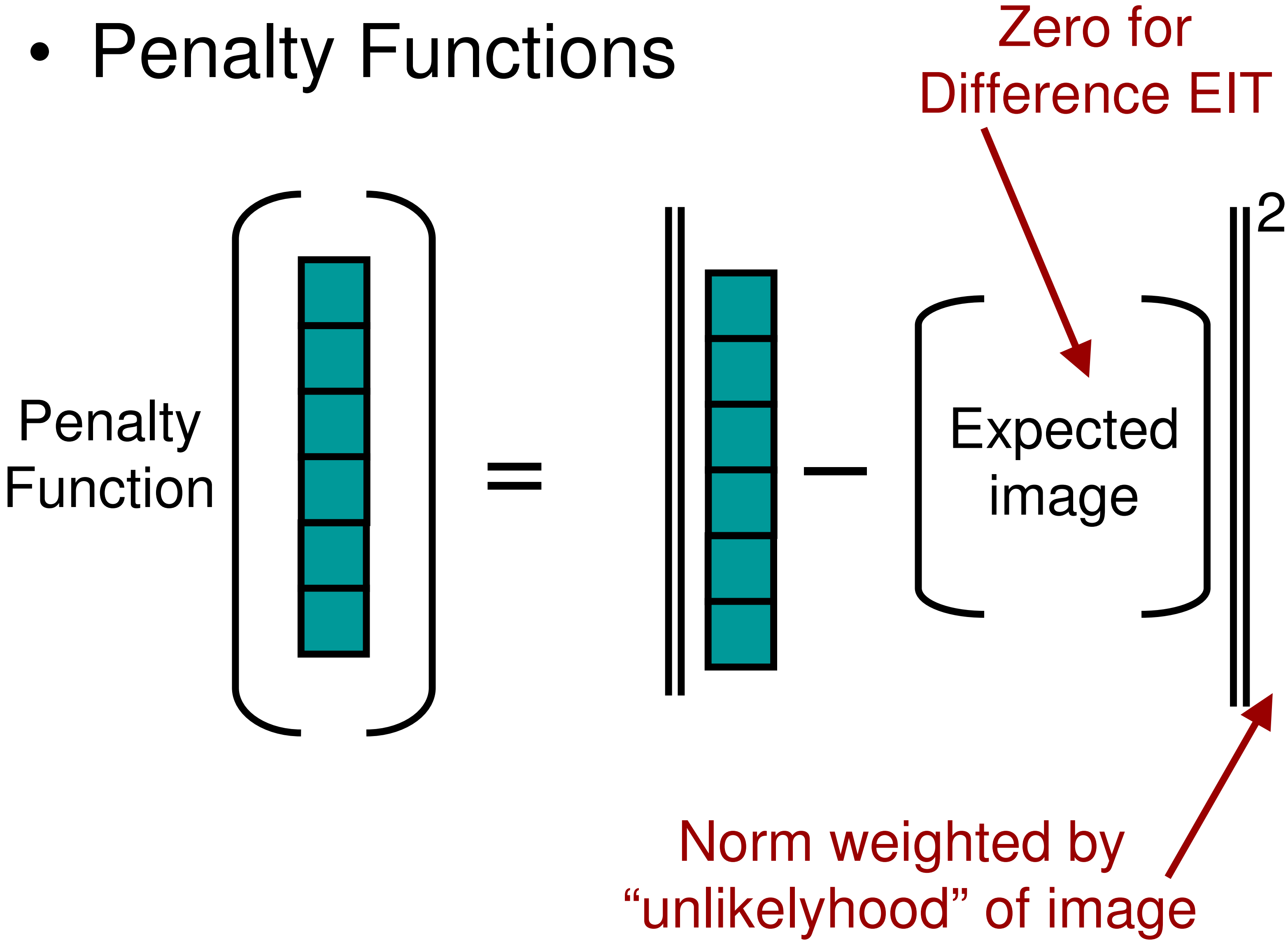


Image Reconstruction

- Penalty functions: Image Amplitude

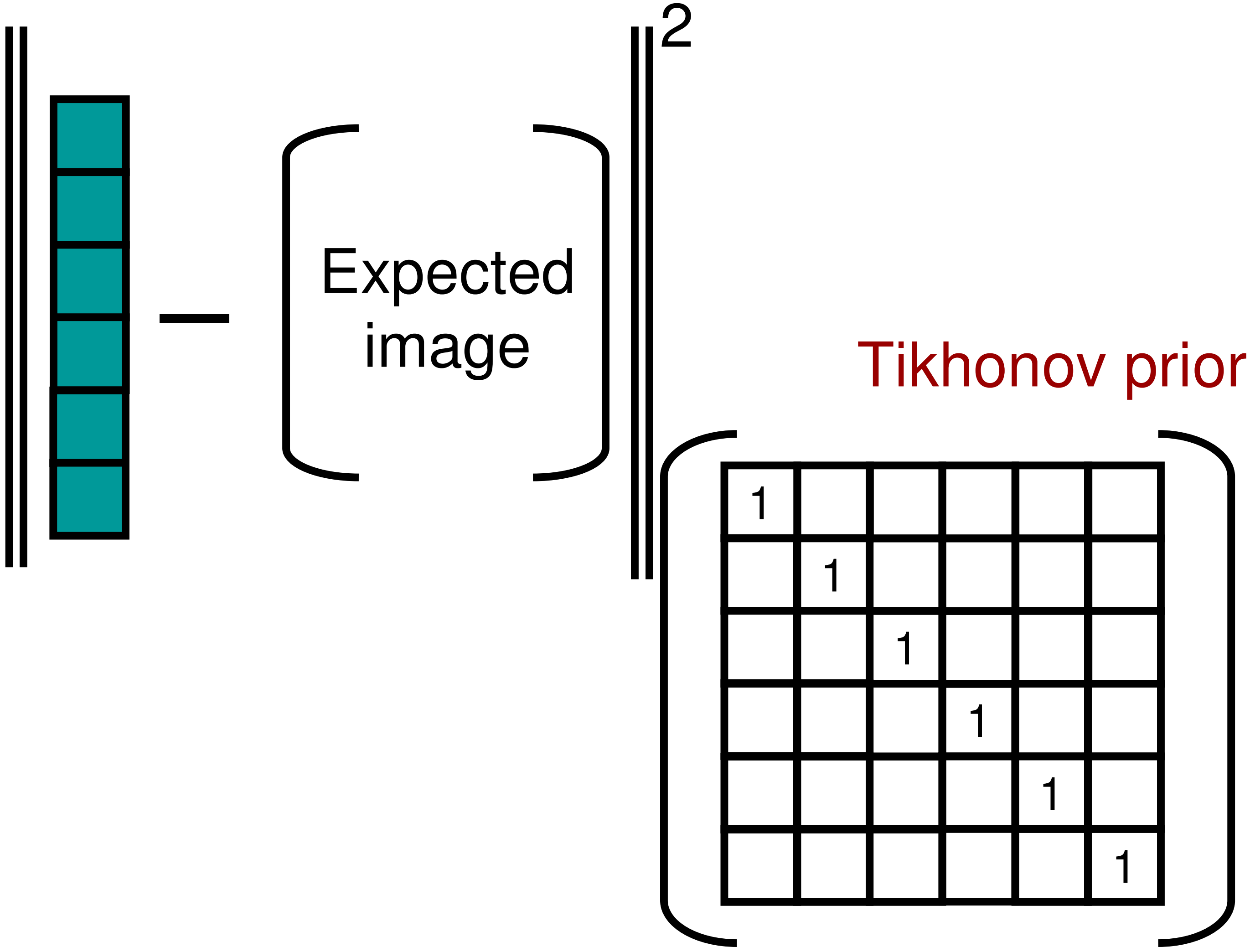
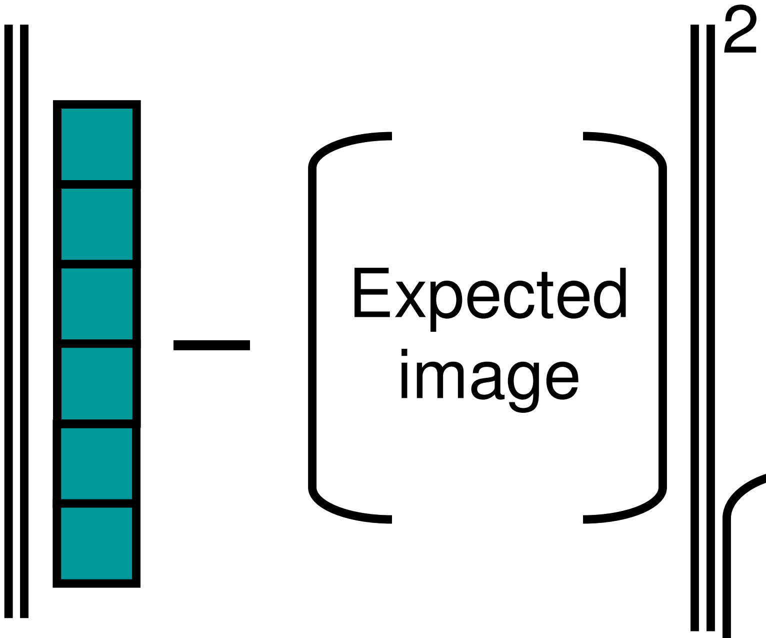


Image Reconstruction

- Penalty functions: Image Smoothness



Laplacian prior

$$
\begin{pmatrix}
1 & -\tfrac{1}{2} & & & & \\
-\tfrac{1}{2} & 1 & -\tfrac{1}{2} & & & \\
 & -\tfrac{1}{2} & 1 & -\tfrac{1}{2} & & \\
 & & -\tfrac{1}{2} & 1 & -\tfrac{1}{2} & \\
 & & & -\tfrac{1}{2} & 1 & -\tfrac{1}{2} \\
 & & & & -\tfrac{1}{2} & 1
\end{pmatrix}
$$

Compare Penalty Functions

Images

(A)

1
1
1

(B)

1
1
1

Priors

1					
	1				
		1			
			1		
				1	
					1

1	-½				
-½	1	-½			
	-½	1	-½		
		-½	1	-½	
			-½	1	-½
				-½	1

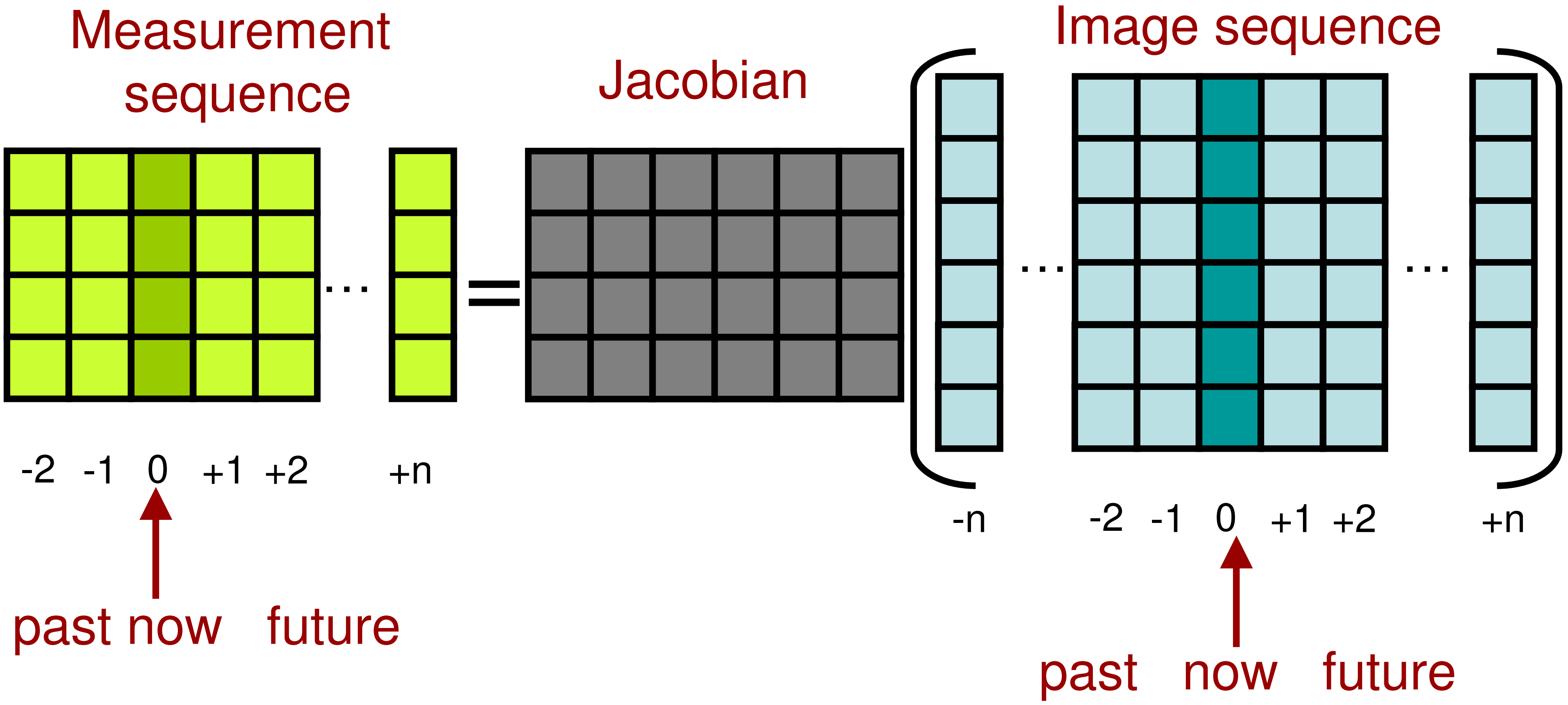
Penalties

A) 3
B) 3

More reasonable
Image A is more likely

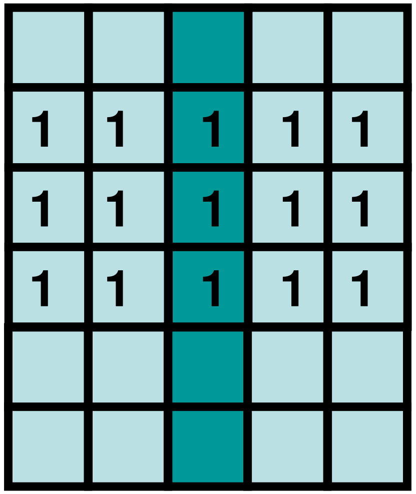
A) 1
B) 3

What about time?

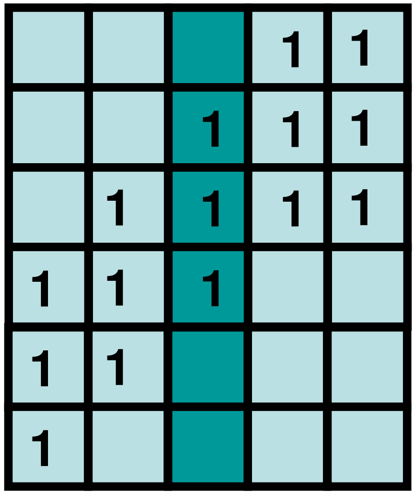


Temporal Reconstruction

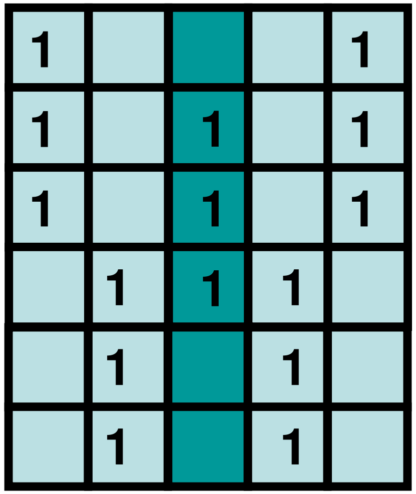
Temporal Penalty Functions



likely



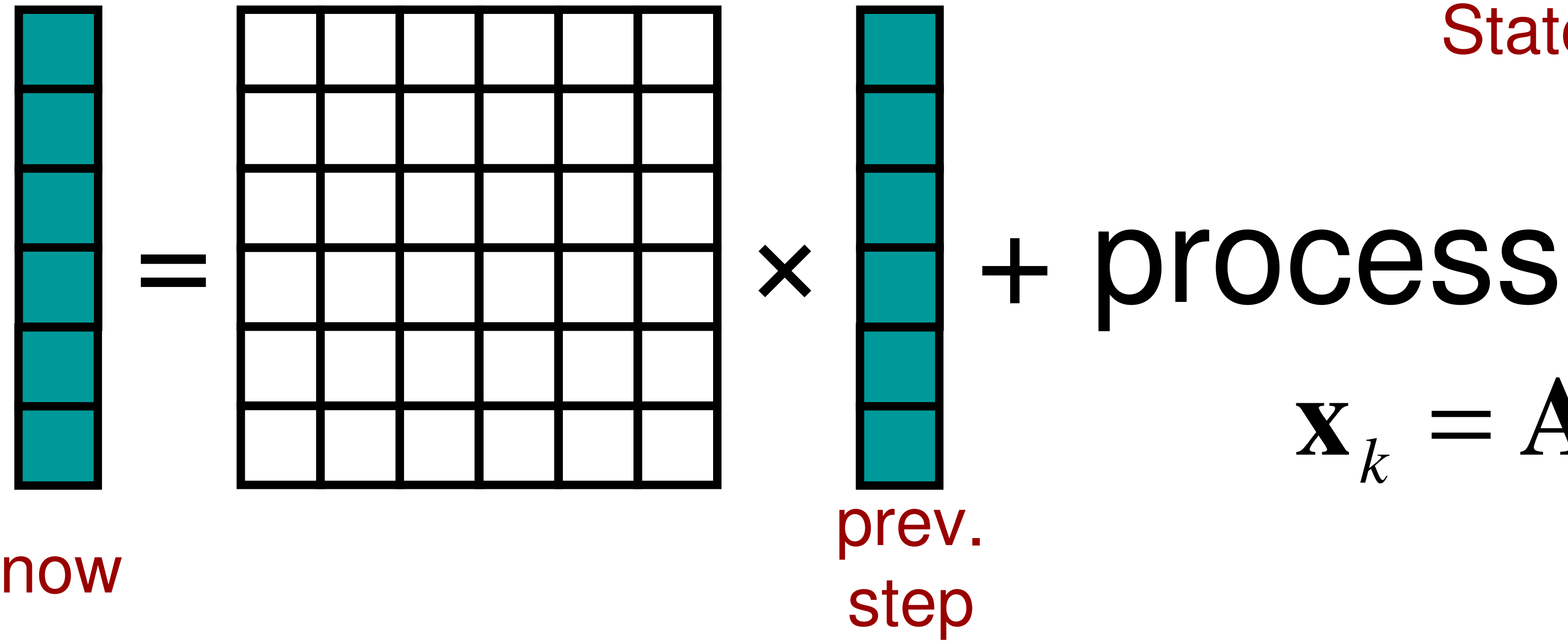
quite likely



unlikely

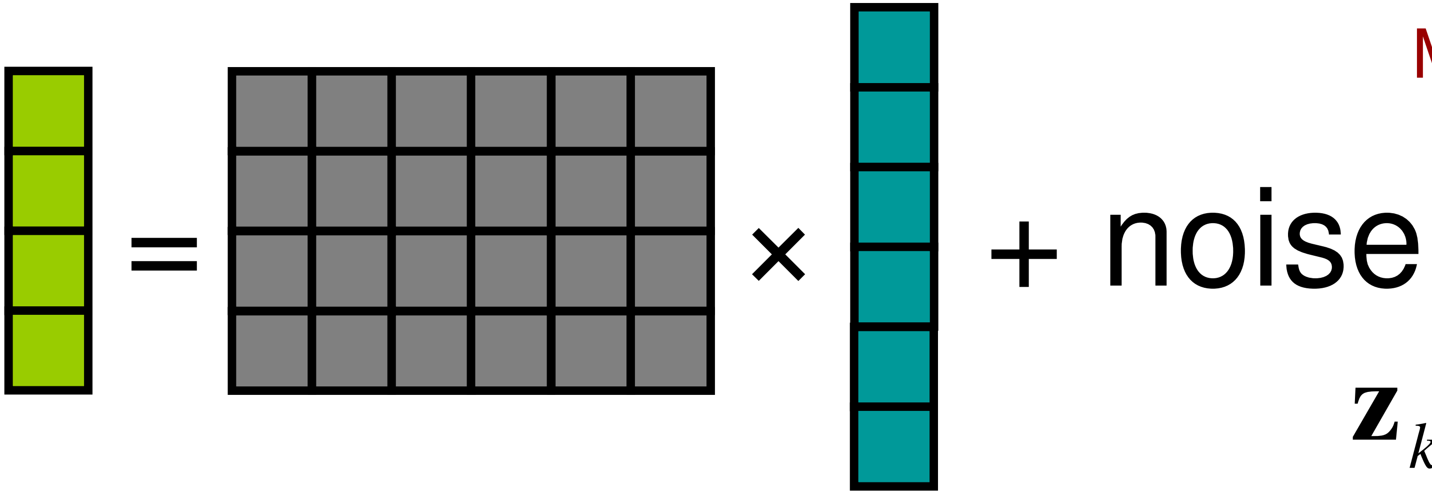
Standard EIT approaches to not take this into account

Kalman Filtering



State transition
Matrix

$$\mathbf{x}_k = \mathbf{A}\mathbf{x}_{k-1} + \mathbf{w}$$



Observation
Matrix (Jacobian)

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{n}$$

Kalman Filtering

Two stage process

- Prediction:

Estimate of now based on old data only

$$\mathbf{x}_k^- = \mathbf{A}\mathbf{x}_{k-1}$$

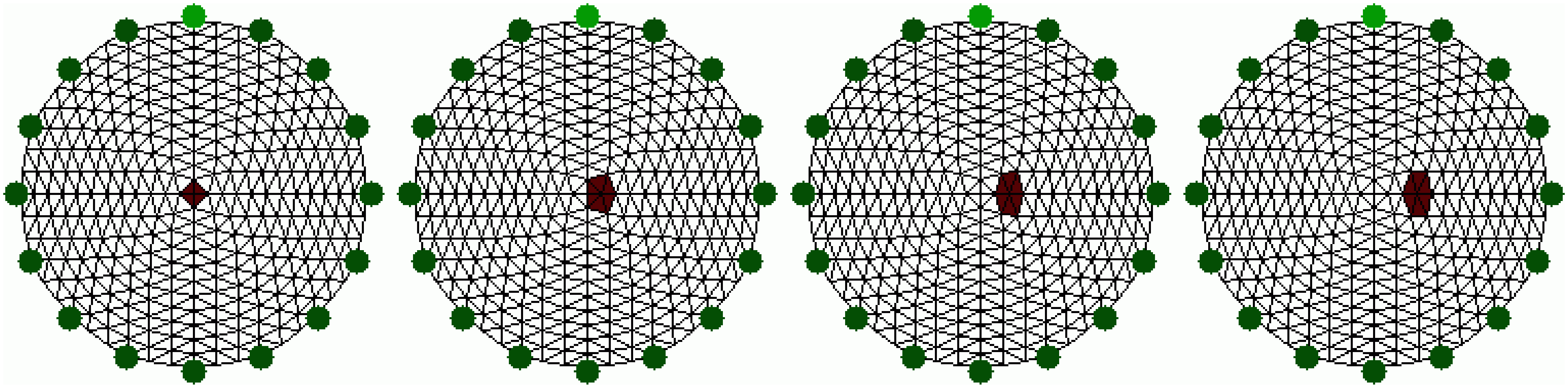
- Update:

$$\hat{\mathbf{x}}_k = \mathbf{x}_k^- + \mathbf{K}_k\left(\mathbf{z}_k - \mathbf{H}\mathbf{x}_k^-\right)$$

- K is *Kalman gain*:
  - Need to update at each step
  - Depends on $\mathbf{P}_k = \operatorname{cov}\left(\hat{\mathbf{x}}_k - \mathbf{x}_k\right)$

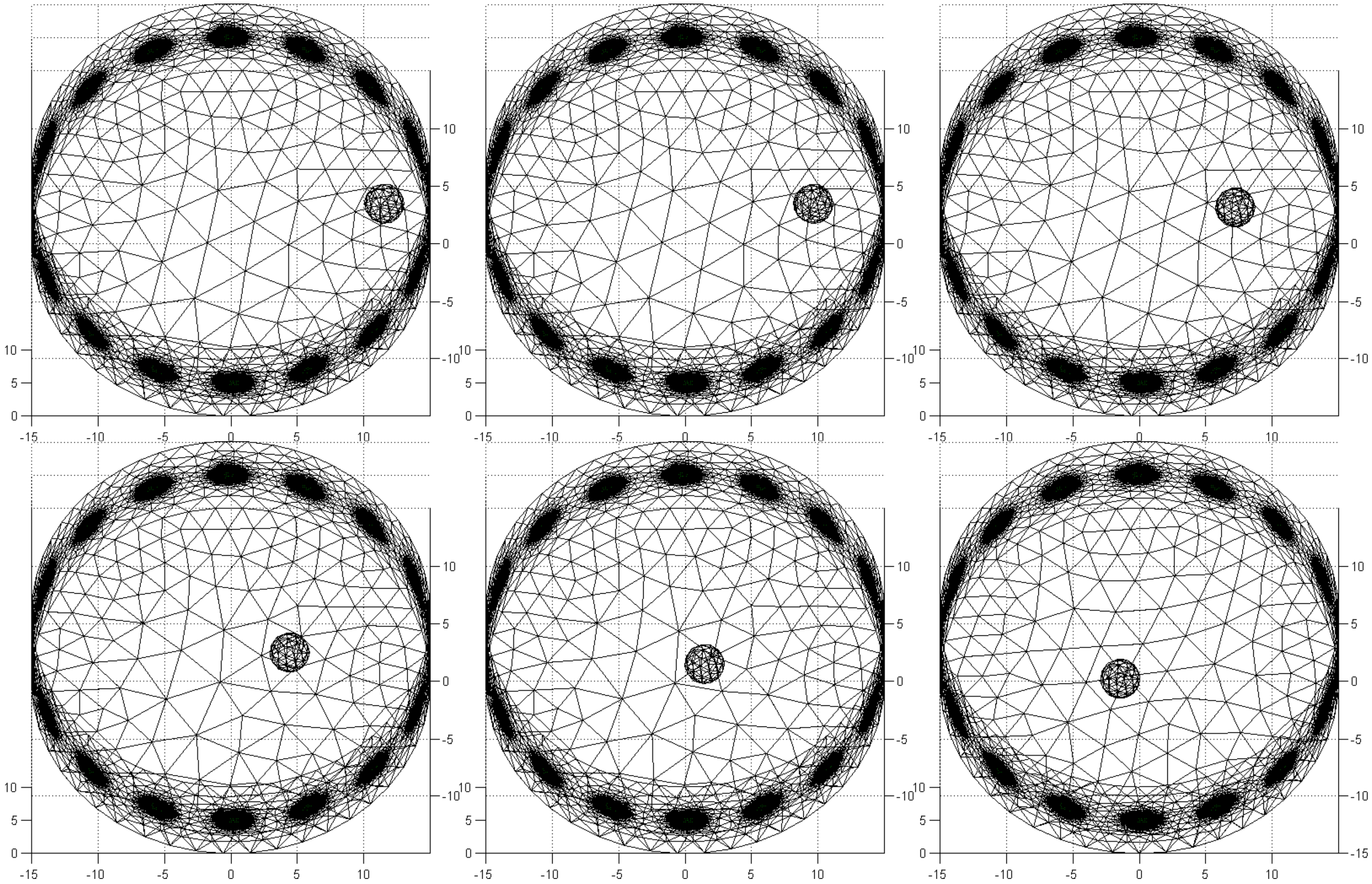
Aside: Simulating movement

- Simulating movement is really tricky
- Simple solution is to choose different elements in a FEM



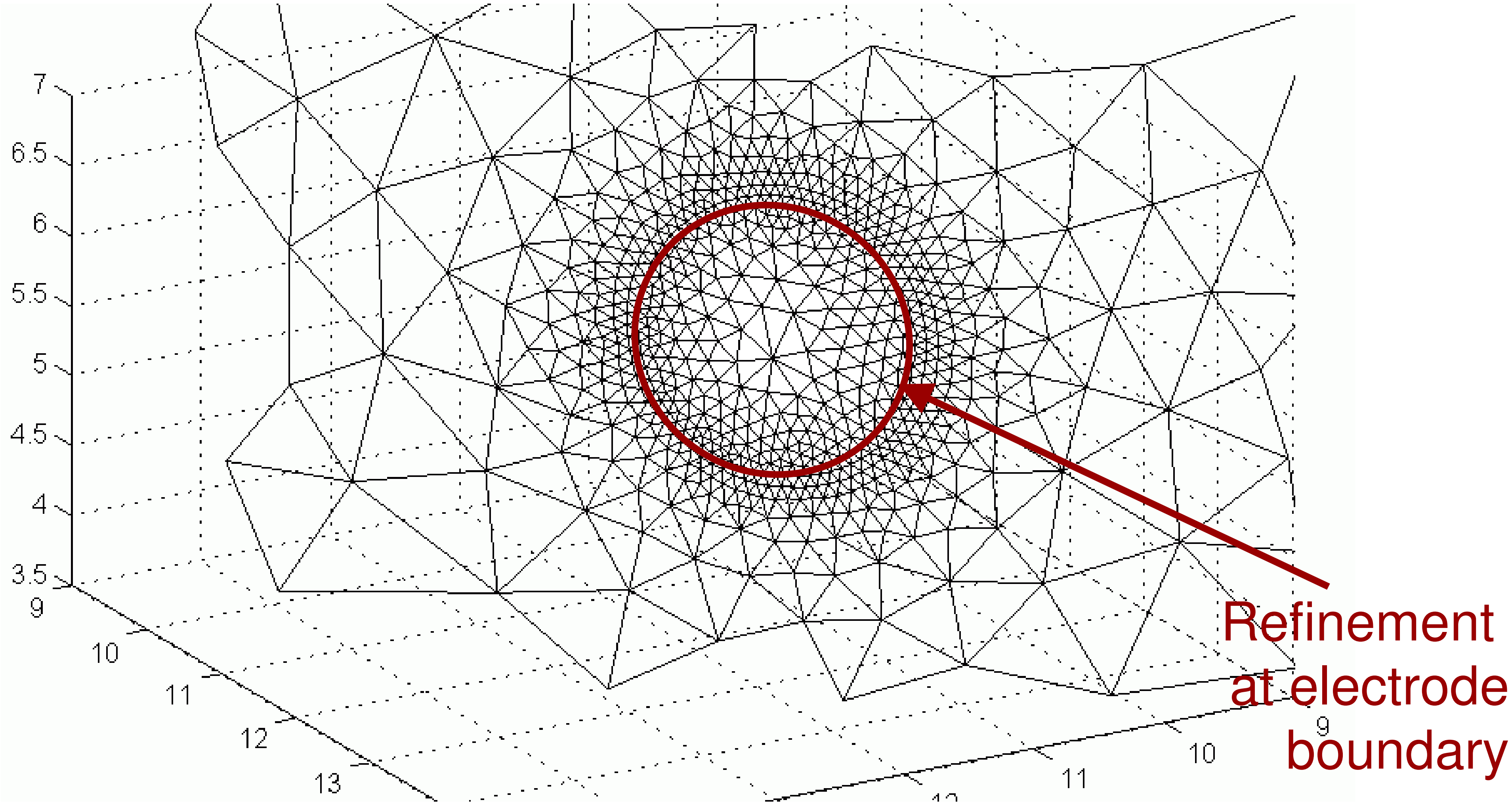
discrete elements make model not smooth

Moving Ball in 16 electrode tank



Electrode models

Need detailed electrode models to avoid geometry errors



Reconstructed Movies

- Algorithm is regularized one-step Gauss-Newton using Laplace prior

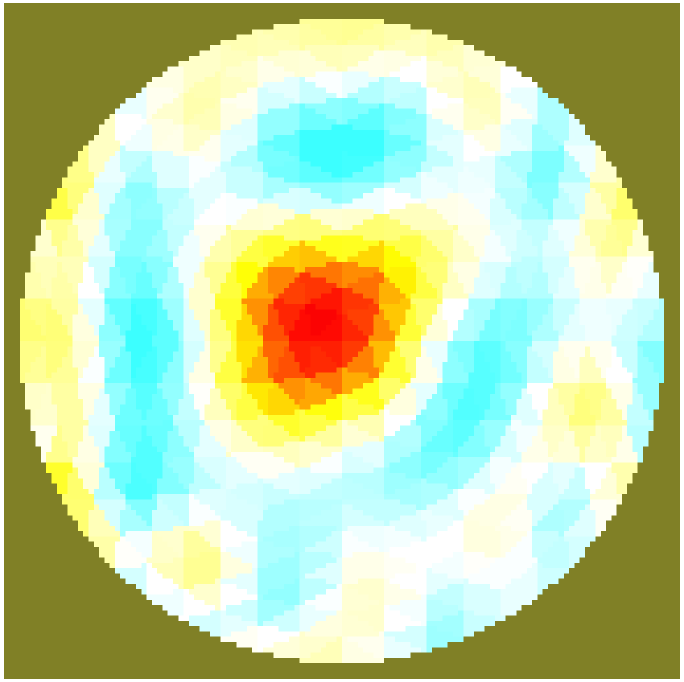
Netgen simulation of moving ball,
Using 100,000 elements per frame

Total simulation time = 3 days

Measurements of moving plexiglas rod in saline tank (thanks to IIRC)

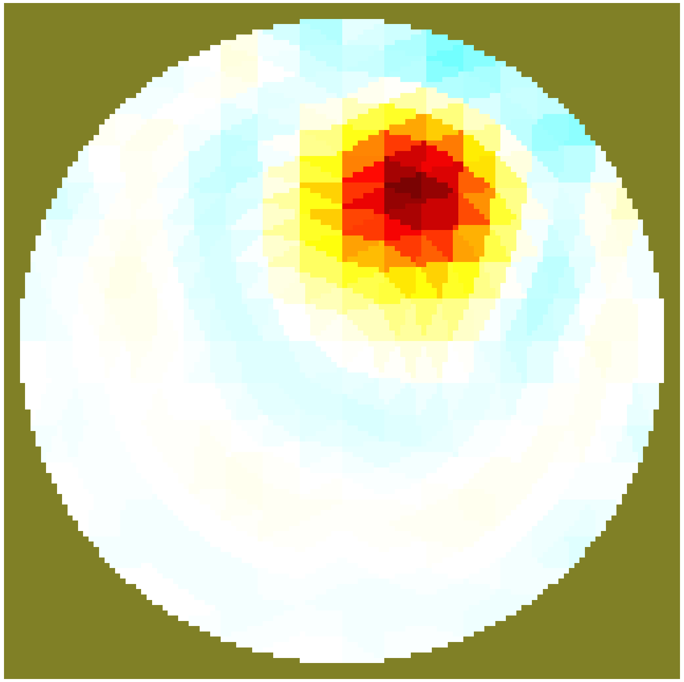
Total model time = 60 seconds

Reconstructed Movies



Netgen simulation of moving ball, (100,000 element FEM)

Simulation time = 3 days



Measurements of moving plexiglas rod in saline tank (thanks to IIRC)

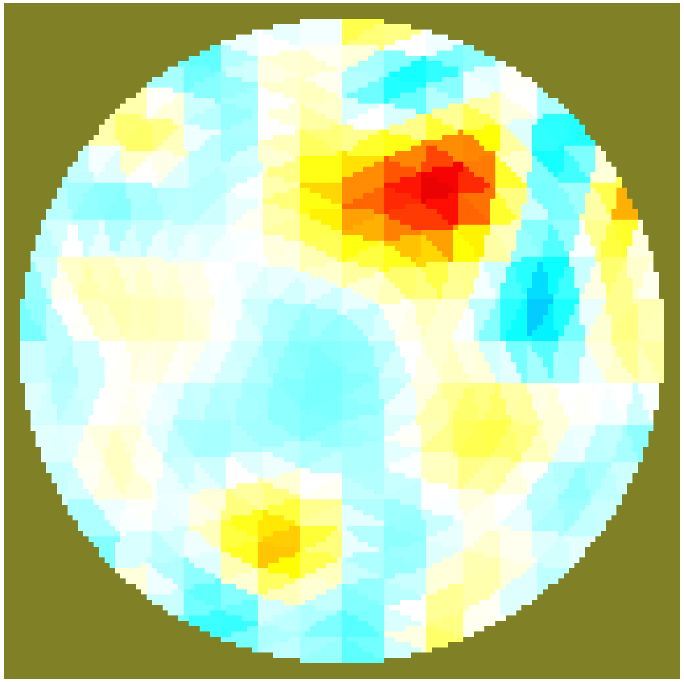
Measurement time = 60 sec

Gauss-Newton vs. Kalman

Data with added 0dB SNR noise

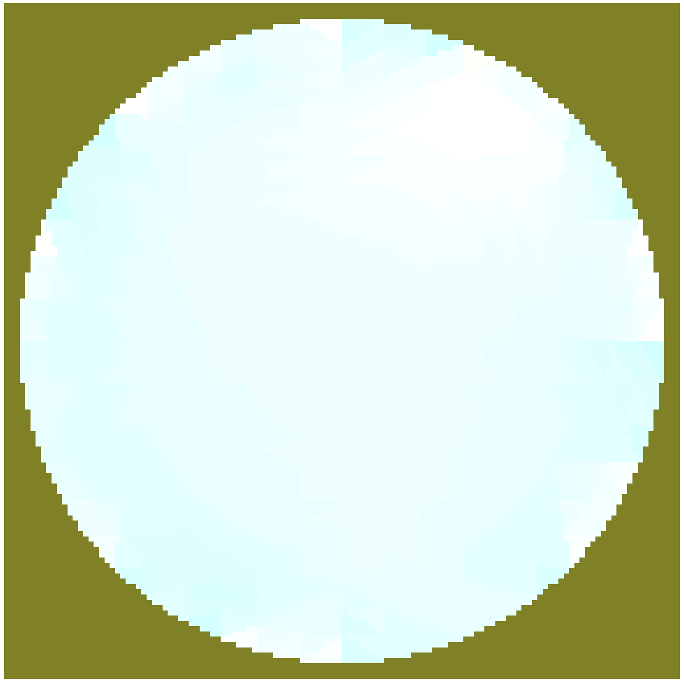
Gauss-Newton solver	Kalman solver
Solve time = 5.33 s (with caching) = 0.22 s	Solve time = 29.6 min

Gauss-Newton vs. Kalman (0dB SNR)



Gauss-Newton solver

Solve time = 5.33 s
(with caching) = 0.22 s



Kalman solver

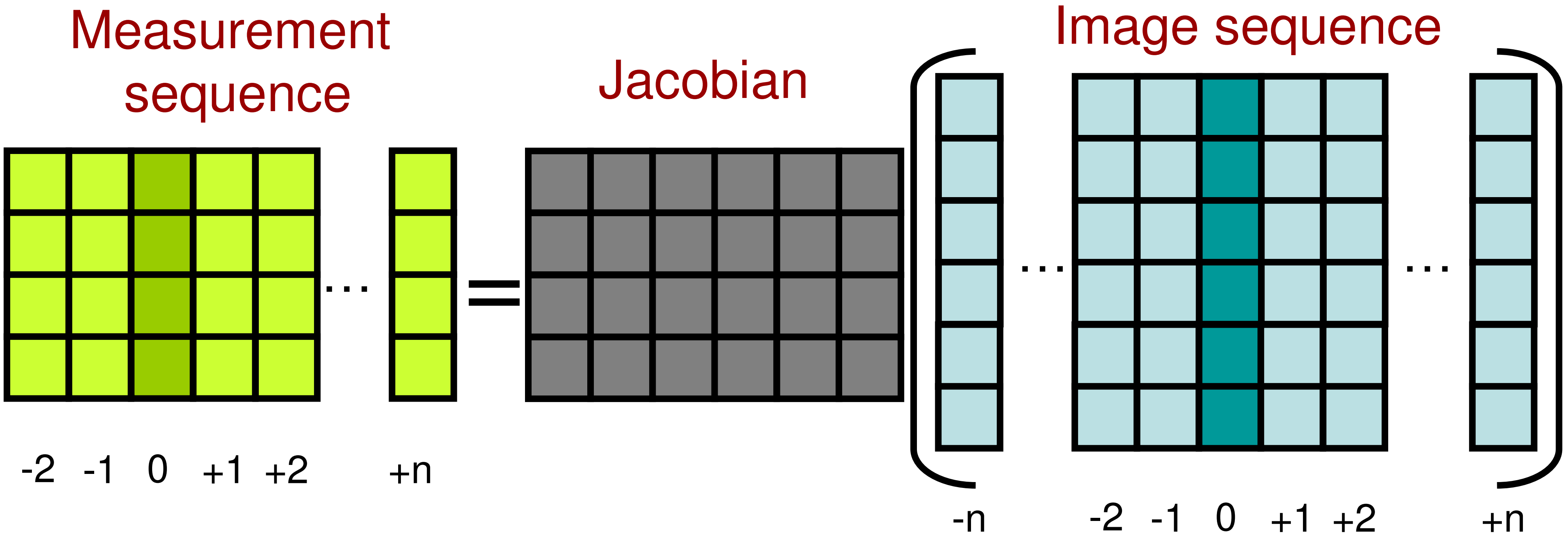
Solve time = 29.6 min

We need a faster solver

We can improve on Kalman in two ways

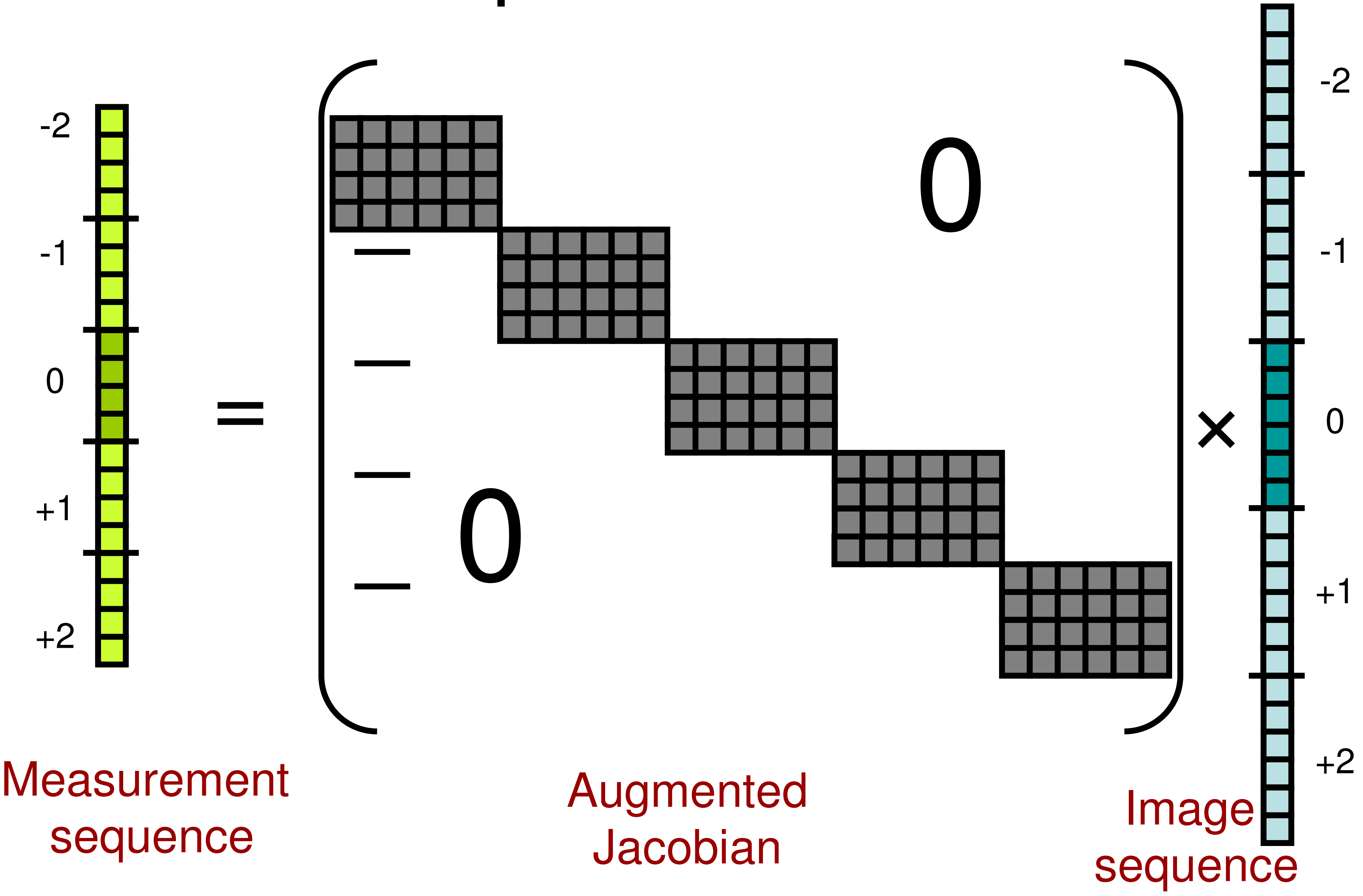
- We can go faster.
  - Kalman calculates the temporal prior. We can directly tell the algorithm
- Use *future* and *past* data
  - Most EIT reconstruction is post-processing
  - For online images, we can delay by a few frames (≈ 100ms)

Direct temporal solver

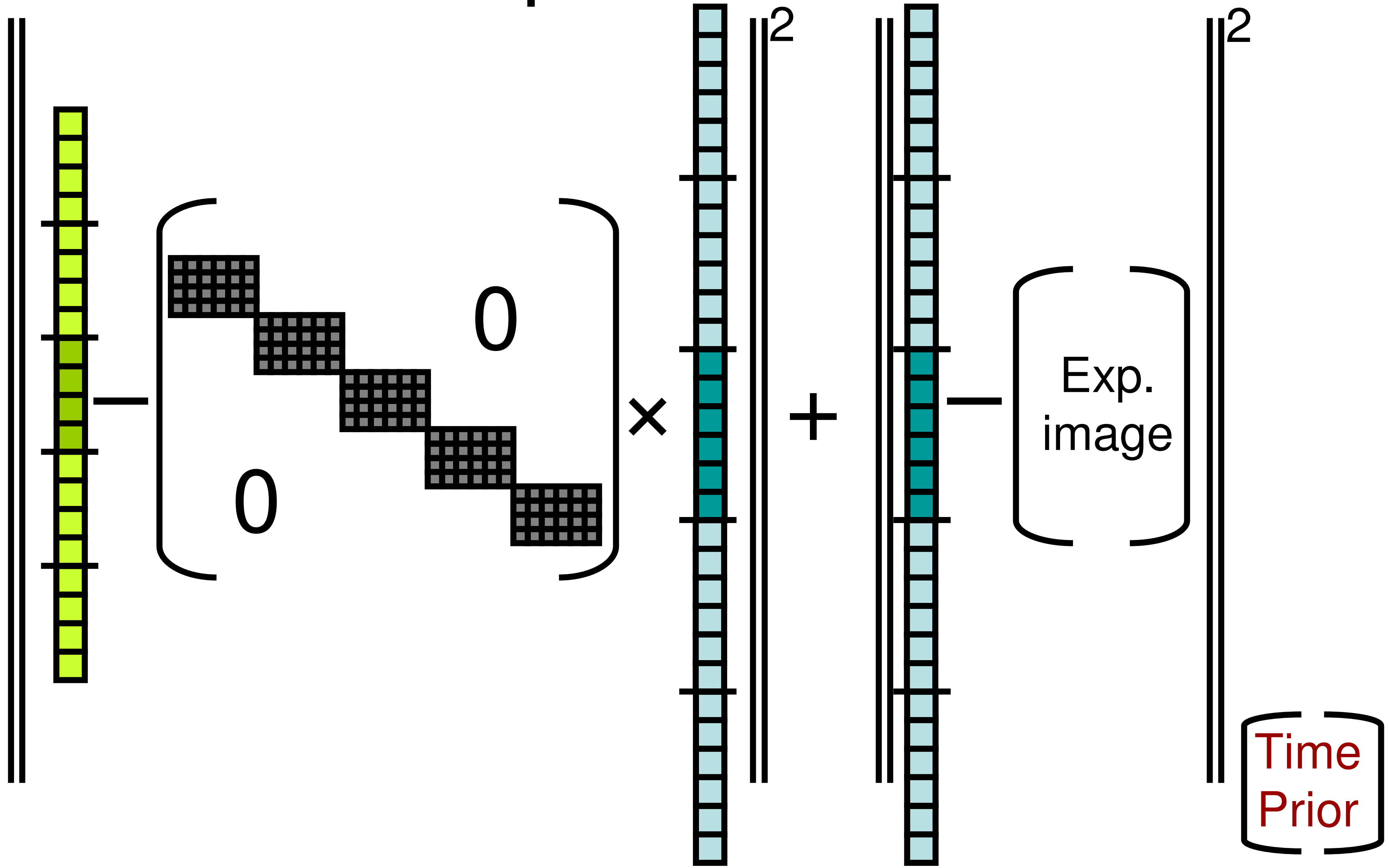


Rewrite as …

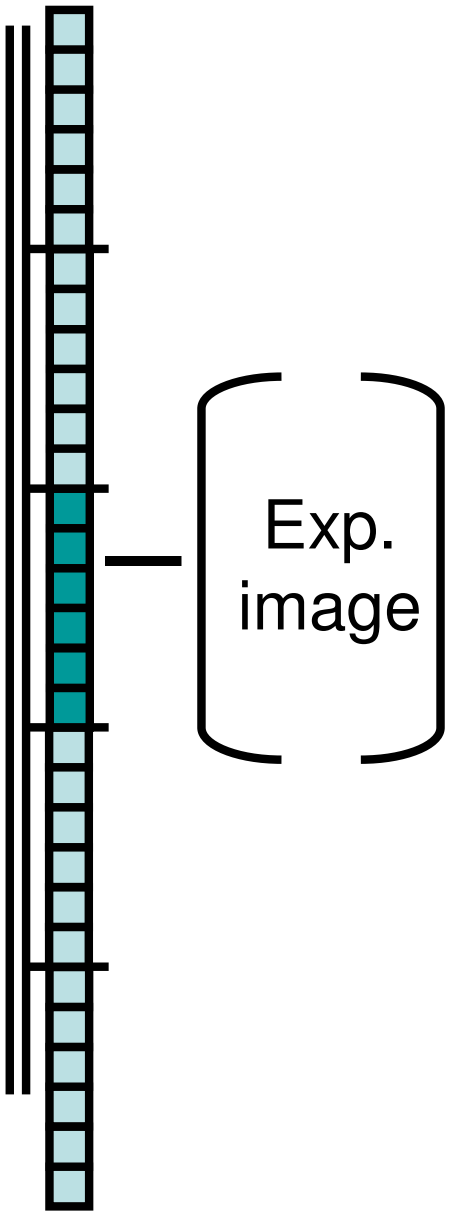
Direct temporal forward model



Direct temporal inverse model



Temporal Priors



Spatial Prior	Time Prior $\Delta t = 1$	Time Prior $\Delta t = 2$	Time Prior $\Delta t = 3$	Time Prior $\Delta t = 4$
Time Prior $\Delta t = 1$	Spatial Prior	Time Prior $\Delta t = 1$	Time Prior $\Delta t = 2$	Time Prior $\Delta t = 3$
Time Prior $\Delta t = 2$	Time Prior $\Delta t = 1$	Spatial Prior	Time Prior $\Delta t = 1$	Time Prior $\Delta t = 2$
Time Prior $\Delta t = 3$	Time Prior $\Delta t = 2$	Time Prior $\Delta t = 1$	Spatial Prior	Time Prior $\Delta t = 1$
Time Prior $\Delta t = 4$	Time Prior $\Delta t = 3$	Time Prior $\Delta t = 2$	Time Prior $\Delta t = 1$	Spatial Prior

One-step inverse

We formulate the one step inverse as:

$$\left\|\mathbf{z}-\mathbf{H}\mathbf{x}\right\|_{\mathbf{W}}^{2}+\lambda^{2}\left\|\mathbf{x}\right\|_{\mathbf{R}}^{2}$$

$$\hat{\mathbf{x}}=\left(\mathbf{H}^{t}\mathbf{W}\mathbf{H}+\lambda^{2}\mathbf{R}\right)^{-1}\mathbf{H}^{t}\mathbf{W}\mathbf{z}$$

Need to cut matrix afterward, we only want to estimate current image from data

Problem is size of matrix inverse:

- For 2 time steps, we have 5 x num_elems square

Underdetermined formulation

We formulate the one step inverse as:

$$\hat{\mathbf{x}} = \left(\mathbf{H}^t \mathbf{W} \mathbf{H} + \lambda^2 \mathbf{R}\right)^{-1} \mathbf{H}^t \mathbf{W} \mathbf{z}$$

$$\hat{\mathbf{x}} = \mathbf{R}^{-1} \mathbf{H}^t \left(\mathbf{H} \mathbf{R}^{-1} \mathbf{H}^t + \lambda^2 \mathbf{W}^{-1}\right)^{-1} \mathbf{z}$$

Now matrix inverse is smaller:

- For 2 time steps, we have 5 x num_meas square

$\mathbf{R}^{-1}$ and $\mathbf{W}^{-1}$ are modelled directly. No need to take the inverse

GN vs. Temporal Inverse

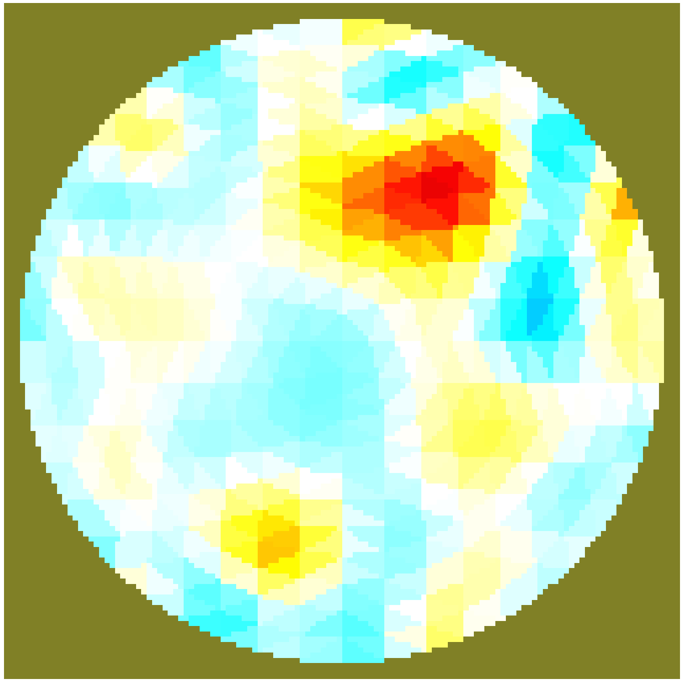
1. Noise free data (IIRC tank)
2. Data with added 6dB SNR noise

Gauss-Newton solver

Solve time = 5.33 s
(with caching) = 0.22 s

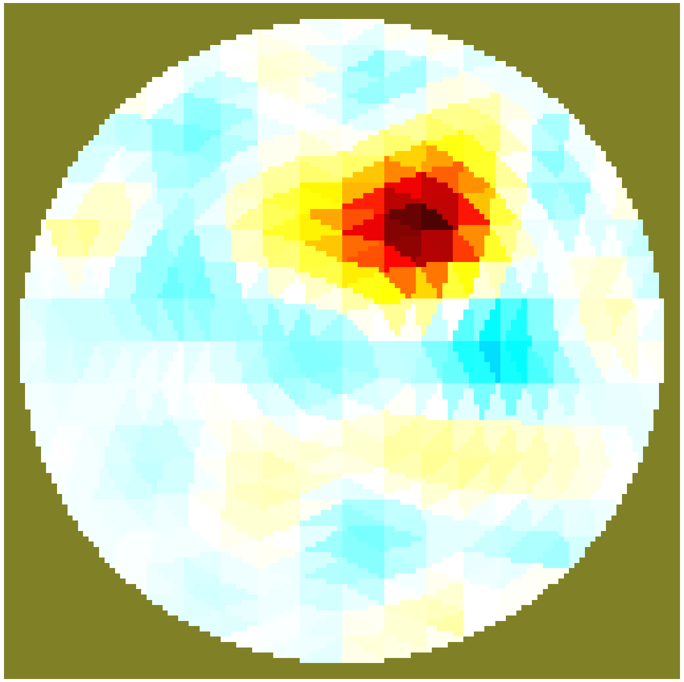
Temporal solver
(4 time steps)
Solve time = 34.81 s
(with caching) = 0.60 s

Gauss Newton vs. Temporal Inverse (6db SNR)



Gauss-Newton solver

Solve time = 5.33 s
(with caching) = 0.22 s



Temporal solver
(4 time steps)
Solve time = 34.81 s
(with caching) = 0.60 s

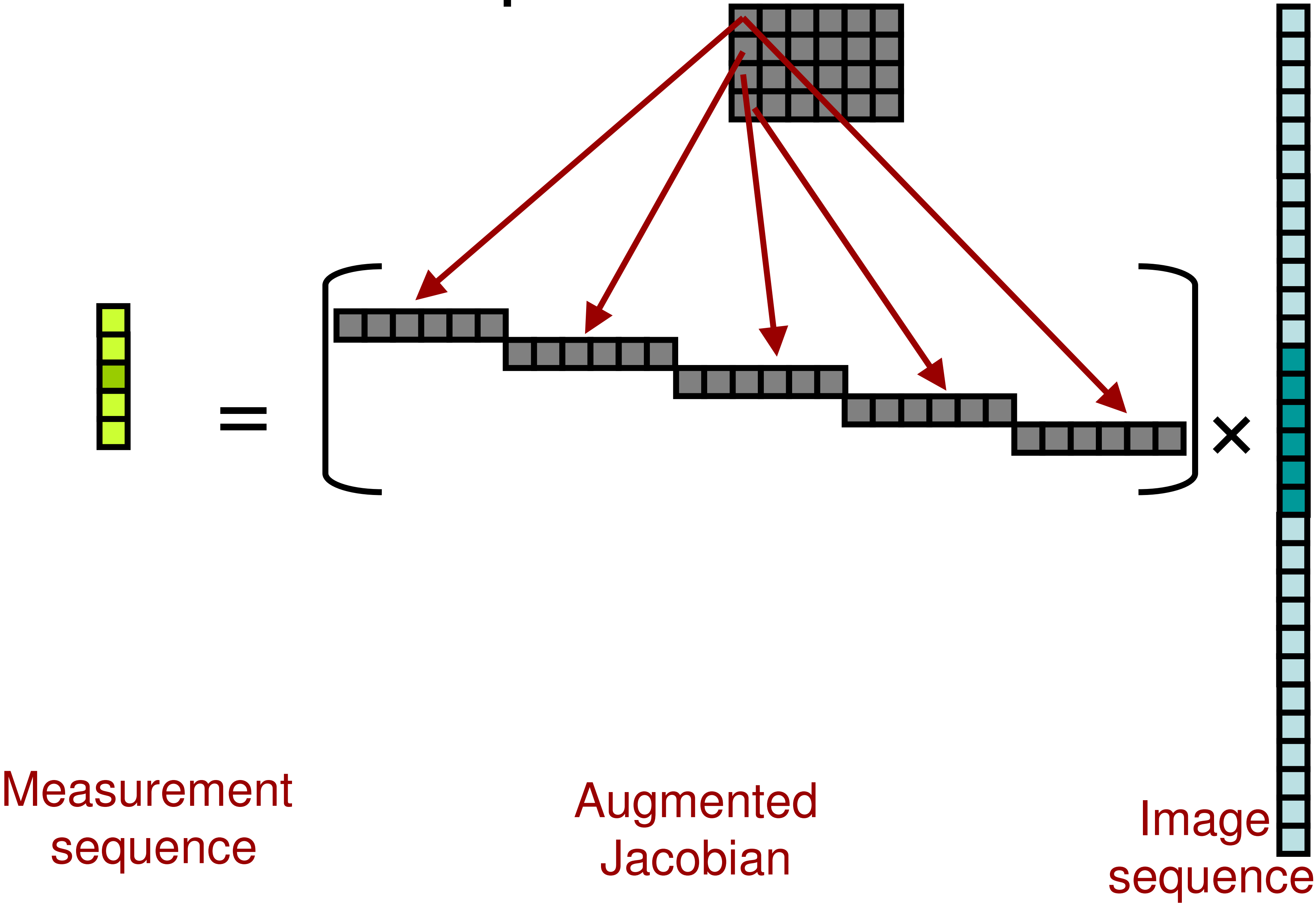
Discussion

- Temporal priors can improve EIT image quality
- Temporal priors can be computationally efficient
  - We're also looking at efficient iterative implementations, allowing reconstruction of entire frame sequence simultaneously

Work in progress: Sequential Stimulations

- One common design for EIT equipment is parallel measurements with sequential current patterns
- This means that the ‘image’ is different at each current pattern instant
- We can formulate this

Direct temporal forward model



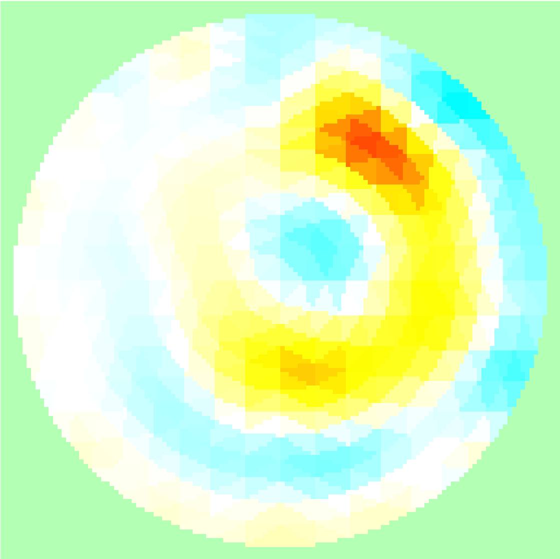
Gauss Newton vs. Kalman

Noise free data (IIRC tank) – only one stimulation pattern kept for each sequence

Gauss Newton solver uses data from nearby frames

Gauss-Newton solver	Temporal solver (4 time steps)
Solve time = 5.33 s	Solve time = 34.81 s
(with caching) = 0.22 s	(with caching) = 0.60 s

Gauss Newton vs. Kalman (sequential – noise free)



Gauss-Newton solver

Kalman solver

Solve time = 5.41 s
(with caching) = 0.38 s

Solve time = 14.2 min