Fast Space-varying Convolution, Fast Matrix Vector Multiplication, and FMRI Activation Detection

Jianing Wei

Advisors: Prof. Jan P. Allebach, Prof. Ilya Pollak
Prof. Charles A. Bouman, Dr. Peter A. Jansson

School of Electrical and Computer Engineering,
Purdue University
August 6, 2009
Fast Space-varying Convolution and Its Application in Stray Light Reduction
Space-varying Convolution

• Space-varying convolution: convolution with a spatially variant point spread function.

• Space-invariant convolution can be computed using FFT.

• Space-varying convolution cannot use FFT, so it is computationally expensive.

• Space-varying convolution finds applications in
  – stray light reduction (Bitlis et. al. 2007 and Wei et. al. 2008)
  – aberration correction (Lam 2003)
  – microscopic imaging (Shaevitz and Fletcher 2007)
  – ...

• Our objective: compute space-varying convolution much faster with small error.
Stray Light Contamination

- Stray light (lens flare) in optical imaging systems
  1. Scattering on lens surfaces
  2. Scattering within transparent glass or plastic lens elements
  3. Undesired reflection between optical element surfaces
Model Formulation

- Model of captured image

\[ y = ((1 - \beta)G + \beta S)x \]

- \( x \) is the underlying image to be restored
- \( y \) is the captured image
- \( S \) is the convolution matrix of stray light
- \( G \) accounts for diffraction and aberration
- \( \beta \) represents the weight of stray light
Stray Light Point Spread Function

- **Features of stray light PSF**
  - Space-varying
  - Large support

- **Model for stray light PSF**

\[
s(i_q, j_q; i_p, j_p) = \frac{1}{z} \left( \frac{1}{1 + \frac{1}{i_p^2 + j_p^2} \left( \frac{(i_p i_q + j_p j_q - i_p^2 - j_p^2)^2}{(c + a(i_p^2 + j_p^2))^2} + \frac{(-i_p j_q + i_q j_p)^2}{(c + b(i_p^2 + j_p^2))^2} \right)^\alpha} \right)
\]

- \( z \) is a normalizing constant
- \((i_p, j_p)\) point source location, \((i_q, j_q)\) response point location
- parameters \(a, b, c, \alpha\) are estimated along with \(\beta\) (Bitlis et. al. 2007 and Wei et. al. 2008)
Stray Light Reduction

• Stray light reduction algorithm (Van Cittert’s method)
  \[ \hat{x} = (1 + \beta)y - \beta S_y \]

• Computational problem
  \[ \bar{x} = S_y \]
  – Directly computing \( S_y \) is quite expensive. For example, for a 6M pixel image, it takes \( 3.6 \times 10^{13} \) (36000G) multiplies.
Theory of Lossy Matrix Source Coding

\[
\begin{pmatrix}
\bar{x}
\end{pmatrix}
= 
\begin{pmatrix}
S
\end{pmatrix}
\begin{pmatrix}
y
\end{pmatrix}
\]

Compress \( S \) – lossy matrix source coding

- Question: How is error due to matrix compression \( \| \delta S \|_F^2 \) related to error in output \( \| \delta \bar{x} \|_2^2 \) ?

- Answer: when the empirical autocorrelation of \( y \) is identity, then \( \| \delta \bar{x} \|_2^2 \) is equal to \( \| \delta S \|_F^2 \).
Matrix Source Coding Approach

\[ \tilde{x} = S y = W^{-1} WST^{-1} \Lambda^{1/2} \Lambda^{-1/2} T y \]

- \( W \) and \( T \) are both wavelet transforms (decorrelation)
- \( T y \) approximately decorrelates \( y \)
- \( \Lambda^{-1/2} \) is diagonal matrix which normalizes autocorrelation of \( T y \)
  \[ \tilde{y} = \Lambda^{-1/2} T y \quad R_{\tilde{y}} = E[\tilde{y}\tilde{y}'] = I \]

- Compress \( S \)
  - Transform to a space where it is sparse
    \[ \tilde{S} = WST^{-1} \Lambda^{1/2} \]
  - Quantize to save storage and computation:
    \[ [\tilde{S}] = [WST^{-1} \Lambda^{1/2}] \]
Fast Convolution Using Matrix Source Coding

• Online computation:
  \[ \tilde{x} \approx \tilde{x} = W^{-1} \begin{bmatrix} \tilde{S} \end{bmatrix} \tilde{y} \]
  – wavelet transforms on image + sparse matrix vector multiply
  – complexity \( O(P) \), where \( P \) is the number of pixels

• Offline computation:
  \[ [\tilde{S}] = [WST^{-1} \Lambda^{1/2}] \]
  – wavelet transforms along rows and columns of huge matrix
  – complexity \( O(P^2) \), need to be improved to \( O(P) \).
Reduction of Offline Computation

- **Problem:** Offline computation too expensive -- order $O(P^2)$.

- **Solution:** Two stage reduction to sparse matrix

\[
\tilde{S} \approx W \left[ ST^{-1} \Lambda^{1/2} \right]
\]

- wavelet transform $W$ on sparse data $\left[ ST^{-1} \Lambda^{1/2} \right]$ saves time

- $\left[ ST^{-1} \Lambda^{1/2} \right]$ can be directly computed with order $O(P)$

- resulting formula for convolution

\[
\tilde{x} = W^{-1} \left[ W \left[ ST^{-1} \Lambda^{1/2} \right] \right] \left( \Lambda^{-1/2} Ty \right)
\]
Fast Computation of Sparse Haar Wavelet Coefficients

• Objective: directly compute $\begin{bmatrix} ST^{-1} \Lambda^{1/2} \end{bmatrix}$

• What is $ST^{-1} \Lambda^{1/2}$?
  \[ ST^{-1} \Lambda^{1/2} = (TS')' \Lambda^{1/2} \]
  – wavelet transform along the rows of $S$, then scale

• Our strategy for each row of $S$
  1. Locate “important” wavelet coefficients
     • “Important”: nonzero after multiplying with $\Lambda^{1/2}$ and quantization
  2. Compute these “important” wavelet coefficients
     • a top-down approach based on recursion
     • reduce the computation from $O(P)$ to $O(1)$
  3. Apply scaling factors $\Lambda^{1/2}$
Locate “Important” Wavelet Coefficients

• For image size 1024x1024, take 49 PSFs due to 49 equally spaced point sources.
• Compute Haar wavelet transform of them.
• Apply scaling factors and quantize the result so that 1000 coefficients survive.
• Find out the relationship between location of “important” wavelet coefficients and the location of point source and level of transform.
Locate “Important” Wavelet Coefficients (continued)

• Predicting location of “important” wavelet coefficients at level $k$
  – location of red circles are related to point source location and level of transform:
    • $i_q^{(k)} = i_q / 2^k$, $j_q^{(k)} = j_q / 2^k$
    • radius $r^{(k)}$ is not related to $(i_q, j_q)$. 

$(i_q^{(1)}, j_q^{(1)})$, $(r^{(1)})$
Computation of Sparse Haar Wavelet Coefficients

- Use Haar wavelet for simple and fast computation.
- Wavelet detail coefficients can be computed from approximation coefficients.
- We only need to compute necessary approximation coefficients, since many detail coefficients are zero.
Computation of Sparse Haar Approximation Coefficients

- Recursive approach for computing sparse approximation coefficients $a[k][m][n]$
  $a[k][m][n] = \frac{1}{2} \left( a[k - 1][2m][2n] + a[k - 1][2m][2n + 1] + a[k - 1][2m + 1][2n] + a[k - 1][2m + 1][2n + 1] \right)$

- Top-down approach

- When all corresponding details coefficients are zeros, it means approximation is perfect. So stop and return value
  $a[k][m][n] = 2^k a[0][2^k m][2^k n]$
Simulation Experiments

- We use Cohen-Daubechies-Feauveau 5-3 wavelet for $W$, Haar wavelet for $T$.
- Three training images are used to obtain gain factors $\Lambda$:
  - take Haar wavelet transform of each training image
  - compute variance of wavelet coefficient in each band, and average over all training images
- Error metric: normalized root mean square error

$$\text{NRMSE} = \sqrt{\frac{\|\tilde{x} - \bar{x}\|^2}{\|\tilde{x}\|^2}}$$

where $\tilde{x} = S_y$

$$\tilde{x} = W^{-1}[W ST^{-1} \Lambda^{1/2} ](\Lambda^{-1/2} T_y)$$
Distortion-Rate Curves

Comparison between with and without transform for image size 256x256

No transform: \( \tilde{x} = [S]y \)

Note that for image size 1024x1024, we reduce the multiplies per output point from \( 1024^2 \) to 10 with only 2% error. The storage is 10M for 1024x1024 image.
Captured Image
Restored Image
Conclusion

- We characterized stray light point spread function and used a deconvolution algorithm to restore images.
- We applied matrix source coding approach to space-varying convolution in the restoration algorithm to achieve reduction in complexity.
- We developed recursive approach for computing sparse Haar wavelet coefficients to save offline computation.
- Experimental results showed that our algorithm can reduce the computation by a significant amount, e.g. for 1024x1024 image, we can achieve a $10^5$:1 reduction in computation, with a small amount of error.
Fast Matrix Vector Multiplication Using Sparse Matrix Transform
Computational Problem

- \[ y = Ax \]
  - \( x \) – input vector \( P \times 1 \)
  - \( y \) – output vector \( P \times 1 \)
  - \( A \) – transform matrix \( P \times P \), big and dense!

- Widely used:
  - stray light reduction
  - large scale electromagnetic integral equation

- Problem:
  - cannot construct or store \( A \) when \( P \) is large
  - complexity \( O(P^2) \) is too high

- Objective:
  - compute \( Ax \) much faster with small error
Background on SMT

- SMT of order $K$
  \[ T = \prod_{k=1}^{K} T_k = T_1 T_2 \cdots T_K \]
  - Each $T_k$ is a sparse matrix
  - Each $T_k$ operates on two coordinates

- SMT for computing ML estimate of covariance matrix given $M$ ($M<P$) sample data vectors
  \[ [T, \Lambda] = \text{SMTCovEst}\left(\{a_1, a_2, \ldots, a_M\}\right) \]
  - $\{a_m\}$ are sample vectors
  - $T$ is a SMT matrix, $\Lambda$ is diagonal
  - $T\Lambda T^t$ is an estimate of $E[a_m a_m']$
Our Approach

• Basic idea: use $N (N << P)$ training input $(X, O)$ and output $(Y, Z)$ vectors to find approximate SVD of $A$.

$$Y = AX \quad Z = A^t O$$

– $X, O$: impulses at random locations
– Jointly train rows and columns of $A$

• Minimize cost

$$c(U,V,\Sigma) = \|U^tY - \Sigma V^t X\|_F^2 + \|V^tZ - \Sigma U^t O\|_F^2$$

– where $U$ and $V$ are orthonormal SMT matrices, $\Sigma$ is diagonal

• Approximate SVD of $A$: $U \Sigma V^t$

• Approximate $Ax$ with $U\Sigma V^t x$

– much faster when order of SMT is low ($<< P^2$)
– further study needed: what kind of matrix $A$ makes it a good approximation?
Details of Our Algorithm

• Two stage approach:
  1. Pre-processing: perform SMT on input and output
     - make second stage optimization easier
  2. Find $\tilde{U}$, $\tilde{V}$, and $\Sigma$ to minimize cost

$$c(U, V, \Sigma) = \left\| \tilde{U}^t \tilde{Y}_0 - \Sigma \tilde{V}^t \tilde{X}_0 \right\|_F^2 + \left\| \tilde{V}^t \tilde{Z}_0 - \Sigma \tilde{U}^t \tilde{O}_0 \right\|_F^2$$

where $\tilde{U}$ and $\tilde{V}$ are orthonormal SMT matrices, $\tilde{X}_0, \tilde{Y}_0, \tilde{O}_0, \tilde{Z}_0$ are processed input and output from first stage.
SMT Pre-processing

- Sample $M$ rows of matrix $A$ to get $A_r$ of size $M \times P$.
  
  $$[\tilde{E}_0, \Lambda_1] = \text{SMTCovEst}(A_r')$$
  
  $\tilde{E}_0$ approximately decorrelates the columns of $A$.

- Sample $M$ columns of matrix $A$ to get $A_c$ of size $P \times M$.
  
  $$[\tilde{F}_0, \Lambda_2] = \text{SMTCovEst}(A_c)$$
  
  $\tilde{F}_0'$ approximately decorrelates the rows of $A$.

- Process training data
  
  $$\tilde{X}_0 = \tilde{E}_0'X \quad \tilde{Y}_0 = \tilde{F}_0'Y$$
  $$\tilde{O}_0 = \tilde{F}_0'O \quad \tilde{Z}_0 = \tilde{E}_0'Z$$
Iterative Approach for Minimizing Cost

- Idea: only operate on two coordinates \((i,j)\) in each iteration to reduce the cost

\[
\begin{align*}
\tilde{Y}_{k-1}^{(i,j)} & \approx T_{k,(i,j)}^{(i,j)} \times \tilde{Y}_{k-1}^{(i,j)} \\
\tilde{Z}_{k-1}^{(i,j)} & \approx (T_{k,(i,j)}^{(i,j)})^t \times \tilde{Z}_{k-1}^{(i,j)}
\end{align*}
\]

- Keep updating training input and output by orthonormal sparse transforms in order to diagonalize the matrix relating them
Optimization Algorithm

• Initialization: first estimate of $\Sigma$

$$
\Sigma_0 = \min_{\Sigma} \|\tilde{Y}_0 - \Sigma \tilde{X}_0\|^2_F + \|\tilde{Z}_0 - \Sigma \tilde{O}_0\|^2_F 
$$

$$
\Sigma_{0,(i,j)} = \frac{\langle \tilde{Y}_{0,(i,:)}, \tilde{X}_{0,(i,:)}\rangle + \langle \tilde{Z}_{0,(i,:)}, \tilde{O}_{0,(i,:)}\rangle}{\langle \tilde{X}_{0,(i,:)}, \tilde{X}_{0,(i,:)}\rangle + \langle \tilde{Z}_{0,(i,:)}, \tilde{Z}_{0,(i,:)}\rangle}
$$

• For each iteration $k$,
  – find a pair of coordinates $(i,j)$ with maximum cost reduction
  
   – compute SVD of corresponding optimal transform matrix $T_{k,(i,j)}^{(i,j)}$
   
   $$
   T_{k,(i,j)}^{(i,j)} = U_k^{(i,j)} D_k^{(i,j)} (V_k^{(i,j)})^t
   $$
  
   – update training vectors and diagonal matrix at coordinates $(i,j)$
  
   – update cost reduction for related coordinates
Optimal Transform for Two Coordinates

- Optimal transform for coordinates \((i,j)\) at iteration \(k\):
  \[
  T_{k,(i,j)}^{(i,j)} = \arg \max \left( c_{k-1,(i,j)} - c_{k,(i,j)} \right)
  \]
  - where
  \[
  c_{k-1,(i,j)} - c_{k,(i,j)} = \| \tilde{Y}_{k-1}^{(i,j)} - \Sigma^{(i,j)} \tilde{X}_{k-1}^{(i,j)} \|_F^2 + \| \tilde{Z}_{k-1}^{(i,j)} - \Sigma^{(i,j)} \tilde{O}_{k-1}^{(i,j)} \|_F^2
  \]
  \[
  - \| \tilde{Y}_{k-1}^{(i,j)} - T_{k,(i,j)}^{(i,j)} \tilde{X}_{k-1}^{(i,j)} \|_F^2 - \| \tilde{Z}_{k-1}^{(i,j)} - (T_{k,(i,j)}^{(i,j)})^t \tilde{O}_{k-1}^{(i,j)} \|_F^2
  \]

- Solution (remove \((i,j)\) and \(k\) sub/sup scripts):
  \[
  \text{vec}(T) = \left( I \otimes (\tilde{O}\tilde{O}^t) + (\tilde{X}\tilde{X}^t) \otimes I \right)^{-1} \text{vec}(\tilde{O}\tilde{Z}^t + \tilde{Y}\tilde{X}^t)
  \]
  - where vec means stacking columns to form a vector, \(\otimes\) means Kronecker product.
Update Training Vectors

• For coordinates \((i,j)\) with maximum cost reduction
  – Perform SVD of transform matrix
    \[
    T_{k,(i,j)}^{(i,j)} = U_k^{(i,j)} D_k^{(i,j)} \left( V_k^{(i,j)} \right)
    \]
  – Update input and output vectors
    \[
    \tilde{X}_k^{(i,j)} = \left( V_k^{(i,j)} \right)^\top \tilde{X}_{k-1}^{(i,j)}
    \]
    \[
    \tilde{Y}_k^{(i,j)} = \left( U_k^{(i,j)} \right)^\top \tilde{Y}_{k-1}^{(i,j)}
    \]
    \[
    \tilde{O}_k^{(i,j)} = \left( U_k^{(i,j)} \right)^\top \tilde{O}_{k-1}^{(i,j)}
    \]
    \[
    \tilde{Z}_k^{(i,j)} = \left( V_k^{(i,j)} \right)^\top \tilde{Z}_{k-1}^{(i,j)}
    \]
  – Update estimate of \(\Sigma\)
    \[
    \Sigma_{k,(i,i)} = D_{k,(1,1)}^{(i,j)}
    \]
    \[
    \Sigma_{k,(j,j)} = D_{k,(2,2)}^{(i,j)}
    \]
SVD Using SMT

• After $K$ iterations, we have

$$U_K^t \cdots U_1^t F_0^t Y \approx \Sigma_K V_K^t \cdots V_1^t E_0^t X$$

$$V_K^t \cdots V_1^t E_0^t Z \approx \Sigma_K U_K^t \cdots U_1^t F_0^t O$$

• Approximate SVD of $A$

$$A \approx F_0 U_1 \cdots U_K \Sigma_K V_K^t \cdots V_1^t E_0^t X$$

• Fast matrix vector multiplication

$$\hat{y} = F_0 U_1 \cdots U_K \Sigma_K V_K^t \cdots V_1^t E_0^t x$$
Computational Complexity

• Searching for \((i,j)\) with maximum cost reduction in each iteration is \(\log P\)
  - Neighborhood based search
  - Red-Black tree data structure

• SVD using SMT costs \(O((K+K_0)\log P)\)
  - Regular SVD costs \(O(P^3)\)

• Fast matrix vector multiplication costs \(O(K+K_0+P)\)
  - Rigorous implementation costs \(O(P^2)\)
A Simple Experiment

- Image size 256x256, which means $P=65536$

- Matrix $A$ is a convolution matrix

$$A_{j,i} = a(x_j, y_j; x_i, y_i),$$

- where $a(x_j, y_j; x_i, y_i)$ is the PSF due to point source $(x_i, y_i)$

$$a(x_j, y_j; x_i, y_i) = \frac{(H - |y_j - y_i|)(W - |x_j - x_i|)}{HW}$$

- We set $M=256$, $N=256$.

- Testing on a natural image

- Error is measured in normalized root mean square error
Distortion vs. Number of Rotations

\[ NRMSE = \sqrt{\frac{||y - \hat{y}||_2^2}{||y||_2^2}} \]
Conclusion and Future Work

• Conclusion:
  – We developed an algorithm to perform approximate SVD of a huge dense matrix using SMT.
  – We reduced the computation of matrix vector multiplication from $O(P^2)$ to $O(K+K_0+P)$ with a small amount of error.

• Future work:
  – Further speed up by quantizing estimated singular values
  – Experiment with more matrices (PSFs)
  – Work with larger images
FMRI Activation Detection
Problem Description

• FMRI data is 3D/4D data which contains a temporal dimension and two or three spatial dimensions.
• Data collected while the subject is presented with a stimulus (in our case, a visual stimulus.)
• Objective: detect activated regions in the brain from a sequence of fMRI images.
Contributions

• We propose a forward model which simultaneously captures spatial and temporal dependencies of the data.
• We develop an effective algorithm for estimating model parameters.
• We introduce total-variation based restoration as a very effective pre-processing tool for fMRI data.
• We adopt Markov random field (MRF) model in fMRI analysis which results in spatially regularized and robust estimate of the parameter map.
Model Formulation

\[ y_{i,j,t} = \sum_{k} \sum_{l} g_{k,l} \left( h_{i-k,j-l,t} + w_{i-k,j-l,t} \right) + \eta_{i,j,t} \]

- \( y_{i,j,t} \) – observed data at location \((i,j)\), time \(t\).
- \( h_{i,j,t} \) – hemodynamic response function (HRF) at location \((i,j)\), time \(t\).
- \( w_{i,j,t} \) – physiological noise at location \((i,j)\), time \(t\).
  - Modeled by AR(1) process \( w_{i,j,t} = \rho w_{i,j,t-1} + \epsilon_{i,j,t} \) where \( \epsilon_{i,j,t} \) is white noise with mean zero and standard deviation \( \sigma_\epsilon \).
- \( \eta_{i,j,t} \) – scanner noise at location \((i,j)\), time \(t\), modeled as white noise with mean zero and standard deviation \( \sigma_\eta \).
- \( g_{k,l} \) – Gaussian blurring kernel characterized by width \( \sigma \).
**HRF Models**

- **Gamma variate model (Dale and Buckner 1997)**

\[
h_{t,ij} = \begin{cases} 
   x_{i,j} \left( \frac{t - \delta_{i,j}}{\tau_{i,j}} \right)^2 e^{\frac{-t-\delta_{i,j}}{\tau_{i,j}}} & \text{if } t \geq \delta_{i,j} \\
   0 & \text{if } t < \delta_{i,j}
\end{cases}
\]

- parameter set \( \theta_{i,j} = \{x_{i,j}, \delta_{i,j}, \tau_{i,j}\} \)
- \( x_{i,j} \) amplitude parameter
- \( \tau_{i,j} \) dispersion parameter

- **Model with undershoot (Friston et al. 1998)**

\[
h_{t,ij} = x_{i,j} \left( \frac{t}{a_{i,j}^{(1)} b_{i,j}^{(1)}} \right)^{a_{i,j}^{(1)} b_{i,j}^{(1)}} e^{\frac{-t-d_{i,j}^{(1)} h_{i,j}^{(1)}}{b_{i,j}^{(1)}}} - c_{i,j} \left( \frac{t}{a_{i,j}^{(2)} b_{i,j}^{(2)}} \right)^{a_{i,j}^{(2)} b_{i,j}^{(2)}} e^{\frac{-t-d_{i,j}^{(2)} h_{i,j}^{(2)}}{b_{i,j}^{(2)}}}
\]

- parameter set \( \theta_{i,j} = \{x_{i,j}, a_{i,j}^{(1)}, b_{i,j}^{(1)}, a_{i,j}^{(2)}, b_{i,j}^{(2)}, c_{i,j}\} \)
- \( x_{i,j} \) amplitude parameter
- \( b_{i,j}^{(1)} \) dispersion parameter
Overall Activation Detection Strategy

• Restore each frame using constrained total variation minimization (Goldfarb and Yin 2005)

• Estimate the HRF parameters using spatial regularization
  – assume the parameters follow generalized Gaussian Markov Random Field (GGMRF) distribution
  – perform maximum a posteriori estimation

• Threshold the amplitude parameters to obtain the activation map
Constrained Total Variation Minimization

- The optimization problem is formulated as:

$$\min_{\{h_{i,j}\}} \sum_i \sum_j \sqrt{(h_{i+1,j} - h_{i,j})^2 + (h_{i,j+1} - h_{i,j})^2}$$

subject to $$y_{i,j} = \sum_k \sum_l g_{k,l} h_{i-k,j-l} + v_{i,j}$$

and $$\frac{1}{N^2} \sum_i \sum_j v_{i,j}^2 \leq \sigma_v^2$$

where $$\sigma_v^2 = \sum_{k,l} g_{k,l}^2 \frac{\sigma_\varepsilon^2}{1 - \rho} + \sigma_\eta^2.$$  

- $$\sigma, \rho, \sigma_\varepsilon, \sigma_\eta$$ are estimated based on spatial and temporal correlations of the data prior to restoration (Wei, Talavage, and Pollak 2007)

- Solved with an interior point method
Models for Parameters and Restored Data

• We model the data after restoration as
  \[ \hat{h}_{i,j,t} = h_{i,j,t}(\theta_{i,j}) + e_{i,j,t} \]
  - \( e_{i,j,t} \) -- i.i.d. Gaussian random variables

• Conditional distribution:
  \[
  p(\hat{h}_{i,j} | \theta_{i,j}) = \frac{1}{(2\pi)^{T/2} \sigma_e^{T}} \exp\left\{ -\frac{\| \hat{h}_{i,j} - h_{i,j}(\theta_{i,j}) \|^2}{2\sigma_e^2} \right\}
  \]

• Prior distribution of parameters (GGMRF)
  \[
  p(\theta^{(r)}) = \frac{1}{z_r} \exp\left\{ -\frac{1}{p\sigma_r^p} \sum_{(i,j) \text{ and } (k,l) \text{ neighbors}} K_{i-k,j-l} |\theta_{i,j}^{(r)} - \theta_{k,l}^{(r)}|^{p} \right\}
  \]
  - \( r=1,\ldots, R \). E.g., for the Gamma variate HRF,
    \[ \theta_{i,j}^{(1)} = x_{i,j}, \quad \theta_{i,j}^{(2)} = \delta_{i,j}, \quad \theta_{i,j}^{(3)} = \tau_{i,j} \]
  - assume the \( R \) parameters are mutually independent
Regularized HRF Parameter Estimation

• Joint posterior distribution

\[ p(\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(R)} | \hat{h}) = \prod_{(i,j)} p(h_{i,j} | \theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(R)}) p(\theta^{(1)}) \ldots p(\theta^{(R)}) / p(\hat{h}) \]

• Maximum a posteriori estimate

\[ \{ \theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(R)} \} = \text{arg max}_{\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(R)}} p(\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(R)} | \hat{h}) \]

• Cost function to minimize

\[ c(\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(R)}) = \frac{1}{2 \sigma_e^2} \sum_{(i,j)} \| \hat{h}_{i,j} - h_{i,j}(\theta_{i,j}) \|_2^2 + \sum_{(i,j) \text{ and } (k,l) \text{ neighbors}} K_{i-k,j-l} \sum_{r=1}^{R} \frac{1}{p \sigma_r^p} |\theta^{(r)}_{i,j} - \theta^{(r)}_{k,l}|^p \]

  – iterative coordinate descend (ICD) algorithm is used
GLM Framework

• General linear model (GLM)

$$\hat{h}_{i,j} = Gx_{i,j} + e_{i,j}$$

- $G$ is the regressor with three columns
  
  - HRF fitted by the averaged temporal data
  
  - HRF temporal and dispersion derivatives to account for temporal shift and change in shape of response
  
  - orthonormalize these three vectors to build $G$

- least squares estimate of regression coefficients

$$\hat{x}_{i,j} = (G'G)^{-1}G'\hat{h}_{i,j}$$

- obtain activation map by thresholding the $t$-statistic

$$t_{i,j} = \frac{c'\hat{x}_{i,j}}{\sqrt{(\hat{h}_{i,j} - G\hat{x}_{i,j})'(\hat{h}_{i,j} - G\hat{x}_{i,j})c'(G'G)^{-1}c / (T - 3)}}$$

$c$ is a contrast vector – set to (1 0 0)' in our experiment
GLM Implementation

• Pre-processing: Gaussian smoothing
  – Advantage: fast
  – Problem:
    • performance is very significantly affected by changing the width of Gaussian kernel

• Summary: spatial regularization + linear temporal analysis
Simulated Experiment

- Active region
- Image at t=5 (peak time)
- Restored image
- Regularized amplitude estimate
- Detection result
Simulation Results

ROC curve for Gamma Variate model

ROC curve for HRF model with undershoot

Note that in these experiments:
- CNR=0.5
- Data size 64x64x135, representing image size 64x64, and 9 trials with 15 time points in each trial.
Robustness of Our Algorithm

- Cox waveform is used to generate the data, whereas the Gamma Variate model is used for detection.

- If we change regularization parameters $\sigma_e, \sigma_1, ..., \sigma_R$ separately by 50%, the maximum change in correct detection rate under the same false alarm rate is 1%.
Different Activation Waveform at Different Location

• Two activation regions
• Each with different HRF
Real Data Experiment

- Two types of visual checkerboard stimuli: left hemifield stimulus and right hemifield stimulus.
- Determine regularization parameters from left hemifield stimulus data (training) and use it on right hemifield stimulus data.
- Threshold the activation map such that 16 pixels are declared active for all methods.
Real Data Results (proposed method)

Stimulus 1

Stimulus 2
Real Data Results (GLM method)

Stimulus 1

Stimulus 2
Benchmark Results

Benchmark results are obtained from block paradigm experiments, which are supposed to have higher detection power.
Thank you!
Backup Slides
Appendix: Ideal Image Construction

Captured image

Ideal image

Horizontal cross-section of image

Pixel value

x

x 10^3

captured image
ideal image
Model Parameter Estimation Experiment

Take pictures of a light box at different positions in the field of view.

Canon EOS 350D: focal length 55mm, ISO100, f8.0
Olympus SP-510UZ: ISO 50, f8.0
Model parameter estimation

• **Construct** ideal images from these pictures

• Estimate parameters by minimizing the cost function

\[
\xi = \sum_{n=1}^{9} \sum_{(i,j) \in I_n} (\hat{y}_n(i,j) - y_n(i,j))^2
\]

\[
= \sum_{n=1}^{9} \sum_{(i,j) \in I_n} \left( \sum_{i_p} \sum_{j_p} x_n(i_p,j_p)\hat{s}(i,j;i_p,j_p) - y_n(i,j) \right)^2
\]

where \( I_n \) is the set of pixels where we compute the error at position \( n \).
What if we further increase $K_0$?