

Multivoxel pattern-based real-time fMRI

Stephen LaConte

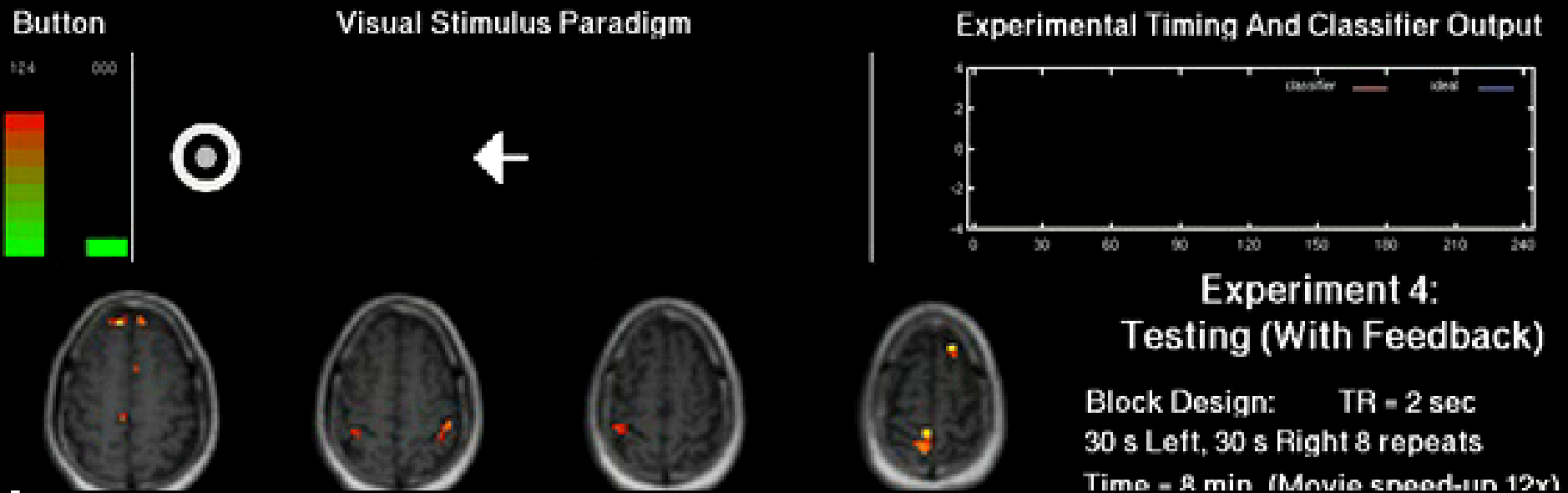
Fralin Biomedical Research Institute
Department of Biomedical Engineering and Mechanics



Multivoxel pattern-based real-time fMRI

- Conceptual overview
- Experimental flexibility
- Practicalities and additional resources

Classification in Real Time



LaConte et al. *Hum Brain Mapp* (2007)

Conceptual overview

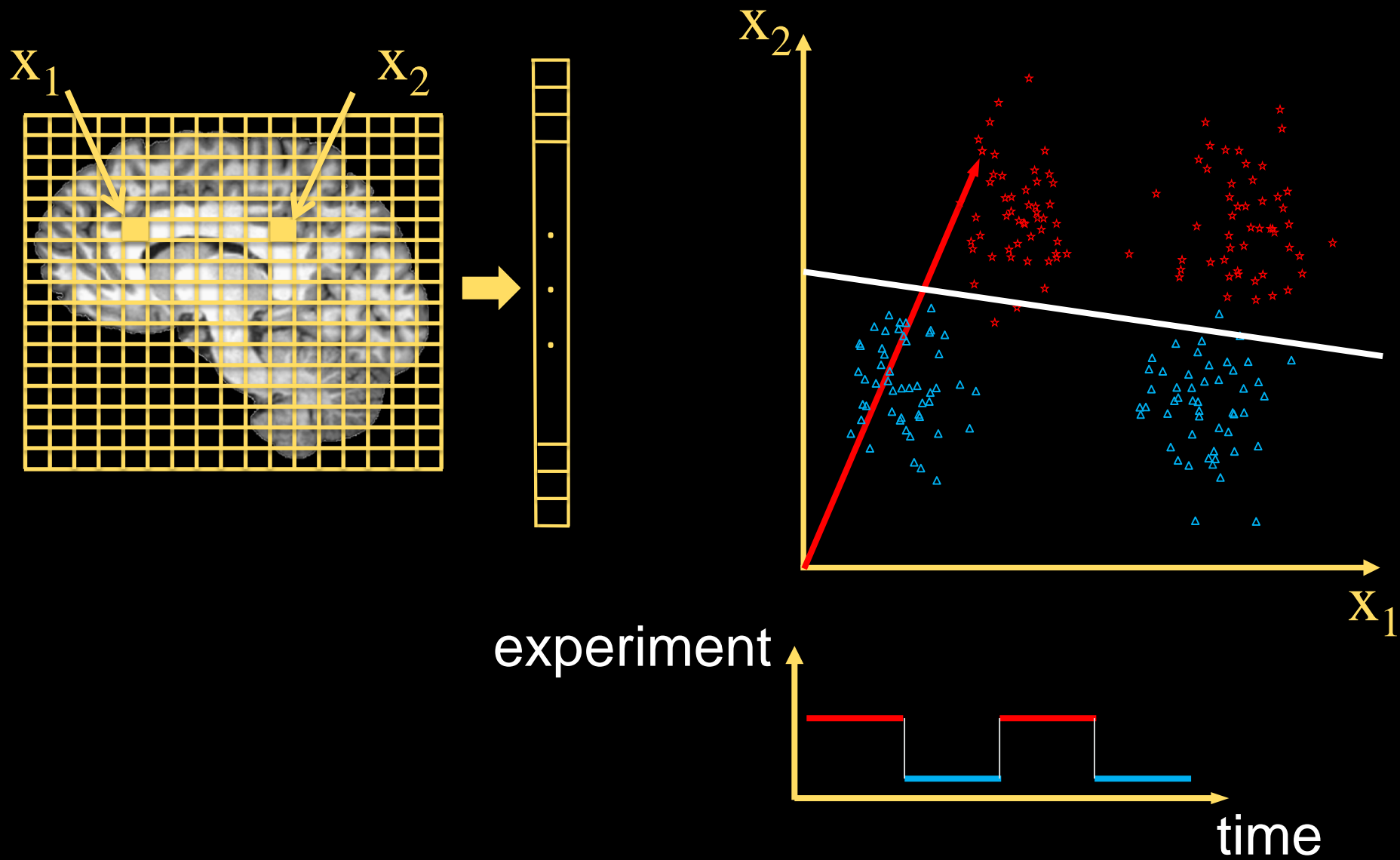
Supervised-learning rtfMRI

- Enables experimental flexibility
- Embodies “MVPA”
 - Complements region-based approaches
 - Enables brain-state decoding
 - Is computationally ideal for rtfMRI

Supervised learning

- **Complements univariate approaches**
(Friston, 1995; McIntosh, 1996; Strother, 2002; Moeller and Habeck 2006)
- **Early demonstrations**
(Lautrup, 1994; Dehaene, 1998)
- **Methodology and validation**
(Strother, 2002; LaConte, 2003; Mitchell 2004)
- **Representation of different classes of stimuli**
(Haxby, 2001; Cox and Savoy, 2003; Haynes & Rees, 2005; Kamitani & Tong 2005)
- **Detecting and tracking cognitive states**
(Polyn, 2005)
- **Natural representation for real-time fMRI**
(LaConte, 2007; Shibata, 2011; deBettencourt, 2015)

Brain State Classification



Classification

high dimensional
input space

x



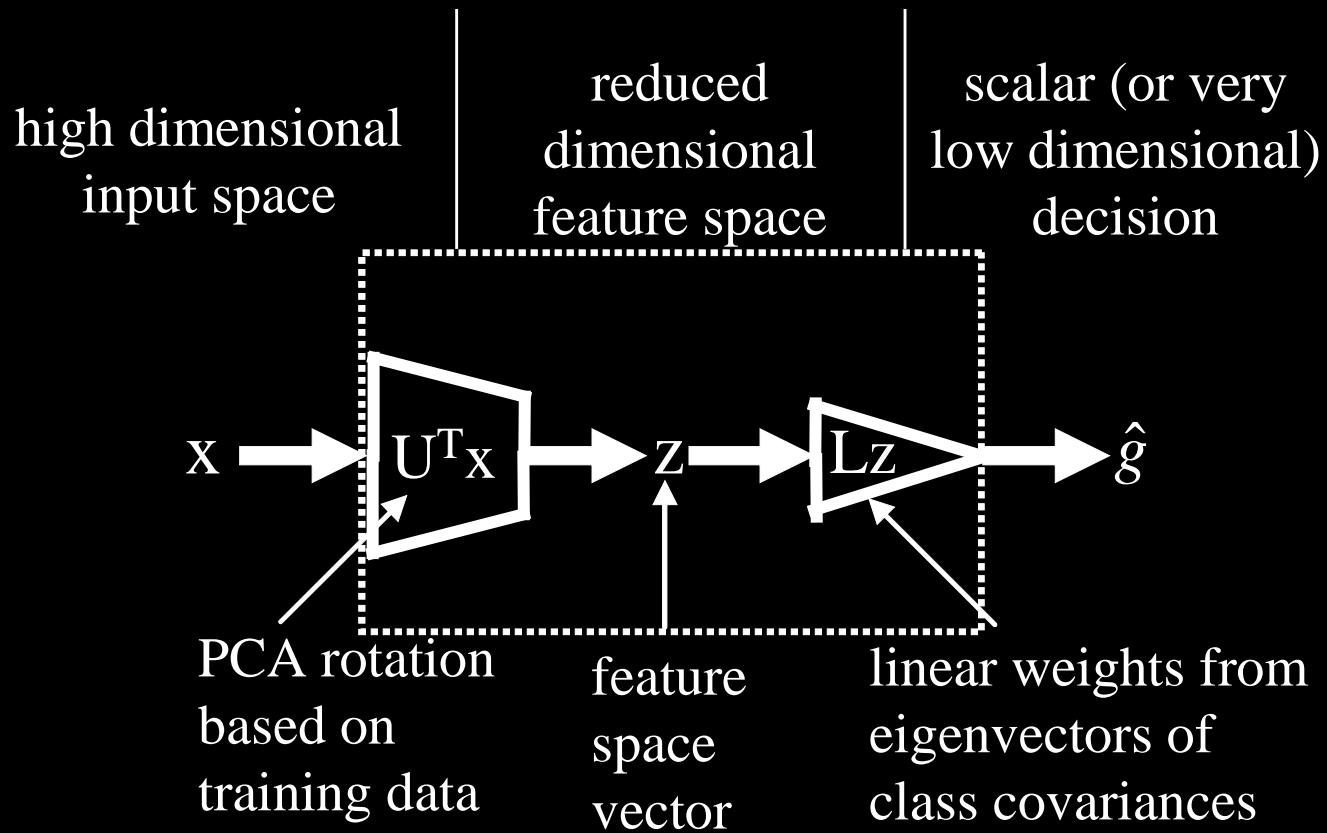
Classifier



\hat{g}

scalar (or very
low dimensional)
decision

Linear Discriminant Analysis



Linear Discriminant Analysis

Data Matrix:  \mathbf{X}

PCA via SVD:

$$\mathbf{U}^T \mathbf{X} = \mathbf{\Lambda} \mathbf{V}^T = \mathbf{Z}$$

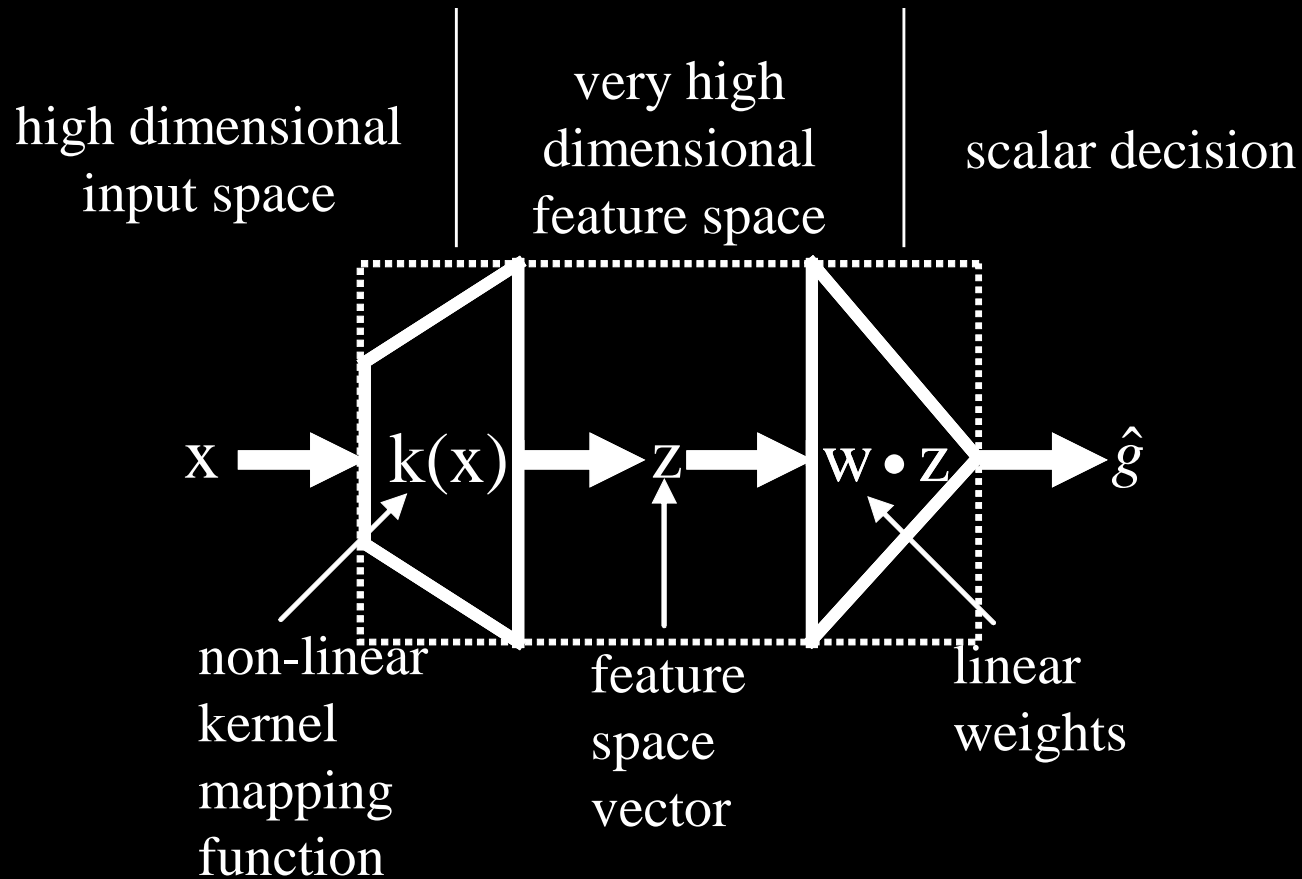
Truncate Q (model complexity)

CVA:

$$\mathbf{C} = \mathbf{LZ}^* = \mathbf{LU}^{T*} \mathbf{X}$$

Columns of \mathbf{L} are determined by the eigenvectors of $\mathbf{W}^{-1}\mathbf{B}$. \mathbf{W} is the within class variance and \mathbf{B} the between class variance, and both are obtained from \mathbf{Z} .

Support Vector Machine



SVM

$$D(\vec{z}_t) = (\vec{w} \cdot \vec{z}_t) + w_0$$

minimize

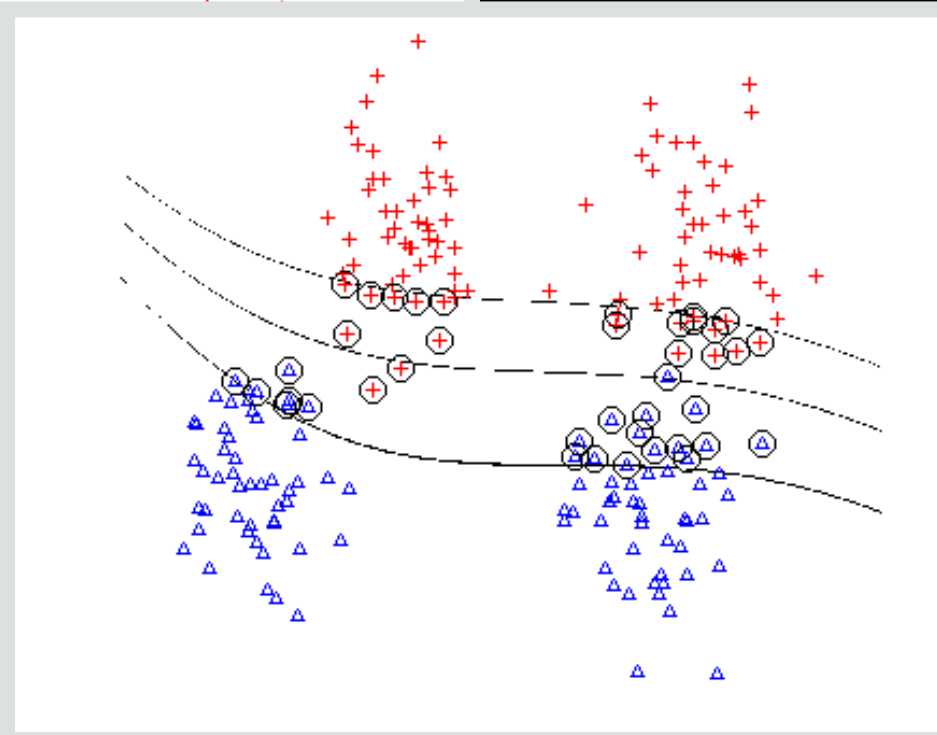
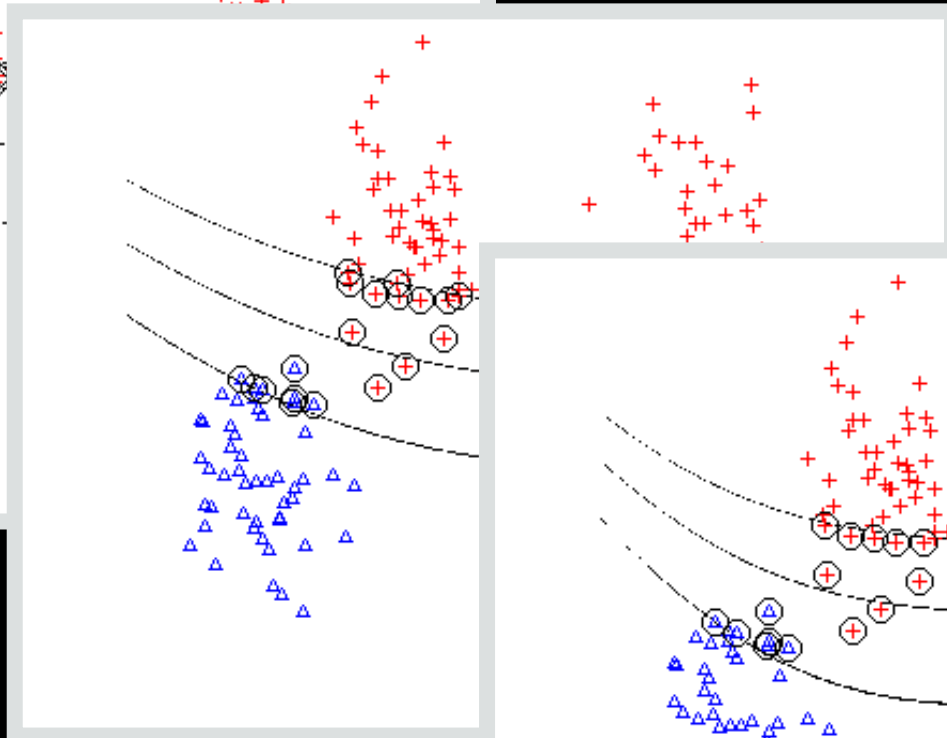
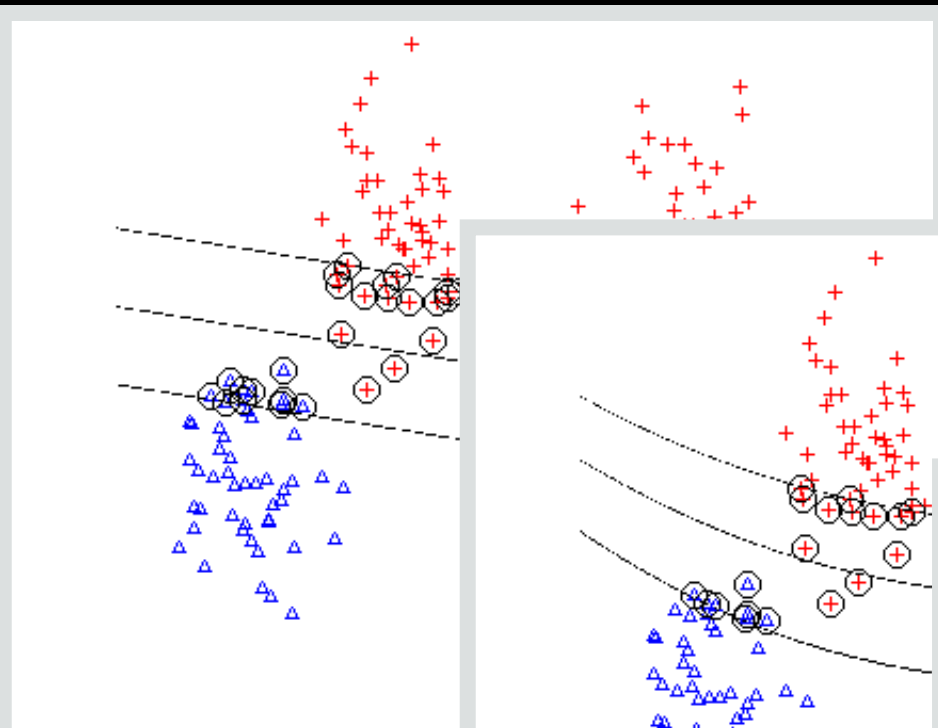
$$\frac{C}{T} \sum_{t=1}^T \xi_t + \frac{1}{2} \|\vec{w}\|^2$$

This term allows some training errors.

This term favors the widest possible margin,

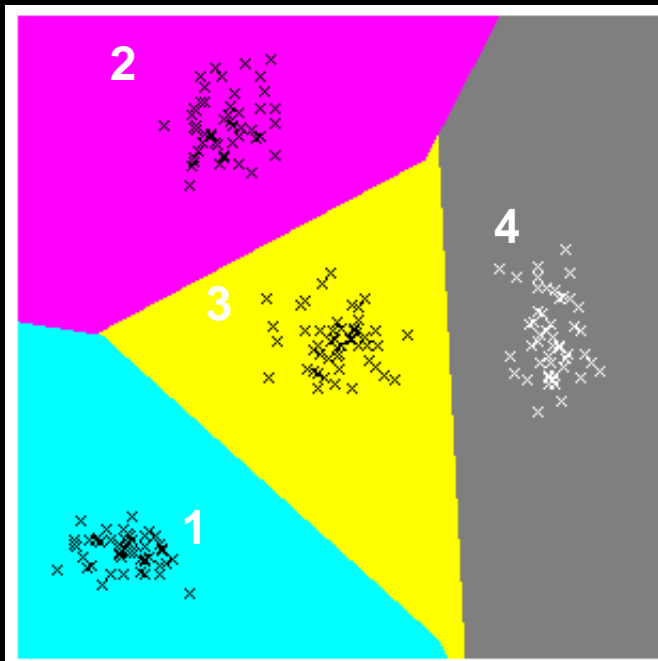
$C = \text{infinity}$ is hard margin SVM (as apposed to soft margin)
because it does not allow any training errors

Nonlinear Decision Boundary

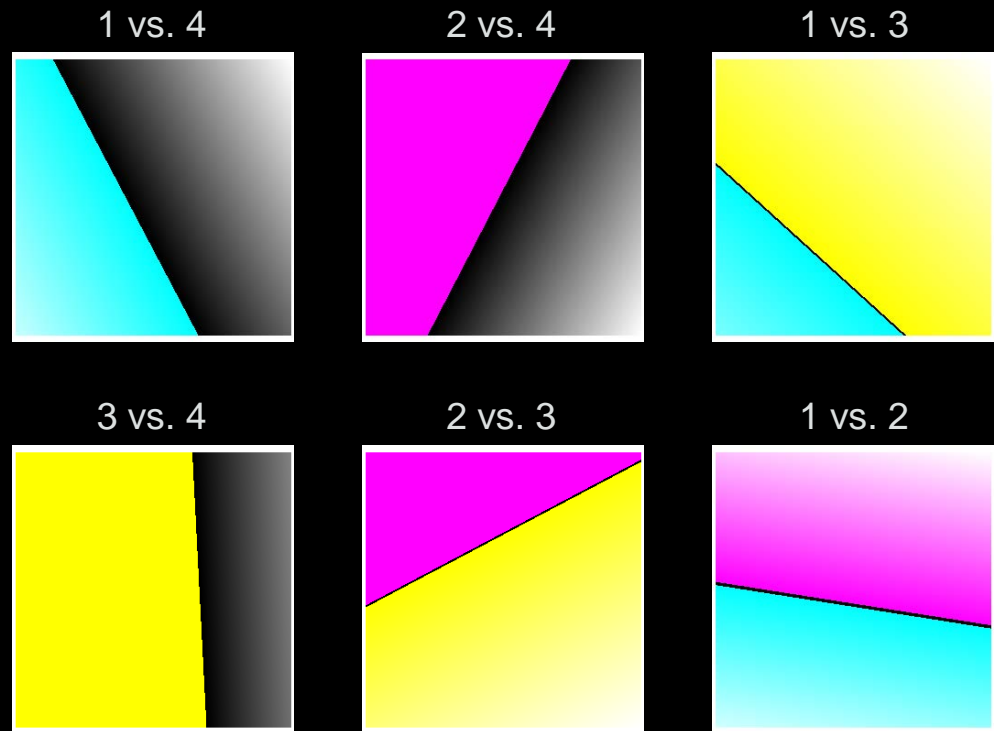


Multi-class

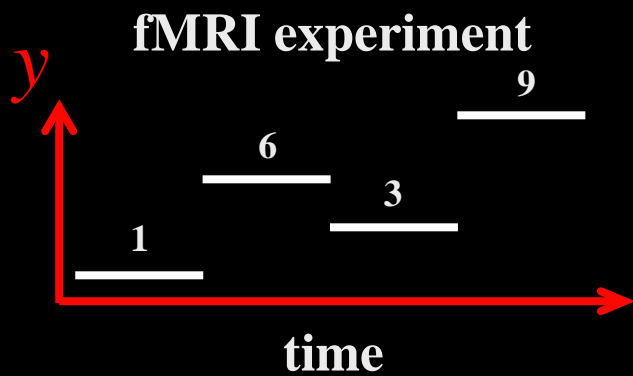
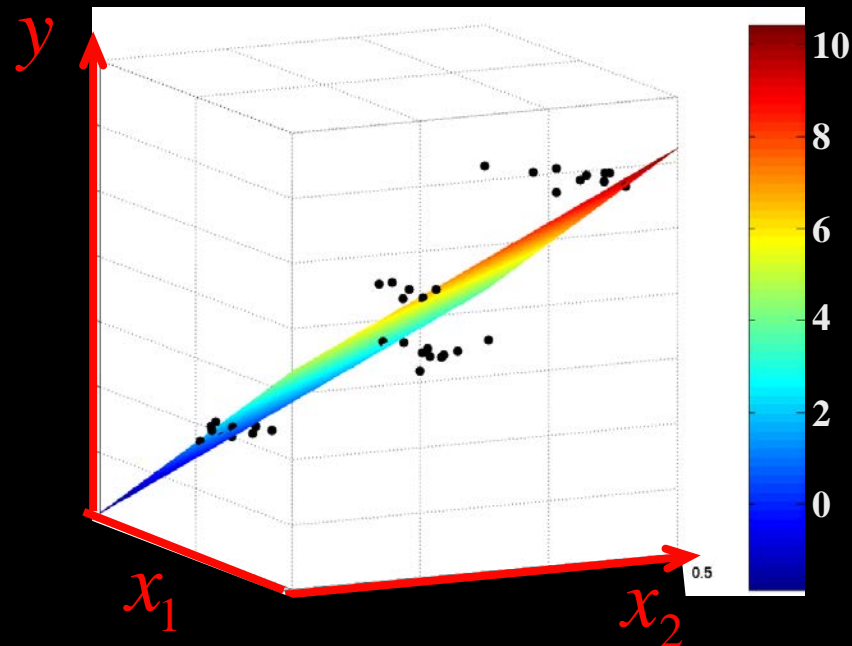
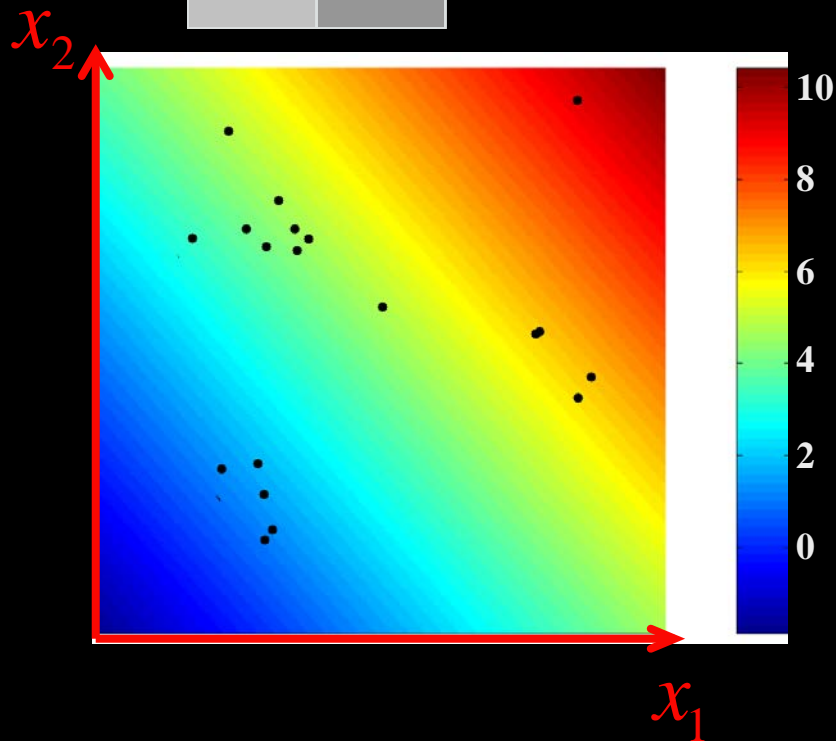
4-Class Model



Individual 2-class models



Temporal Regression

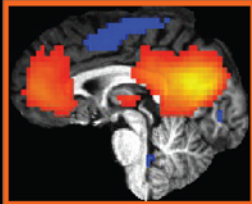


$$f(x) = x^T \beta + \beta_0$$

$$L(y, f(x))$$

Predicting Network Time Series

Network Template



Training Data



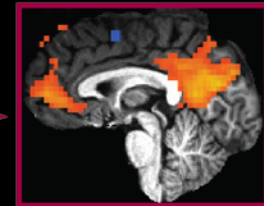
Spatial Regression

Network Time Series



SVR Training

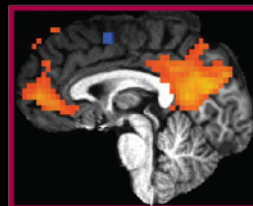
SVR Model



Testing Data



SVR Model

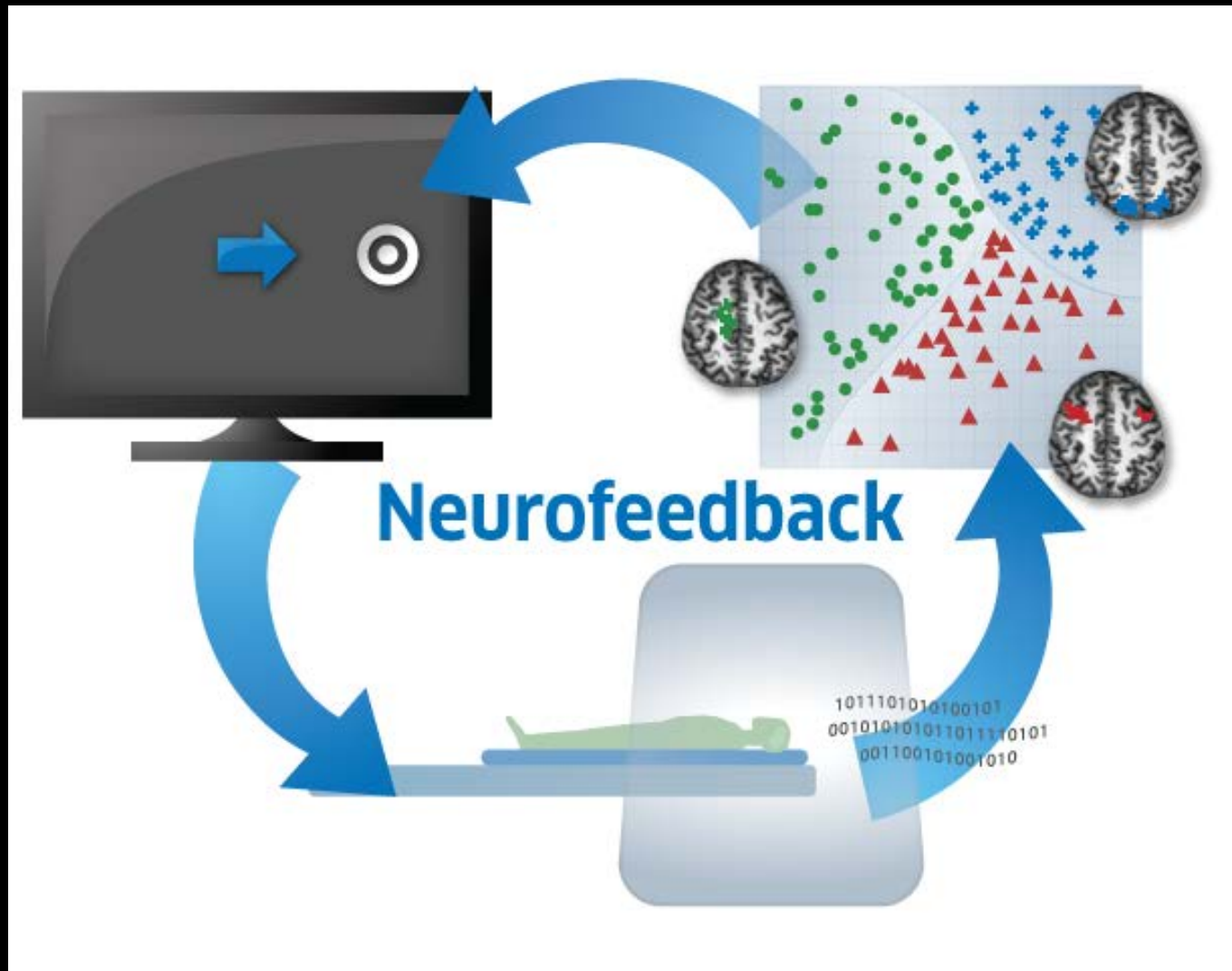


Network Time Series

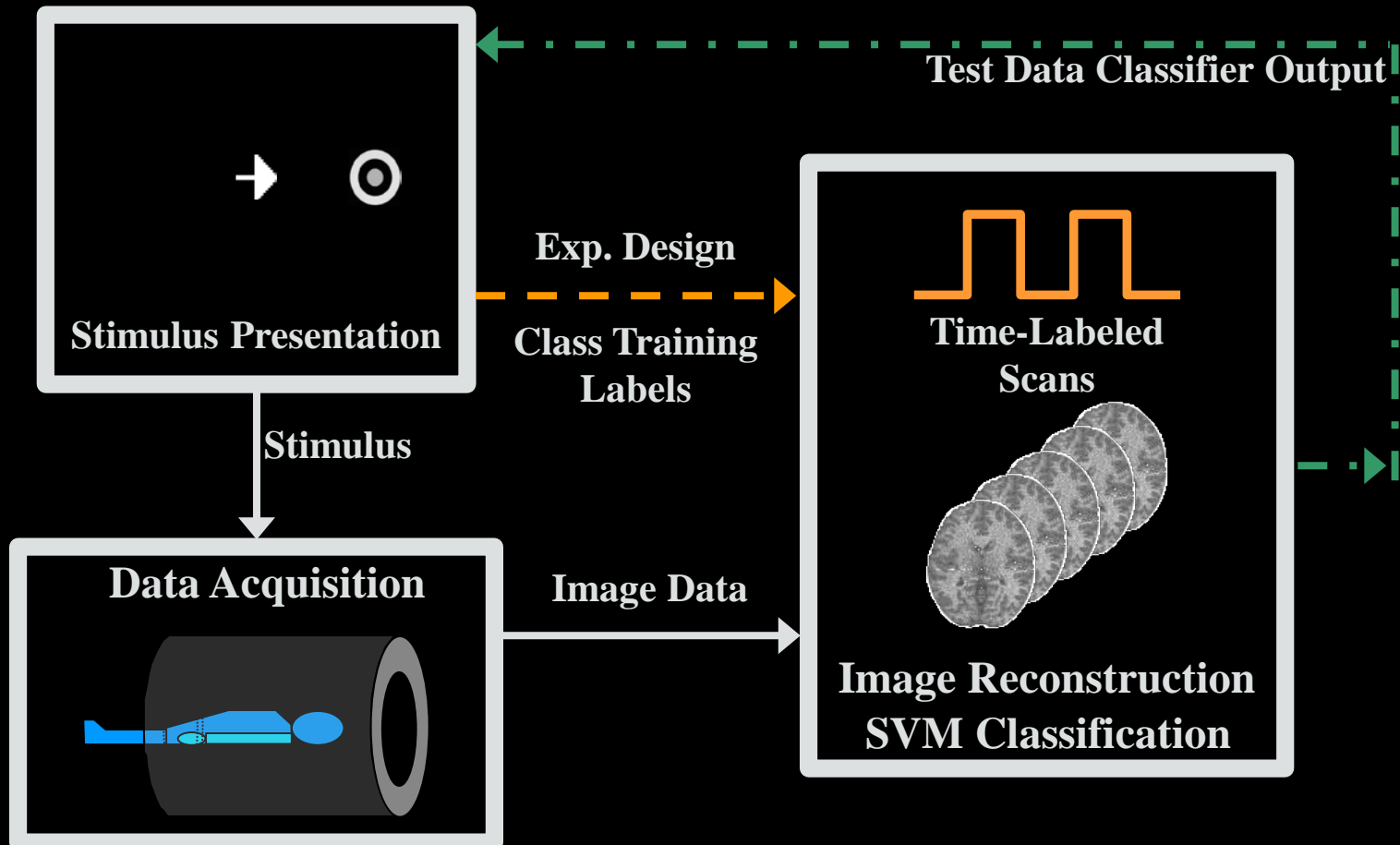


Craddock, R.C. et al. OHBM 2012.

Pattern-based rtfMRI

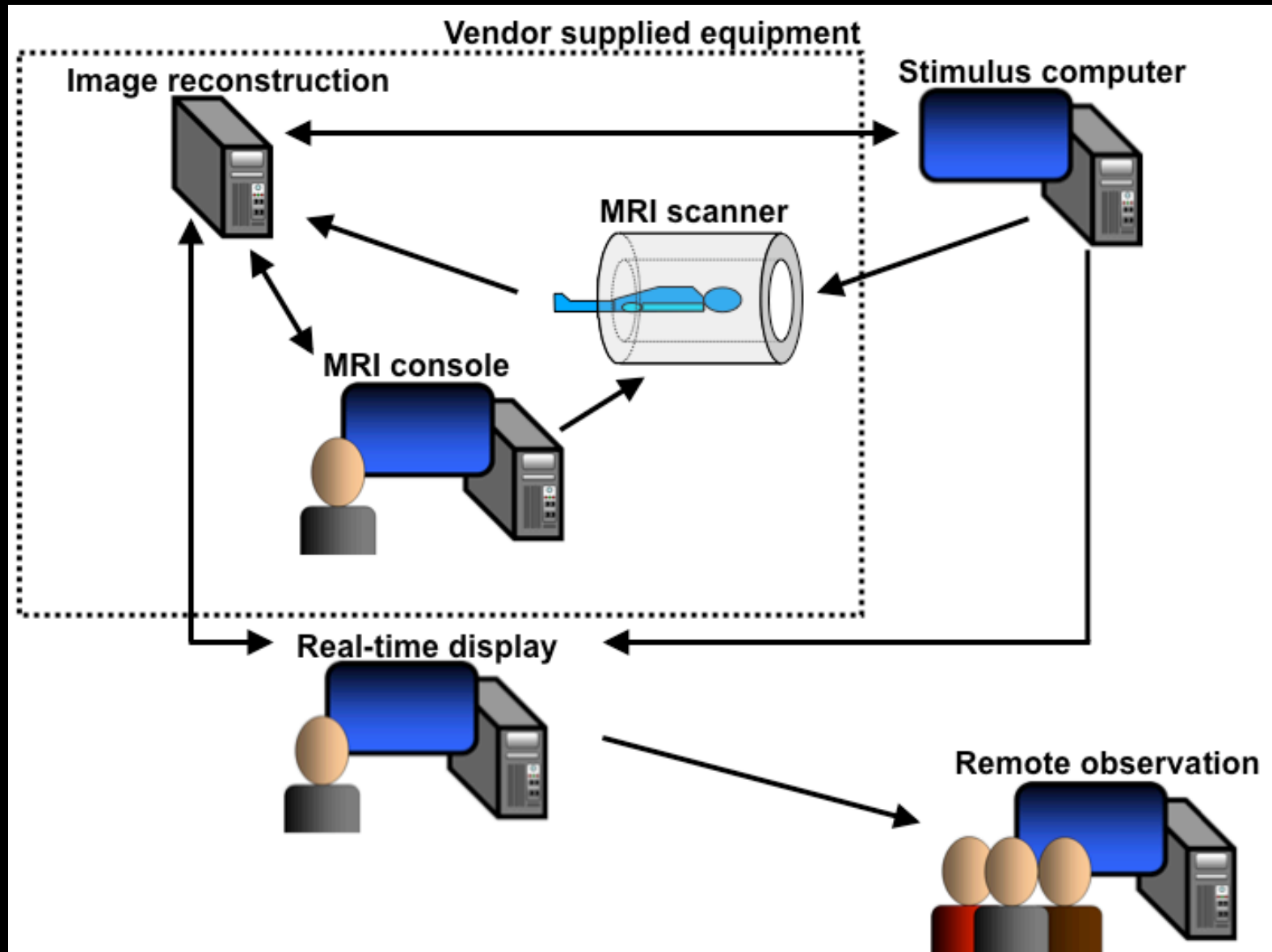


Pattern-based rtfMRI

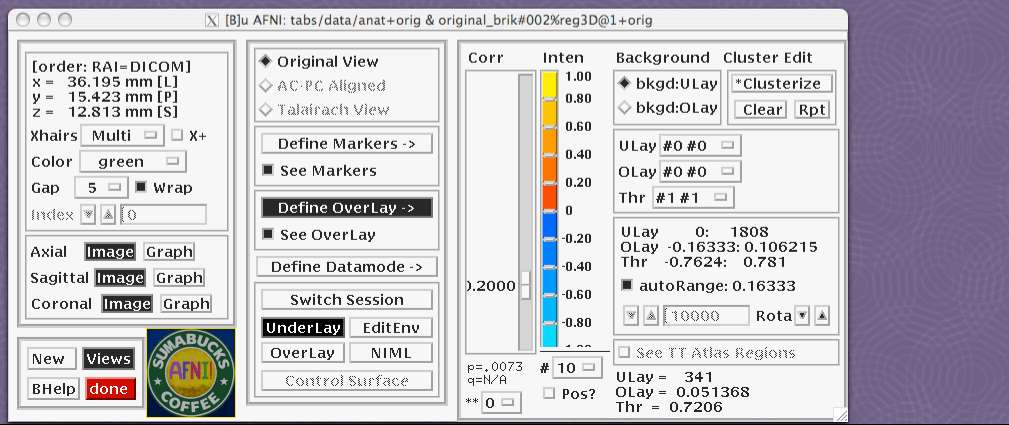
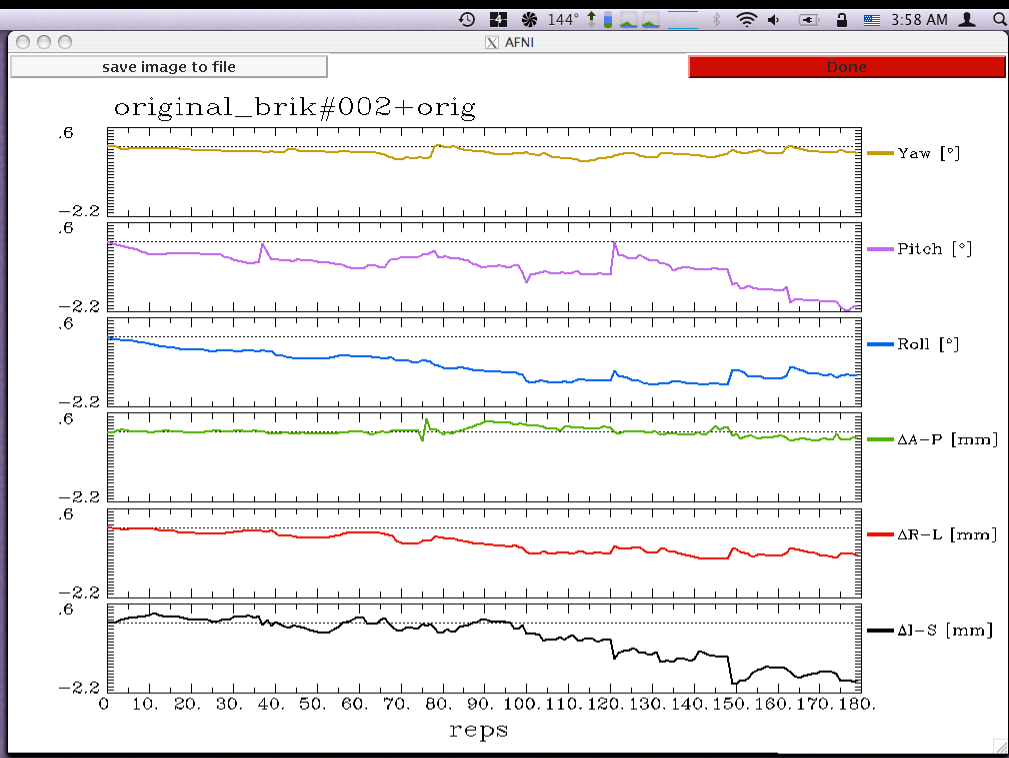
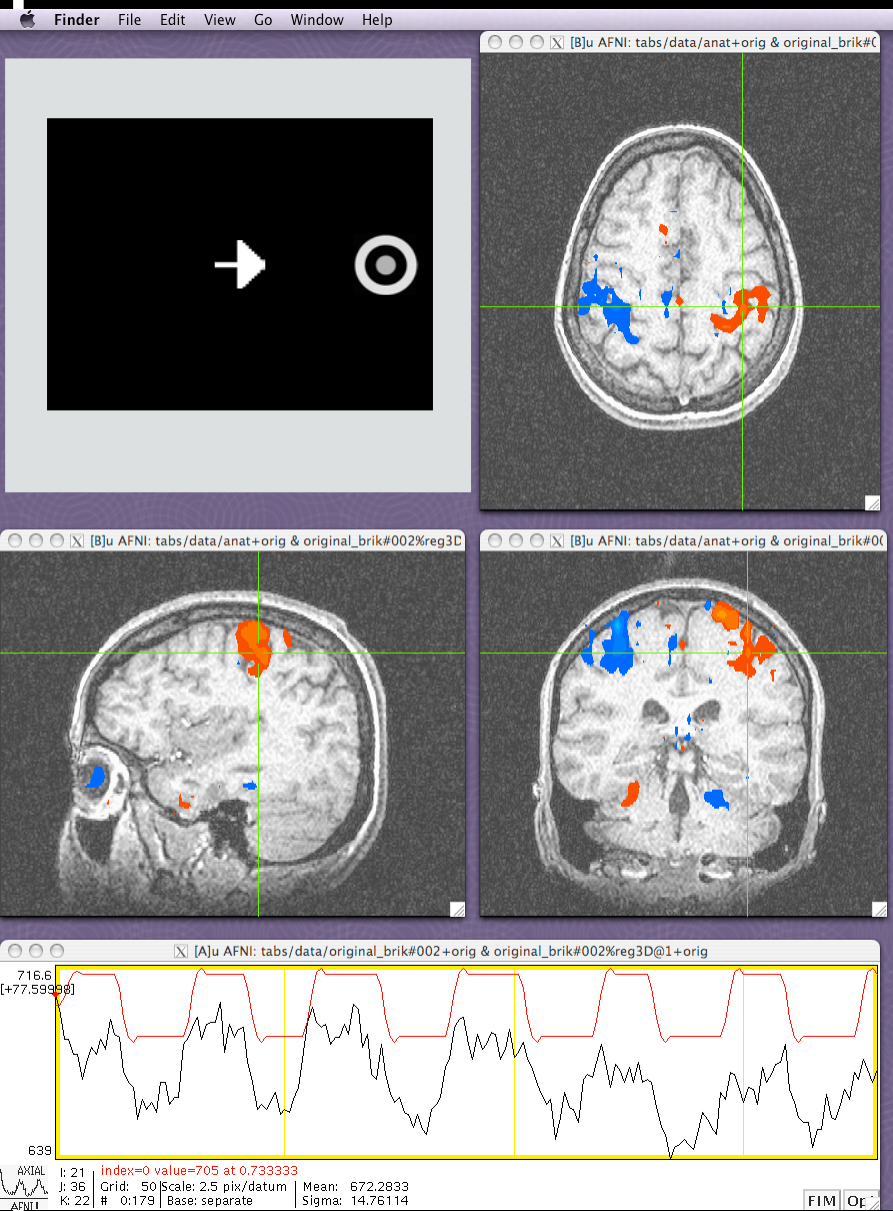


— Conventional fMRI - - Training run - · - Testing Run

Network configuration



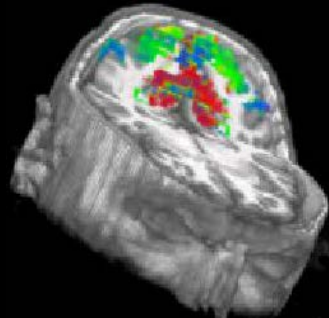
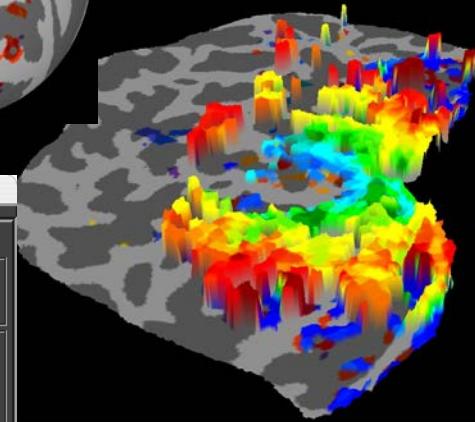
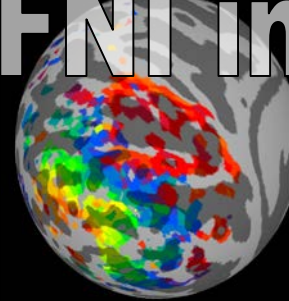
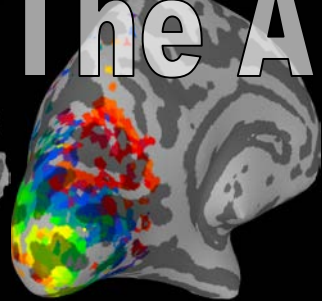
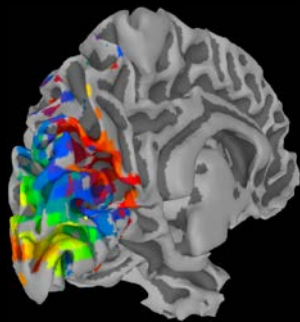
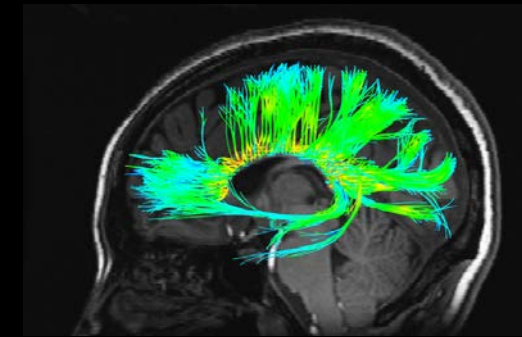
Stimulus seen by volunteer Updated fMRI results Motion tracking and correction



Intensity (brightness) of a single pixel, changing during stimulus conditions

Controller interface for some display parameters

The AFNI interface



[A] AFNI: suma_demo/afni/DemoSubj_SurfVol_AInld_Exp+orig & DemoSubj_EccExpavir.DEL+orig

[Order: RAI=DICOM]
x = 0.500 mm [L]
y = 83.500 mm [P]
z = -0.500 mm [I]

Xhairs Multi X+
Color black
Gap 5 Wrap
Index

Axial Image Graph
Sagittal Image Graph
Coronal Image Graph

New Views
BHelp done

Original View
AC-PC Aligned
Talairach View

Define Markers
 See Markers

Define Overlay
 See Overlay

Define Datamode
Switch Session
Switch UnderLay

Corr Inten Options

Ulay underlay
OLay underlay

Ulay #0 #0
OLay #0 Delay
Thr #2 Corr Conf

Ulay
OLay
Thr

1614
[+170]

+++++ nearby Atlas structures +++++
Focus point (LPI)=
-3 mm [L], -80 mm [P], 12 mm [S] {T-T Atlas}
-3 mm [L], -83 mm [P], 9 mm [S] {MNI Brain}
-3 mm [L], -88 mm [P], 20 mm [S] {MNI Anat.}

Atlas TI_Daemon: Talairach-Tournoux Atlas
Focus point: Left Cuneus
-AND- Left Brodmann area 17
Within 2 mm: Left Brodmann area 18
Within 5 mm: Left Brodmann area 23
Within 7 mm: Left Lingual Gyrus
-AND- Left Brodmann area 30

Atlas CA_N27_MPM: Cytoarch. Max. Prob. Maps (N27)
Focus point: HIP1
Within 3 mm: Awvg. (SF)

Atlas CA_N27_ML: Macro Labels (N27)
Focus point: Left Calcarine Gyrus
Within 1 mm: Left Cuneus
Within 7 mm: Right Cuneus

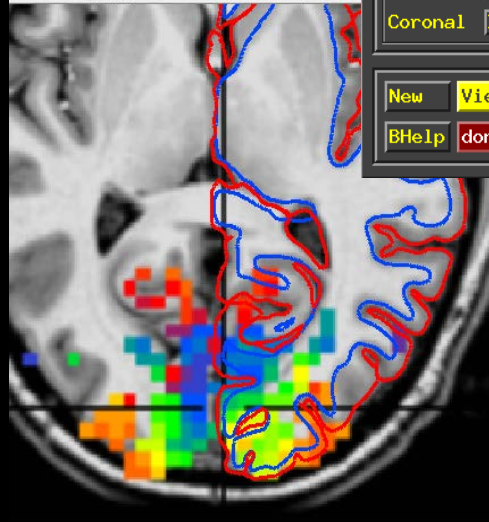
Atlas CA_N27_PM: Cytoarch. Probabilistic Maps (N27)
Focus point: Area 17 (p = 0.50)
-AND- Area 18 (p = 0.60)

Atlas CA_N27_LR: Left/Right (N27)
Focus point: Left Brain
Within 6 mm: Right Brain

127
left=Right byte=0..252 ent=5,18

Disp Sav1.ppm Mont Done Rec

[A] AFNI: suma_demo/afni/DemoSubj



127
left=Right byte=0..252 ent=5,18
Disp Sav1.ppm Mont Done Rec

[B] AFNI: suma_demo/afni/DemoSubj_EccExpavir+orig & DemoSubj_EccExp

1614
[+170]

1444

AXIAL X: 31 index=0 value=1552 at 1.411765
Y: 31 Grid: 20 Scale: 1 pix/datum Mean: 1497.313
Z: 7 # 0.133 Base: separate Sigma: 24.08449

AFNI FIM Op

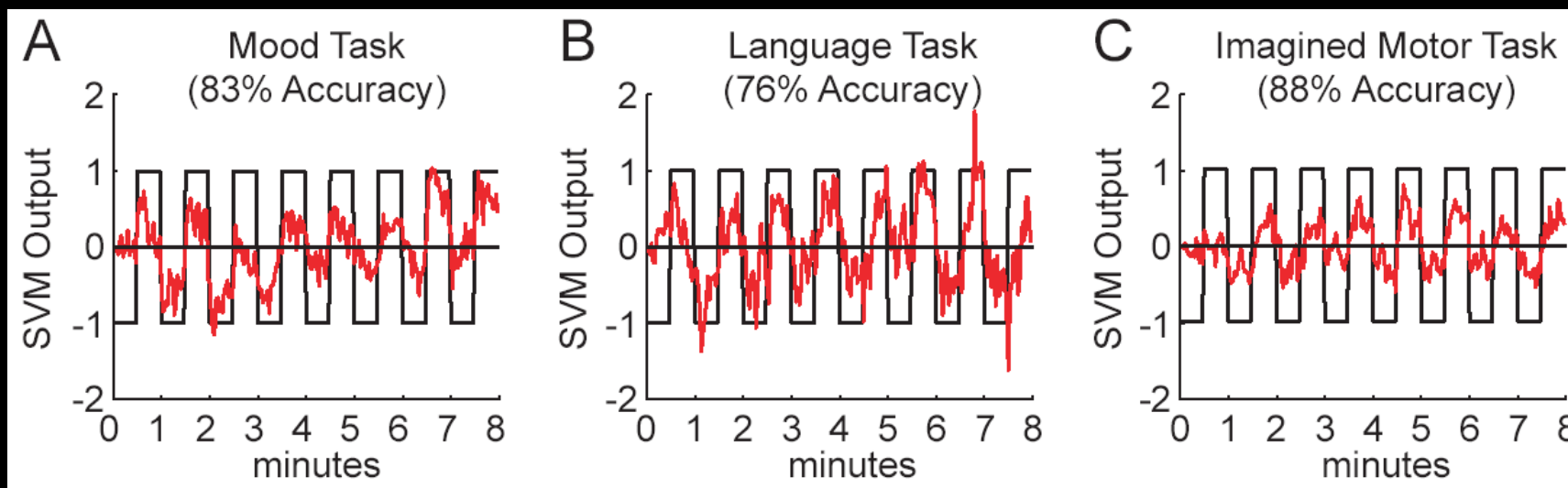
Multivoxel pattern-based real-time fMRI

- Conceptual overview
- Experimental flexibility
- Practicalities and additional resources

Flexibility of brain state classification



With the exact same experimental setup (different instructions), subjects can learn to move the arrow



LaConte, et al. (2007) Hum Brain Mapp

Experimental flexibility: Task appropriate interfaces

Cigarette craving



Covert counting rate



Experimental flexibility: Support vector regression of RSNs

neurofeedback

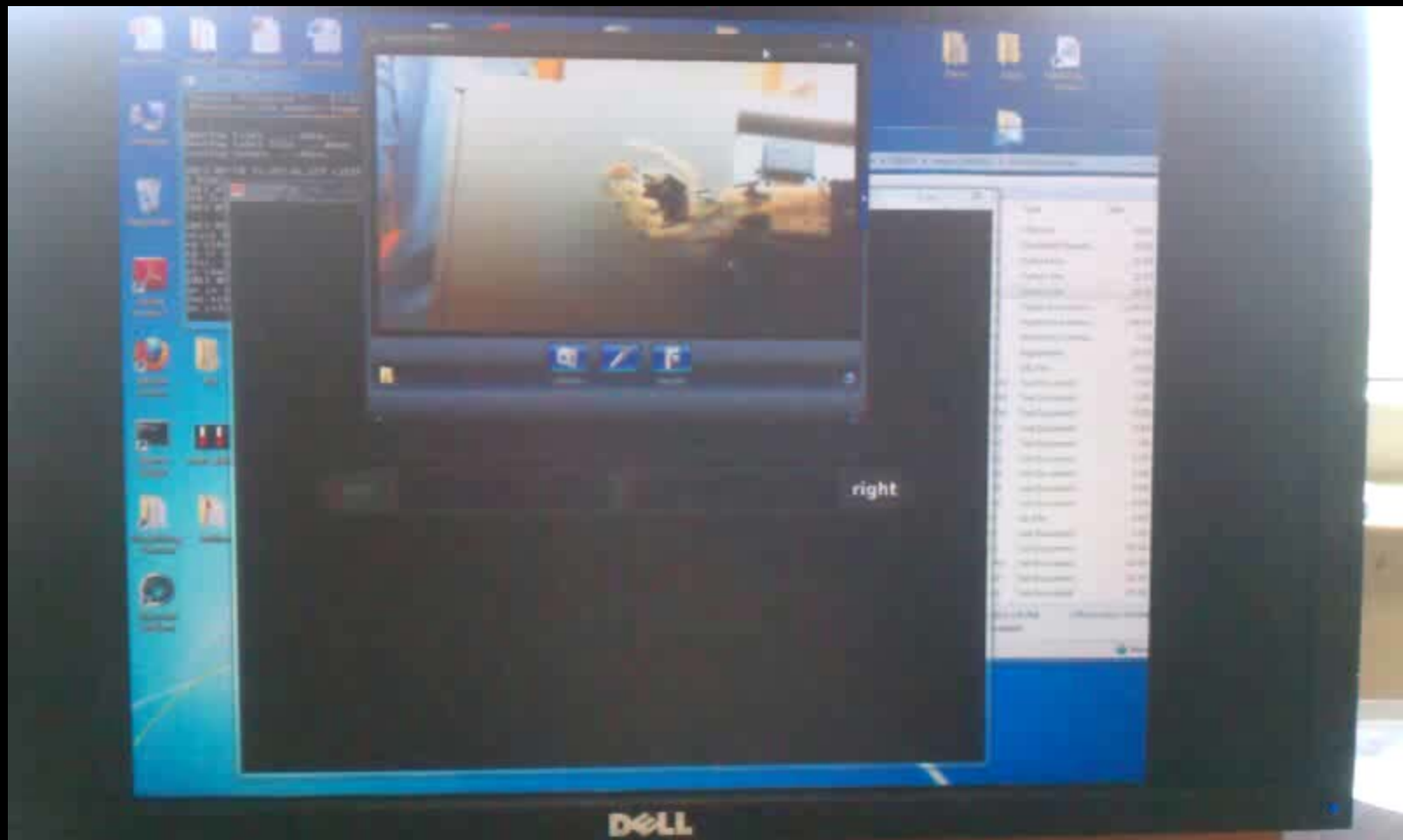
controlling events



8

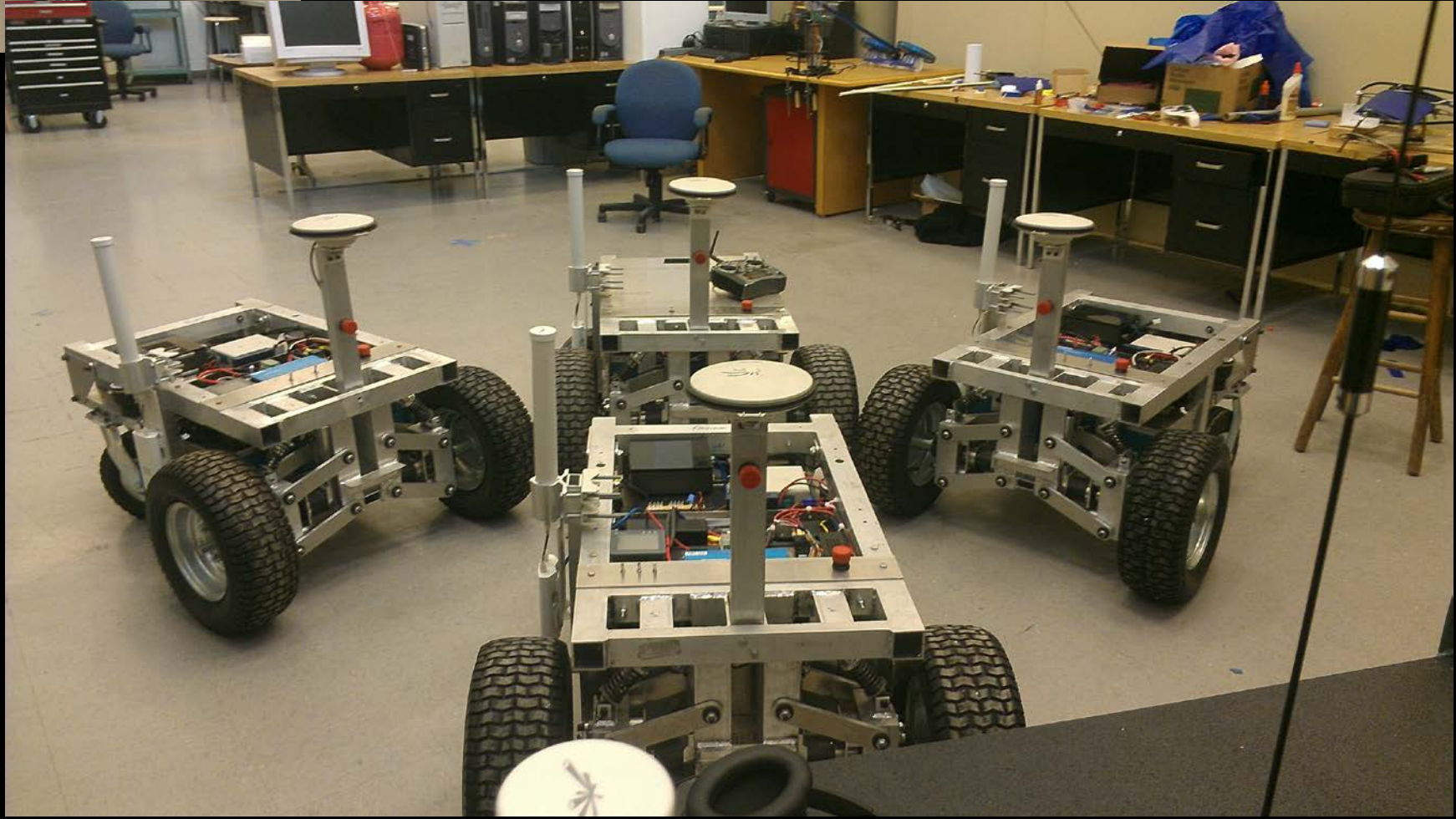


Experimental flexibility: Brain Machine Interfaces

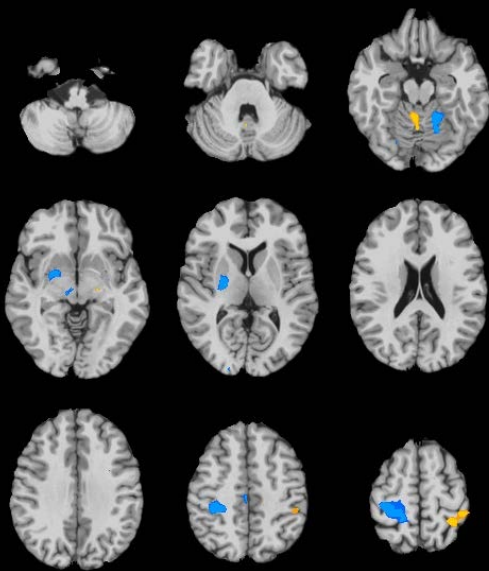


with Shashank Priya and Read Montague

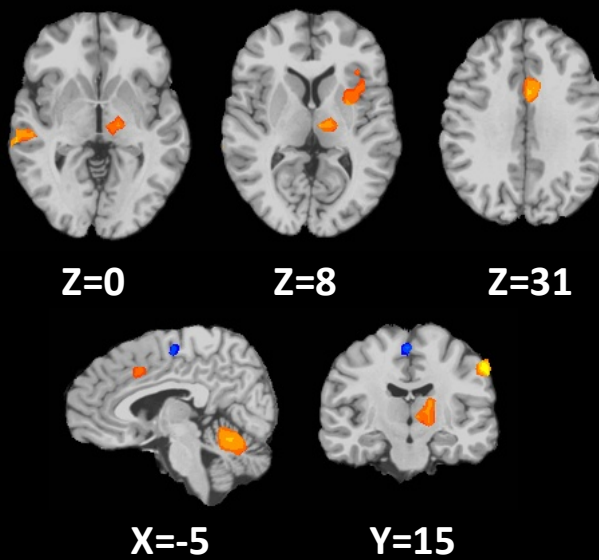
Experimental flexibility: Brain Machine Interfaces



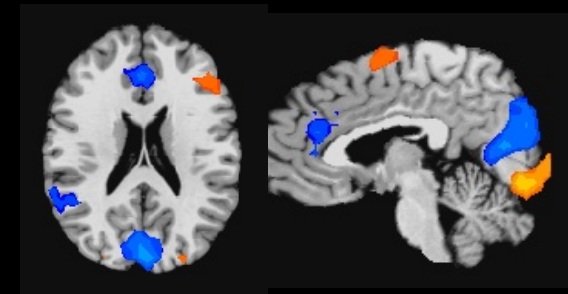
Support Vector Machine Maps of Real-Time Tasks



Right vs. Left Tapping

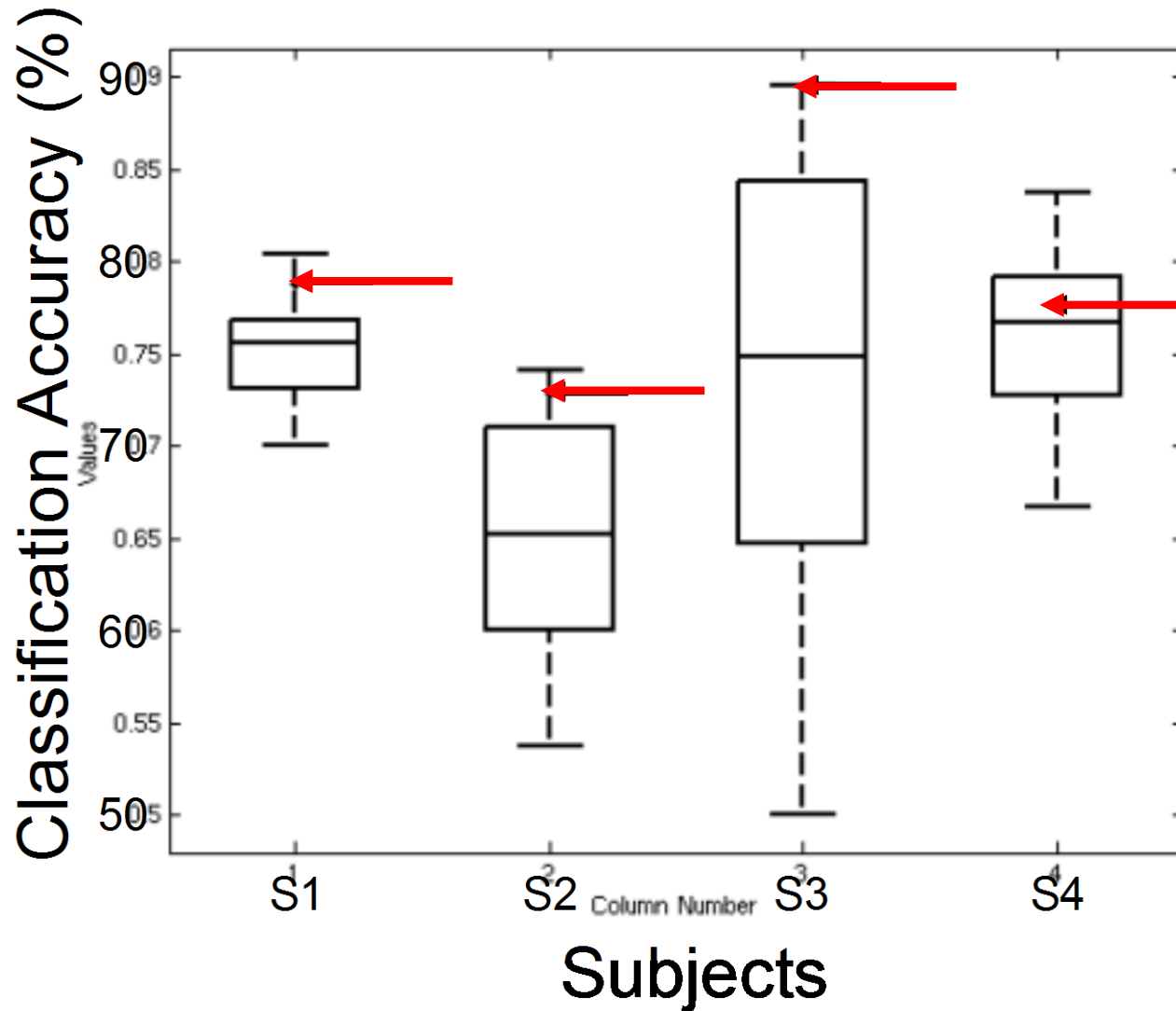


Fast vs. Slow Counting



Crave vs. Don't Crave

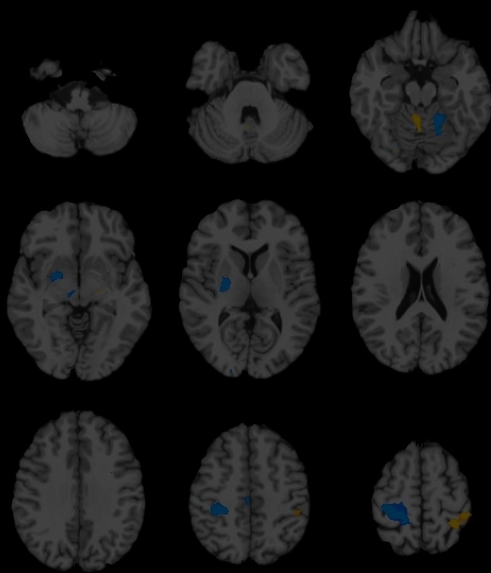
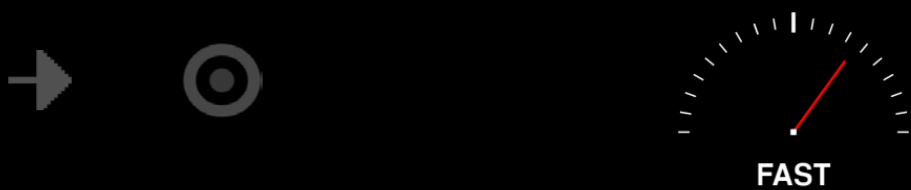
Do feedback runs differ from no-feedback runs?



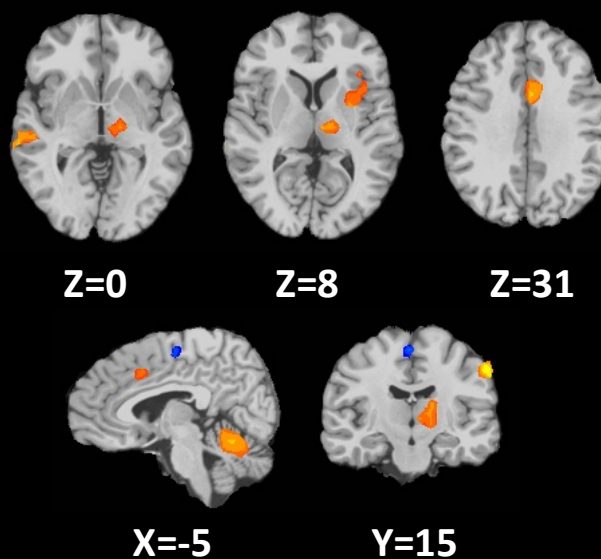
¹We h

(, 2007)

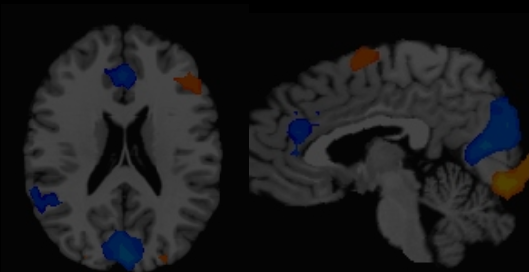
Support Vector Machine Maps of Real-Time Tasks



Right vs. Left Tapping

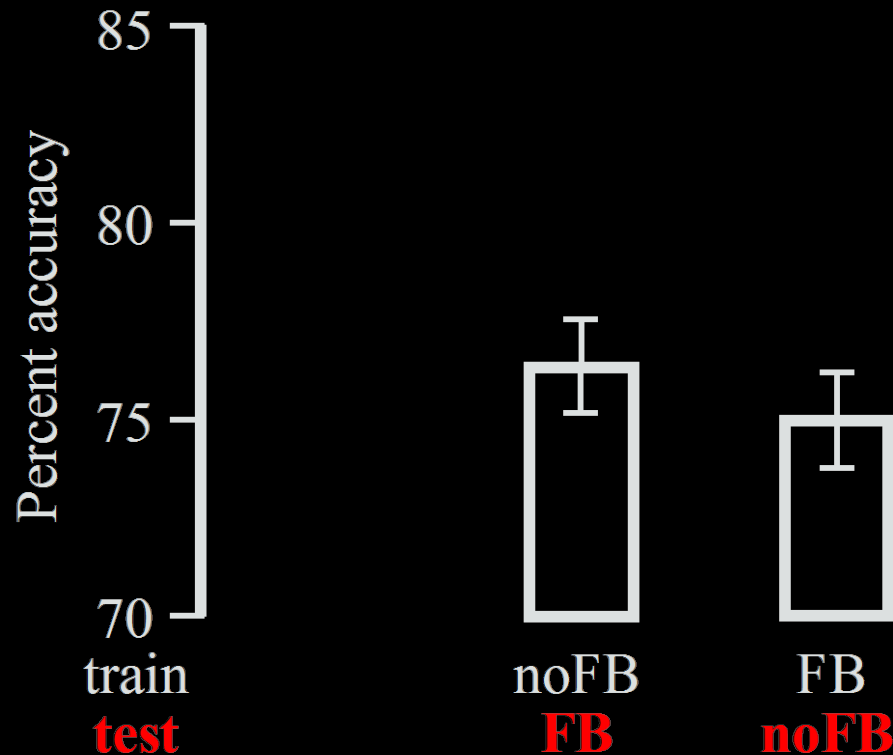


Fast vs. Slow Counting

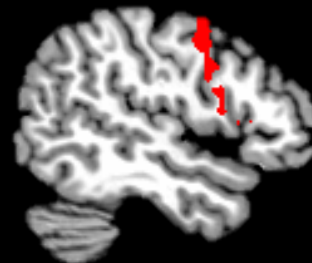
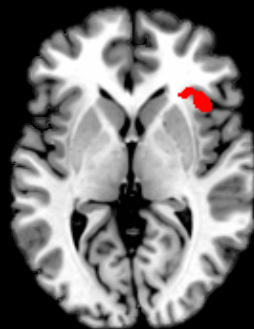
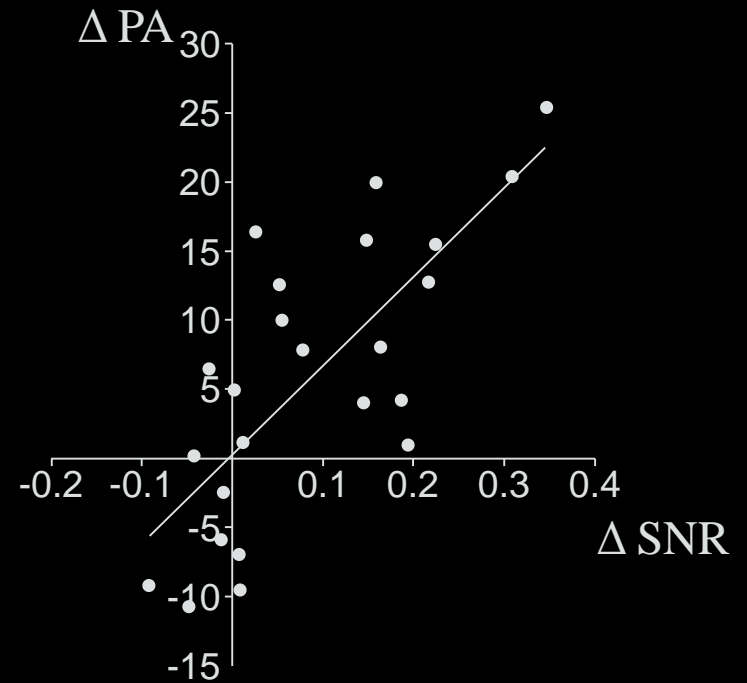
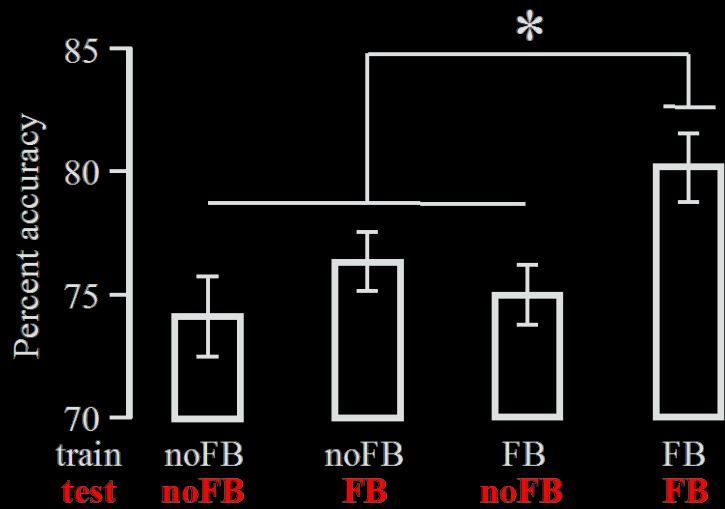


Crave vs. Don't Crave

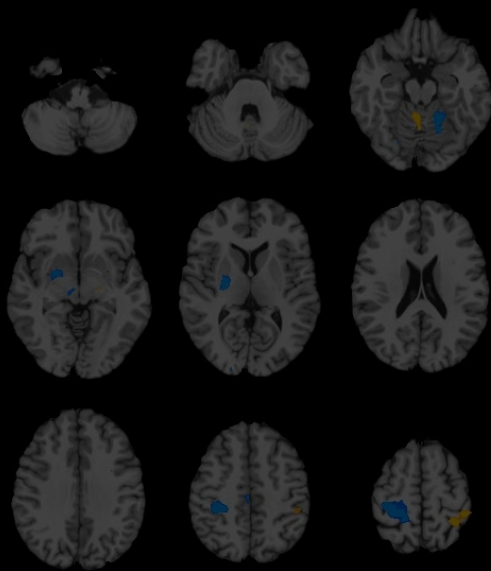
Speech: Covert counting Classification accuracy



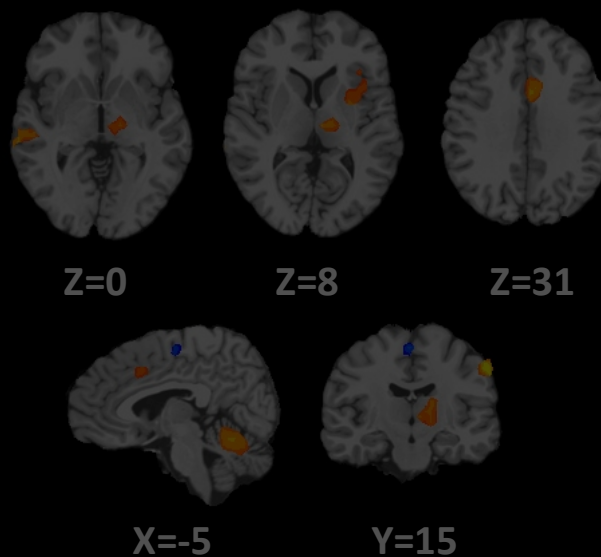
Speech: Covert counting



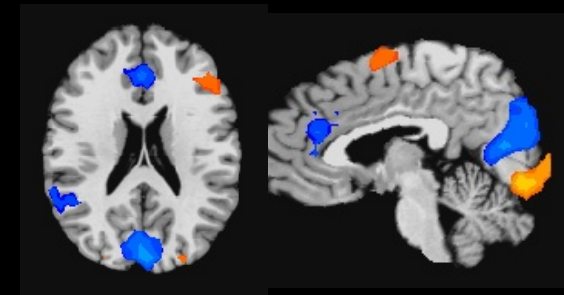
Support Vector Machine Maps of Real-Time Tasks



Right vs. Left Tapping

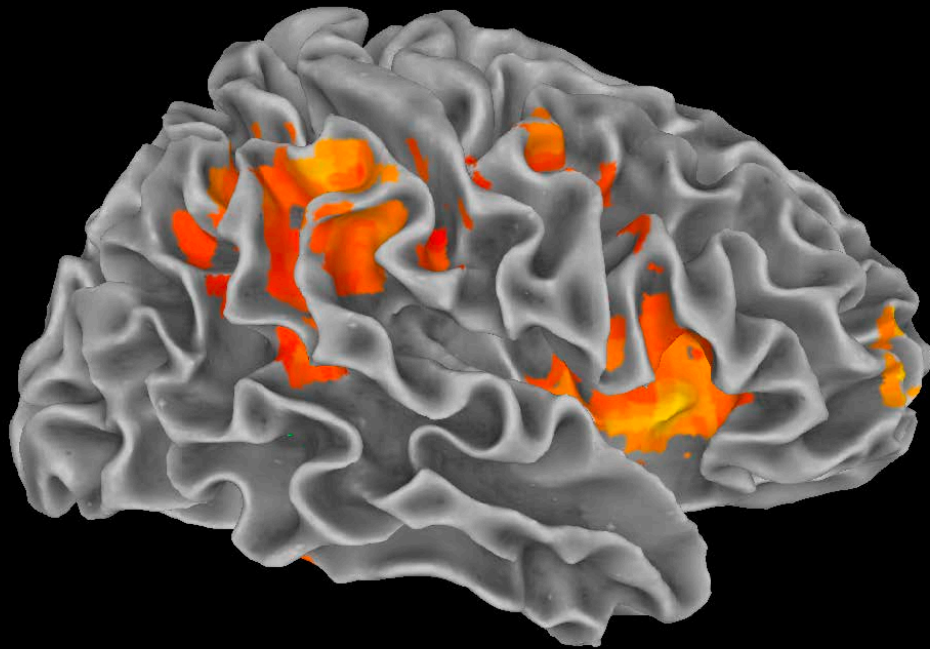


Fast vs. Slow Counting

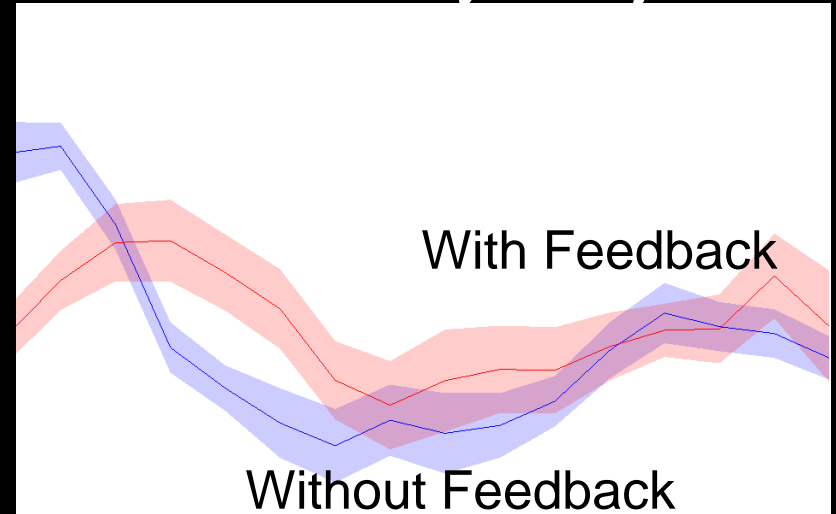


Crave vs. Don't Crave

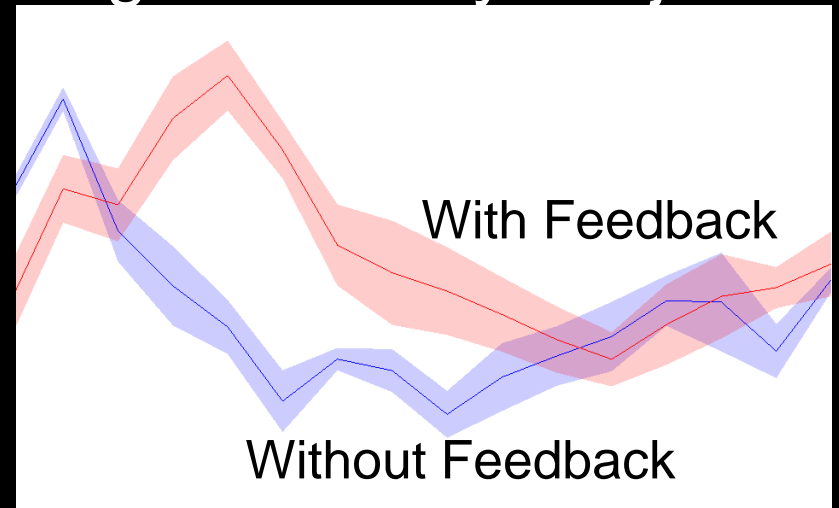
Frontoparietal



Low accuracy subjects



High accuracy subjects



Multivoxel pattern-based real-time fMRI

- Conceptual overview
- Experimental flexibility
- Practicalities and additional resources

Computational considerations

First real-time fMRI (Cox et al., 1995), developed a recursive partial correlation algorithm.

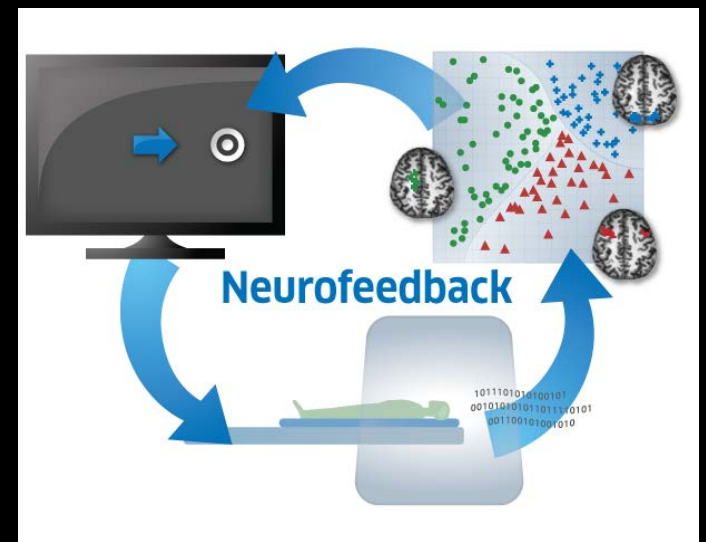
$$\rho = \frac{(\mathbf{P}\mathbf{r})^T(\mathbf{P}\mathbf{x})}{|\mathbf{P}\mathbf{r}| \cdot |\mathbf{P}\mathbf{x}|} = \frac{\mathbf{r}^T\mathbf{P}\mathbf{x}}{(\mathbf{r}^T\mathbf{P}\mathbf{r} \cdot \mathbf{x}^T\mathbf{P}\mathbf{x})^{1/2}} \quad \alpha = \frac{\mathbf{r}^T\mathbf{P}\mathbf{x}}{\mathbf{r}^T\mathbf{P}\mathbf{r}} \quad [2]$$

Equation [2] is not well suited for real-time fMRI...As the number of images grows (i.e., as the vectors increase in dimension), the amount of calculation will grow. In a real-time application, this is unacceptable, because at some point the computer will not be able to finish processing a new image before the next one is ready.

Computational considerations

- Real-time classification
 - Train with a deterministic algorithm
 - Training is computationally intensive but highly doable
 - Converge “immediately” after scan
 - Classify on an image-by-image basis

Basic Benchmarks



- Classifier safety factor $> 20,000x$.
 - Approximately $1\mu\text{sec}$ / dot product.
- Network/AFNI Transfers $> 20x$ volumes
 - Approximately $100\mu\text{sec}$ / slice to transfer

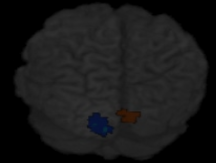
3dsvm Plugin Screenshot

Support Vector Machine Analysis

The screenshot displays the 'AFNI Plugin: Set Real-Time Options for 3dsvm - An AFNI SVM-Light Plugin' window. The interface is organized into a sidebar on the left and a main configuration area on the right. The sidebar contains several sections, each with a checkbox: 'Real-time' (checked), 'Training', 'Train Data', 'Train Params', 'Kernel', 'Model Output', 'Model Inspection', 'Testing', 'Test Data', 'TCP/IP', and 'Predictions'. The main area features a title bar, three buttons ('Run+Keep', 'Run+Close', 'Help'), and a grid of parameter settings. The 'Real-time' section is currently active, showing a 'Type' dropdown set to 'classification'. Other parameters include 'Labels' and 'Censors' (both set to '-Choose Timeseries-'), 'Mask' (set to 'C'), 'Kernel' (set to 'linear'), 'poly order (d)' (set to 3), 'Epsilon' (set to 0.001), 'rbf gamma (g)' (set to 1), 'Prefix', 'FIM Prefix', 'Alpha Prefix (.ID)', 'Model' (set to '-Choose Dataset-'), 'IP', 'PORT', and 'Prefix (.ID)'.

Section	Parameter	Value
Real-time	Type	classification
	Labels	-Choose Timeseries-
	Censors	-Choose Timeseries-
	Mask	C
	Kernel	linear
	poly order (d)	3
	Epsilon	0.001
	rbf gamma (g)	1
	Prefix	
	FIM Prefix	
Alpha Prefix (.ID)		
Testing	Model	-Choose Dataset-
	IP	
TCP/IP	PORT	
	Prefix (.ID)	

3dsvm



LaConte Lab

[Home](#) | [Research](#) | [Publications](#) | [Facility](#) | [People](#) | [Links](#)
[Slides](#) | [3dsvm](#)

3dsvm - an SVM-Light plugin for AFNI

[description](#) | [interactive screen shot](#) | [download info](#) | [data and use](#)
[todo](#) | [developers](#) | [reference](#)

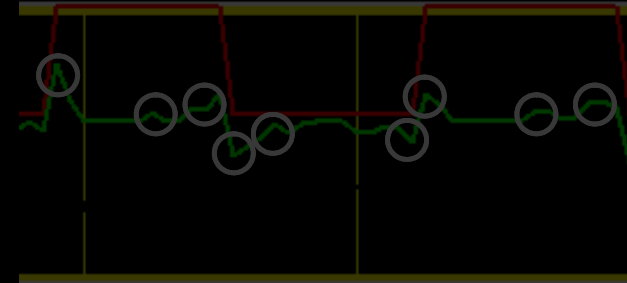
description

3dsvm is a command line program and plugin for AFNI, built around SVM-Light. It provides the ability to analyze functional magnetic resonance imaging (fMRI) data as described in (LaConte et al., 2005).

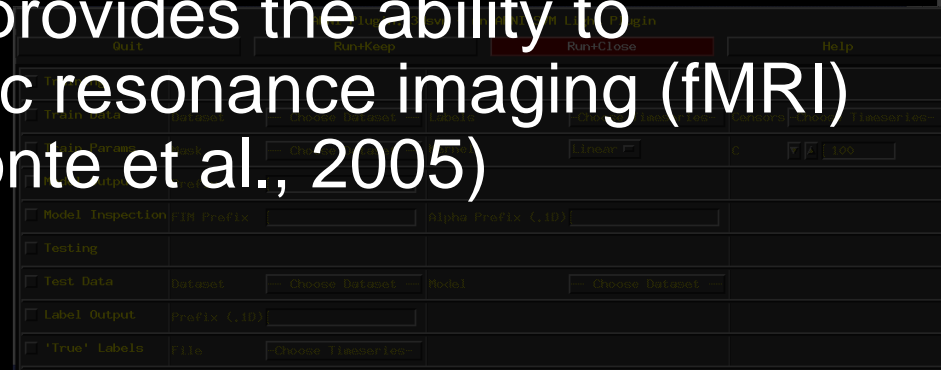
Features:

- Reading AFNI-supported binary image formats
- Masking of variables (brain pixels)
- Censoring training samples
- Visualizing alphas as time series and linear weight vectors as functional overlays
- Classifying multiple categories

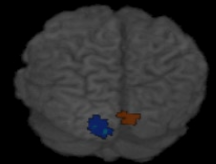
Supervised learning of fMRI with 3dsvm can be used for predicting brain states to enhance our understanding of brain systems, complementing the conventional emphasis on spatial mapping. The figure below classification formalism used by 3dsvm. For each time point, the brain voxel intensities can be represented in a high-dimensional vector space. During an fMRI experiment, each image is a point in the vector space of these points, a classification model can be estimated to distinguish between experimental states. After the model is determined, independent data can be assigned a class membership.



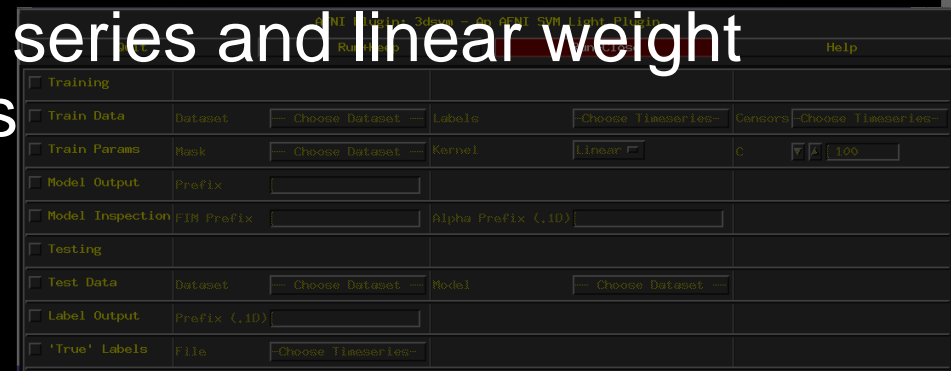
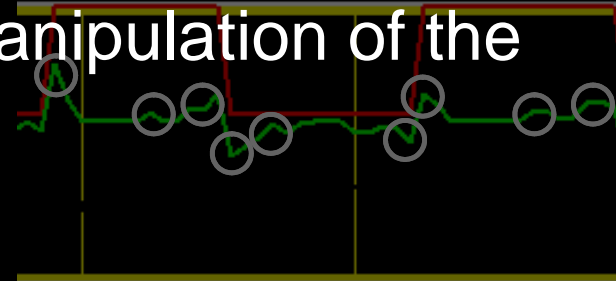
- 3dsvm is a command line program and plugin for AFNI, built around SVM-Light. It provides the ability to analyze functional magnetic resonance imaging (fMRI) data as described in (LaConte et al., 2005)
- lacontelab.org/3dsvm.html



3dsvm features



- Distributed with AFNI
- Reading AFNI-supported formats (including NIfTI)
 - Thus all preprocessing and data manipulation of the major software packages
- Masking of variables (brain pixels)
- Censoring training samples
- Visualizing alphas as time series and linear weight vectors as functional maps
- Multi-class classification
- Regression
- Support for non-linear kernels



3dsvm tour: basic steps

- Prepare training and test data sets

- fMRI (3D+t)
- Labels (1D) – labels for test data are optional (needed to calculate accuracy)
- Mask for training data (3D) – 3dsvm considers mask to be part of the model it generates

- 3dsvm training

- Creates a model that can be tested with independent data
- For convenience, inspecting the model
 - Model alphas (1D)
 - Weight vector map (3D)

- 3dsvm testing

- Calculates class and/or distance measure for each new timepoint
- Prediction accuracy (if test set labels are available)

3dsvm Plugin Snapshot

Support Vector Machine Analysis

AFNI Plugin: 3dsvm - An AFNI SVM Light Plugin

Quit Run+Keep Run+Close Help

<input type="checkbox"/> Training				
<input type="checkbox"/> Train Data	Dataset	--- Choose Dataset ---	Labels	--Choose Timeseries-- Censors --Choose Timeseries--
<input type="checkbox"/> Train Params	Mask	--- Choose Dataset ---	Kernel	Linear <input type="checkbox"/> C <input type="text" value="100"/>
<input type="checkbox"/> Model Output	Prefix	<input type="text"/>		
<input type="checkbox"/> Model Inspection	FIM Prefix	<input type="text"/>	Alpha Prefix (.1D)	<input type="text"/>
<input type="checkbox"/> Testing				
<input type="checkbox"/> Test Data	Dataset	--- Choose Dataset ---	Model	--- Choose Dataset ---
<input type="checkbox"/> Label Output	Prefix (.1D)	<input type="text"/>		
<input type="checkbox"/> 'True' Labels	File	--- Choose Timeseries--		

training

testing

Command Line

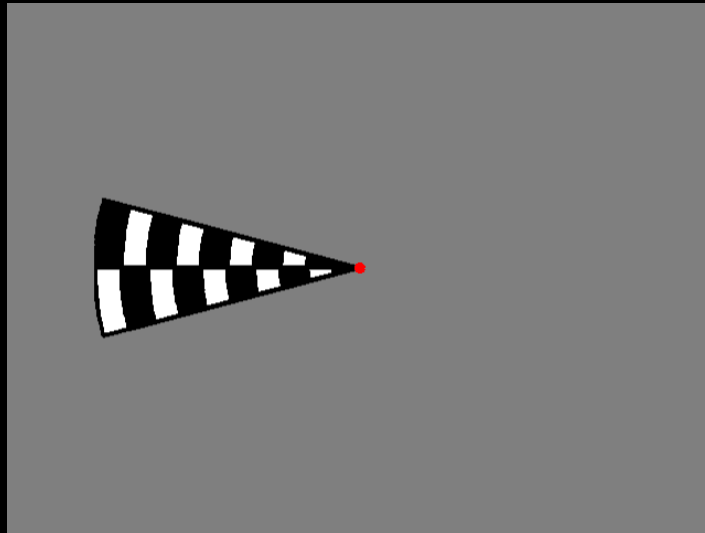
```
Training - 3dsvm -trainvol run1+orig \  
            -trainlabels run1_categories.1D \  
            -mask mask+orig \  
            -model model_run1
```

```
Testing - 3dsvm -testvol run2+orig \  
            -model model_run1+orig \  
            -predictions pred2_model1
```

Example:

Left vs. Right visual stimulus

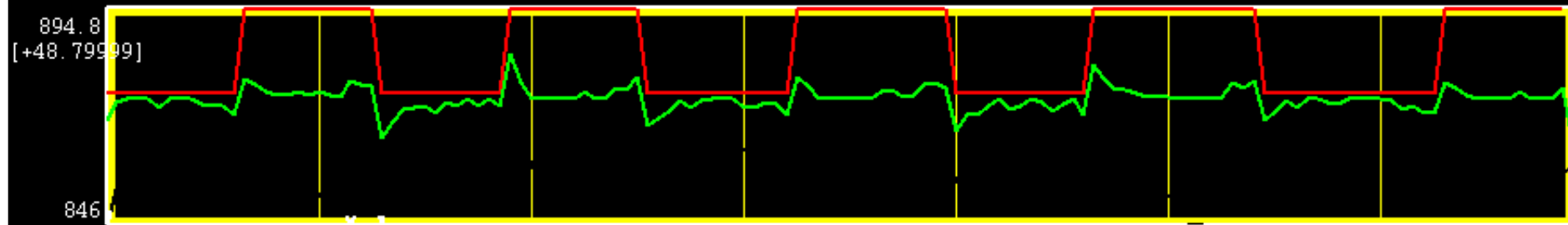
- 3T fmri 31 axial EPI slices, TR/TE = 2000/31 msec, voxel=3.4 X 3.4 X 4 mm).
- Randomized block lengths alternating between left and right stimulus.



Example:

Left vs. Right visual stimulus

- 3T fmri 31 axial EPI slices, TR/TE = 2000/31 msec, voxel=3.4 X 3.4 X 4 mm).
- Randomized block lengths alternating between left and right stimulus.

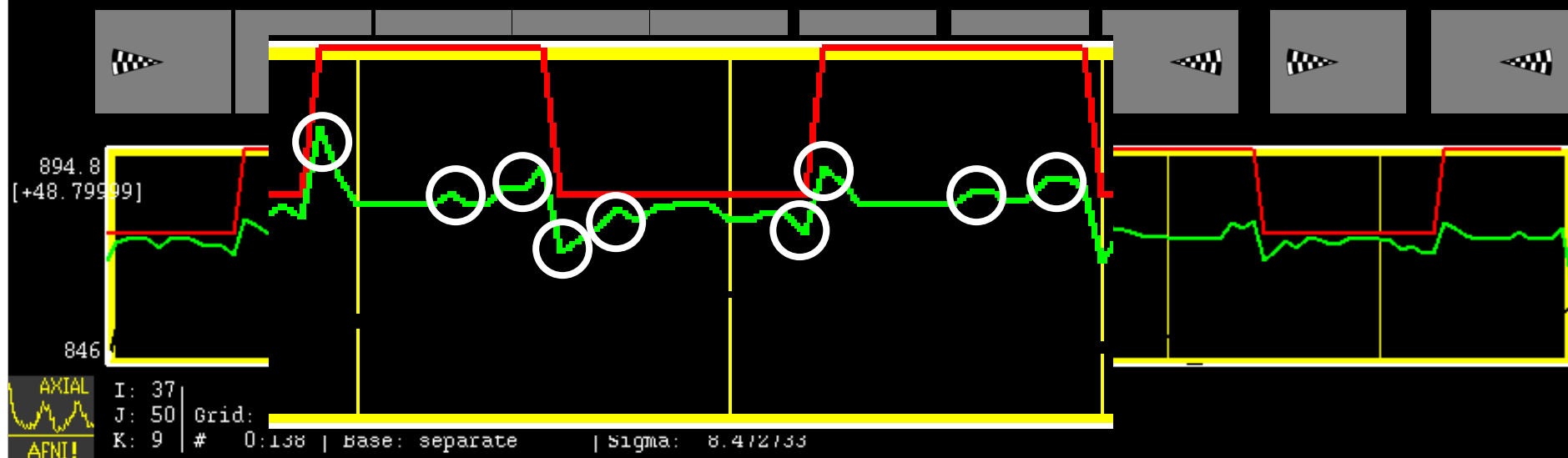


AXIAL I: 37
J: 50 | Grid: 20 | Scale: 2.5 pix/datum | Mean: 860.7698
AFNI! K: 9 | # 0:138 | Base: separate | Sigma: 8.472733

Example:

Left vs. Right visual stimulus

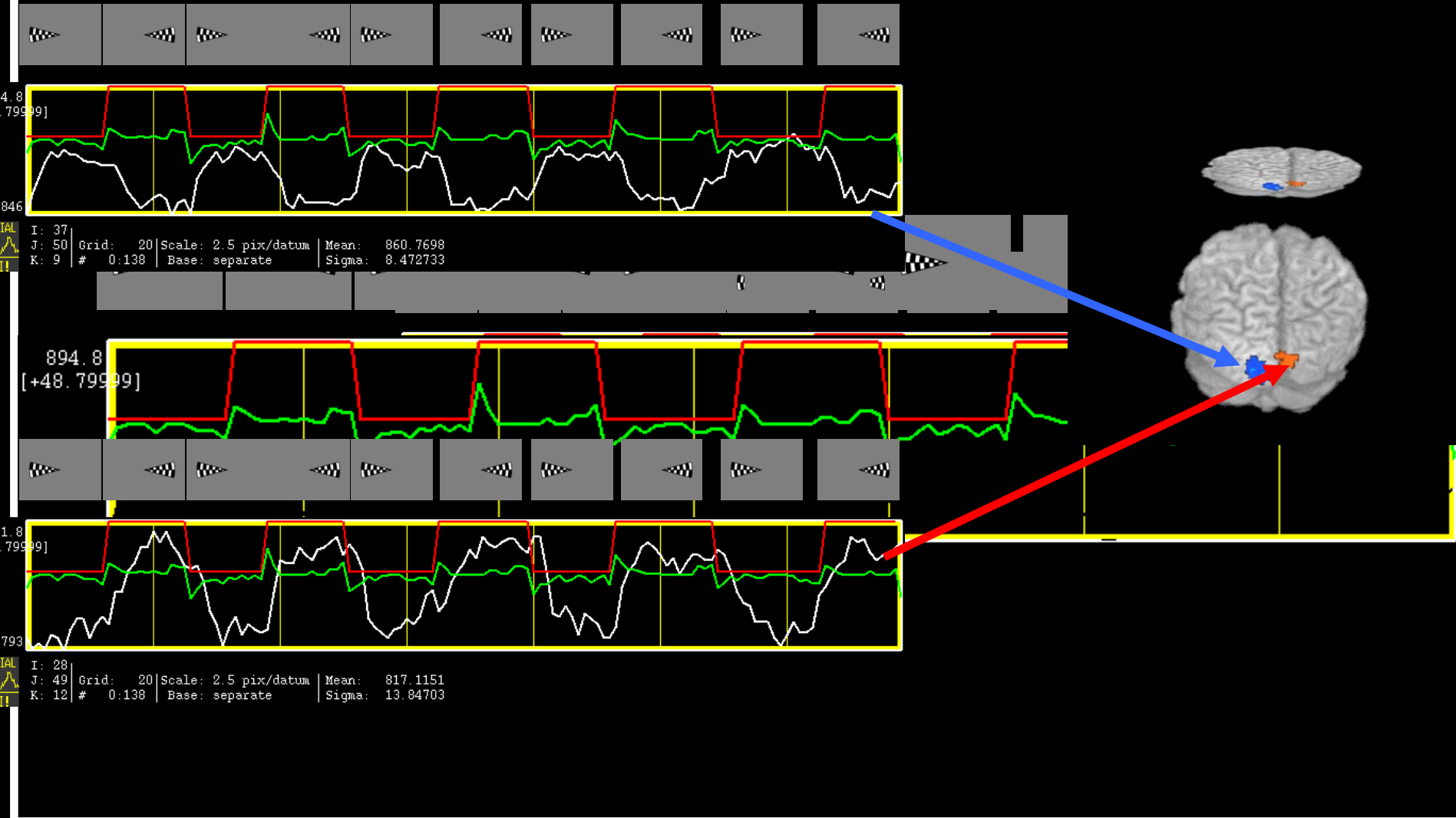
- 3T fmri 31 axial EPI slices, TR/TE = 2000/31 msec, voxel=3.4 X 3.4 X 4 mm).
- Randomized block lengths alternating between left and right stimulus.



Example:

Left vs. Right visual stimulus

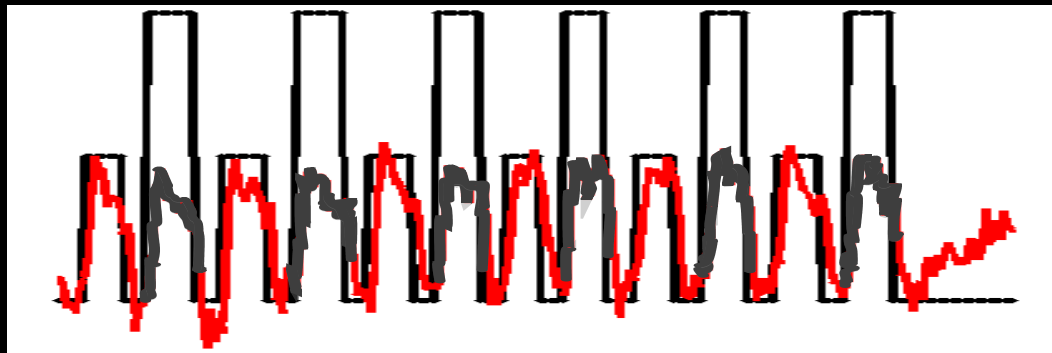
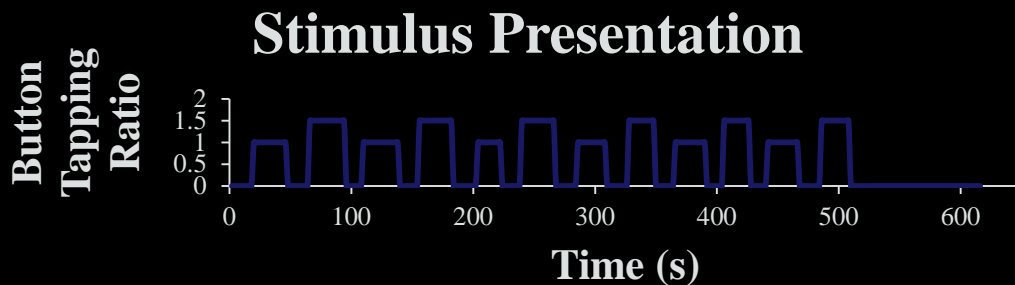
- 3T fmri 31 axial EPI slices, TR/TE = 2000/31 msec, voxel=3.4 X 3.4 X 4 mm).
- Randomized block lengths alternating between left and right stimulus.



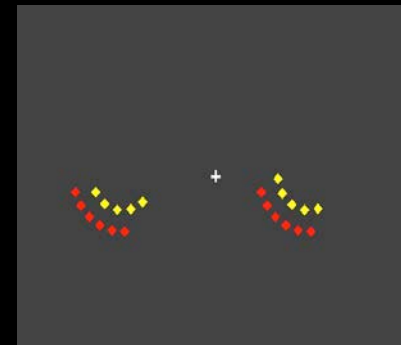
example 2

Bimanual coordination task:

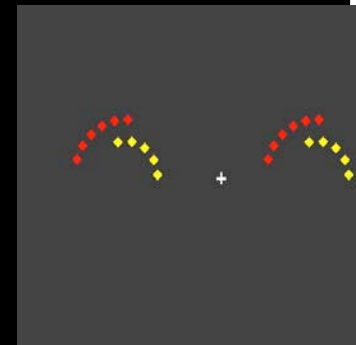
Feedback of both the left and right hand performance was provided by a four arc display described in (Klaiman and Karniel, 2006), where button tapping was used to move the inner arches to match the speed of the outer arches.



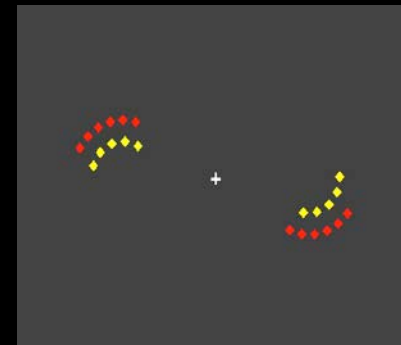
Example Stimulus Frames



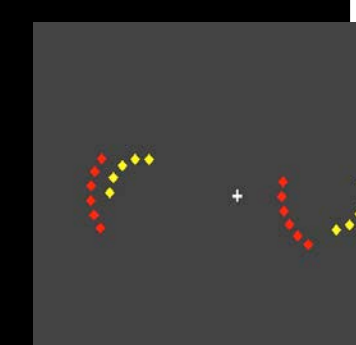
1:1 – Good Performance



1:1 – Poor Performance



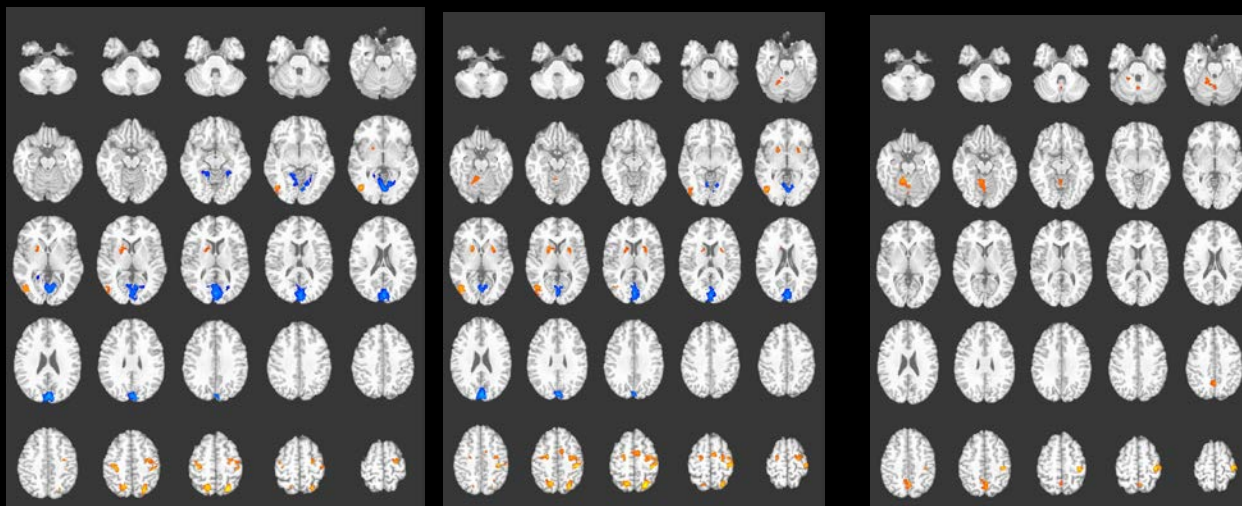
3:2 – Good Performance



3:2 – Poor Performance

example 2

Group brain maps of the SVM models for the 14 subjects



1:1 (red) vs. Fixation (blue)
FDR-corrected; $q < 0.01$

3:2 (red) vs. Fixation (blue)
FDR-corrected; $q < 0.01$

3:2 (red) vs. 1:1 (blue)
 $p = 0.001$

Distributed pattern for 1:1 & 3:2 rhythms:

- Bil. Precuneus (BA7)
- L. (1:1) & Bil. (3:2) postcentral (BA4)
- Bil. Lentiform
- R. culmen

Distributed pattern during Fixation:

- L. cuneus (BA18)

Distributed pattern unique to 1:1 rhythm:

- Bil. Precentral/M1 (BA4)
- R. inf. Temporal (BA37)
- L. sup. Parietal

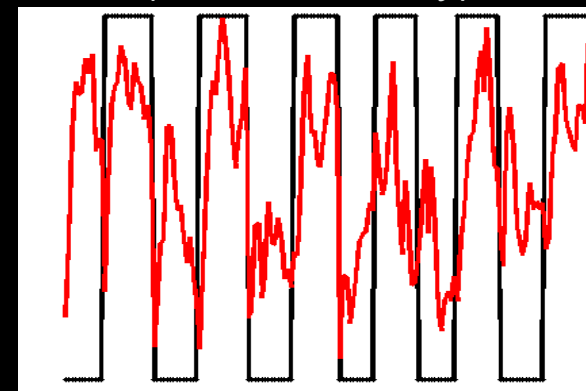
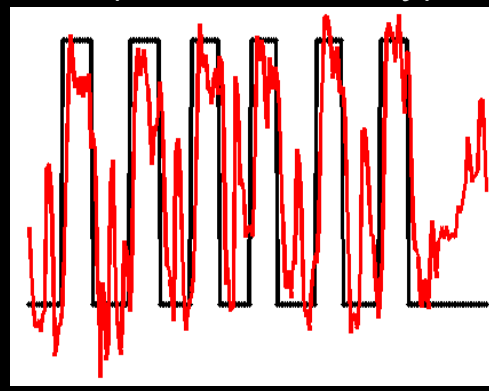
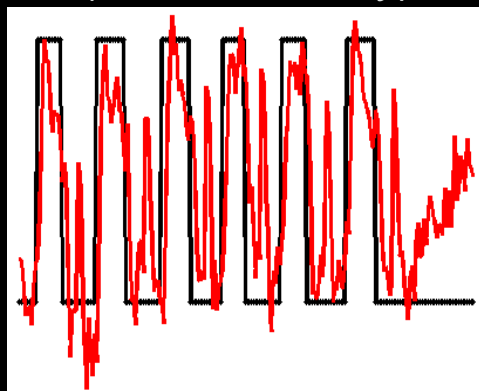
Distributed pattern unique to 3:2 rhythm:

- R. middle frontal (BA6)
- L. medial frontal (BA6)

1:1 vs. Fixation
(93% Accuracy)

3:2 vs. Fixation
(93% Accuracy)

3:2 vs. 1:1
(77% Accuracy)



Raw classifier output plotted over label files for a relatively accurate subject. The first two TRs have been removed for each transition between stimuli.

example 2

```
#!/bin/csh
```

```
# example by Prashant Prasad
```

```
3dsvm -trainvol volreg_run1_PPA+orig \  
-trainlabels LABEL_PPA_1.1D \  
-mask automask_run1_PPA+orig \  
-bucket bucket_run1_PPA \  
-model model_run1_PPA
```

```
3dsvm -classout \  
-testvol volreg_run2_PPA+orig \  
-testlabels LABEL_PPA_2.1D \  
-model model_run1_PPA+orig \  
-predictions pred_run2_frmRun1_classout
```

```
3dsvm -testvol volreg_run2_PPA+orig \  
-testlabels LABEL_PPA_2.1D \  
-model model_run1_PPA+orig \  
-predictions pred_run2_frmRun1
```

example 2

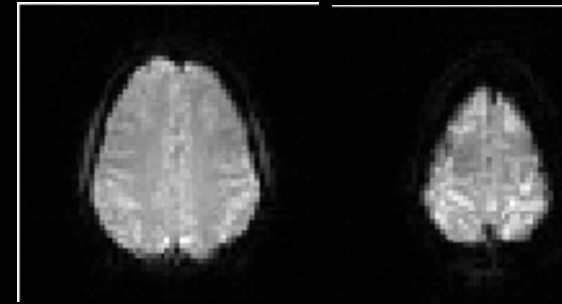
```
#!/bin/csh
```

```
# example by Prashant Prasad
```

```
3dsvm -trainvol volreg_run1_PPA+orig \  
-trainlabels LABEL_PPA_1.1D \  
-mask automask_run1_PPA+orig \  
-bucket bucket_run1_PPA \  
-model model_run1_PPA
```

```
3dsvm -classout \  
-testvol volreg_run2_PPA+orig \  
-testlabels LABEL_PPA_2.1D \  
-model model_run1_PPA+orig \  
-predictions pred_run2_frmRun1_classout
```

```
3dsvm -testvol volreg_run2_PPA+orig \  
-testlabels LABEL_PPA_2.1D \  
-model model_run1_PPA+orig \  
-predictions pred_run2_frmRun1
```



example 2

```
#!/bin/csh
```

```
# example by Prashant Prasad
```

```
3dsvm -trainvol volreg_run1_PPA+orig \  
-trainlabels LABEL_PPA_1.1D \  
-mask automask_run1_PPA+orig \  
-bucket bucket_run1_PPA \  
-model model_run1_PPA
```

```
3dsvm -classout \  
-testvol volreg_run2_PPA+orig \  
-testlabels LABEL_PPA_2.1D \  
-model model_run1_PPA+orig \  
-predictions pred_run2_frmRun1_classout
```

```
3dsvm -testvol volreg_run2_PPA+orig \  
-testlabels LABEL_PPA_2.1D \  
-model model_run1_PPA+orig \  
-predictions pred_run2_frmRun1
```



example 2

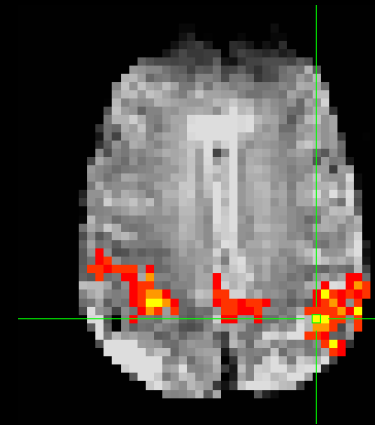
```
#!/bin/csh
```

```
# example by Prashant Prasad
```

```
3dsvm -trainvol volreg_run1_PPA+orig \  
-trainlabels LABEL_PPA_1.1D \  
-mask automask_run1_PPA+orig \  
-bucket bucket_run1_PPA \  
-model model_run1_PPA
```

```
3dsvm -classout \  
-testvol volreg_run2_PPA+orig \  
-testlabels LABEL_PPA_2.1D \  
-model model_run1_PPA+orig \  
-predictions pred_run2_frmRun1_classout
```

```
3dsvm -testvol volreg_run2_PPA+orig \  
-testlabels LABEL_PPA_2.1D \  
-model model_run1_PPA+orig \  
-predictions pred_run2_frmRun1
```



example 2

```
#!/bin/csh
```

```
# example by Prashant Prasad
```

```
3dsvm -trainvol volreg_run1_PPA+orig \  
-trainlabels LABEL_PPA_1.1D \  
-mask automask_run1_PPA+orig \  
-bucket bucket_run1_PPA \  
-model model_run1_PPA
```

```
3dsvm -classout \  
-testvol volreg_run2_PPA+orig \  
-testlabels LABEL_PPA_2.1D \  
-model model_run1_PPA+orig \  
-predictions pred_run2_frmRun1_classout
```

```
3dsvm -testvol volreg_run2_PPA+orig \  
-testlabels LABEL_PPA_2.1D \  
-model model_run1_PPA+orig \  
-predictions pred_run2_frmRun1
```


example 2

```
#!/bin/csh
```

```
# example by Prashant Prasad
```

```
3dsvm -trainvol volreg_run1_PPA+orig \  
-trainlabels LABEL_PPA_1.1D \  
-mask automask_run1_PPA+orig \  
-bucket bucket_run1_PPA \  
-model model_run1_PPA
```

```
3dsvm -classout \  
-testvol volreg_run2_PPA+orig \  
-testlabels LABEL_PPA_2.1D \  
-model model_run1_PPA+orig \  
-predictions pred_run2_frmRun1_classout
```

```
3dsvm -testvol volreg_run2_PPA+orig \  
-testlabels LABEL_PPA_2.1D \  
-model model_run1_PPA+orig \  
-predictions pred_run2_frmRun1
```

```
#!/bin/csh
```

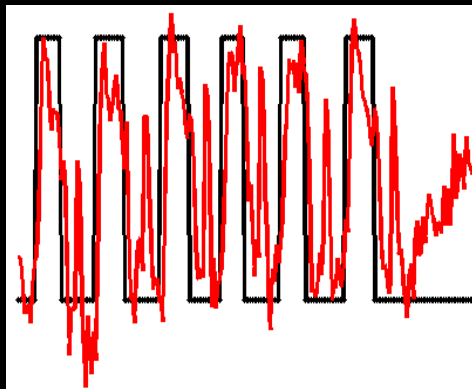
```
# example by Prashant Prasad
```

```
3dsvm -trainvol volreg_run1_PPA+orig \  
-trainlabels LABEL_PPA_1.1D \  
-mask automask_run1_PPA+orig \  
-bucket bucket_run1_PPA \  
-model model_run1_PPA
```

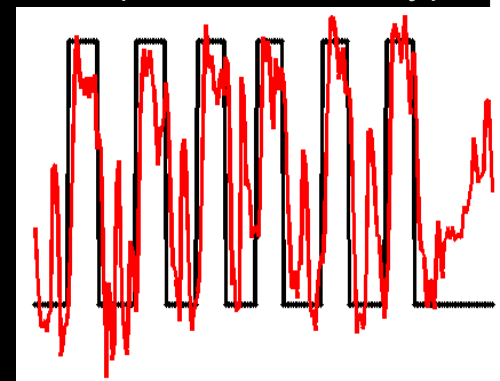
```
3dsvm -classout \  
-testvol volreg_run2_PPA+orig \  
-testlabels LABEL_PPA_2.1D \  
-model model_run1_PPA+orig \  
-predictions pred_run2_frmRun1_classout
```

```
3dsvm -testvol volreg_run2_PPA+orig \  
-testlabels LABEL_PPA_2.1D \  
-model model_run1_PPA+orig \  
-predictions pred_run2_frmRun1
```

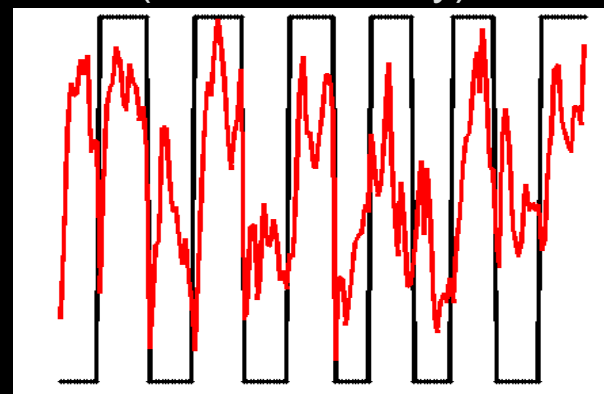
1:1 vs. Fixation
(93% Accuracy)



3:2 vs. Fixation
(93% Accuracy)



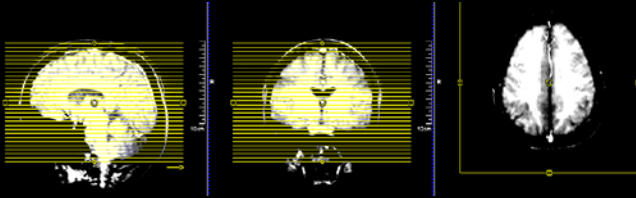
3:2 vs. 1:1
(77% Accuracy)



Demonstration Experiment

Initial scans

- Localizer (9 seconds)

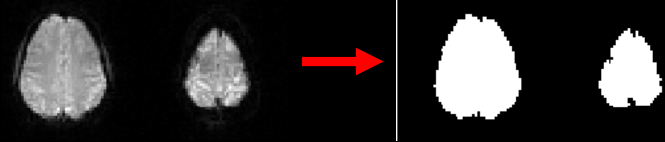


- Anatomical scan (4.5 minutes)

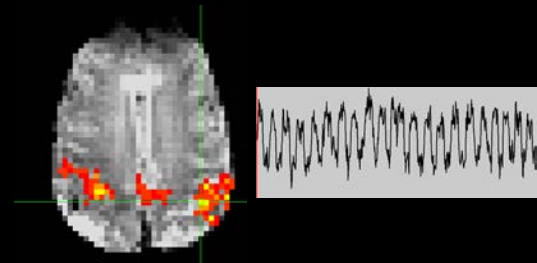


fMRI runs

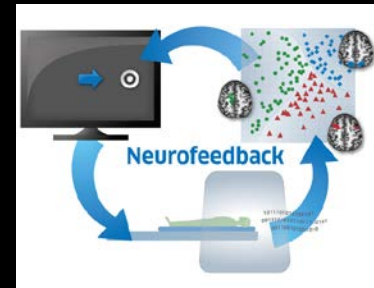
- Masking run (10 seconds)



- Training run (6 minutes)

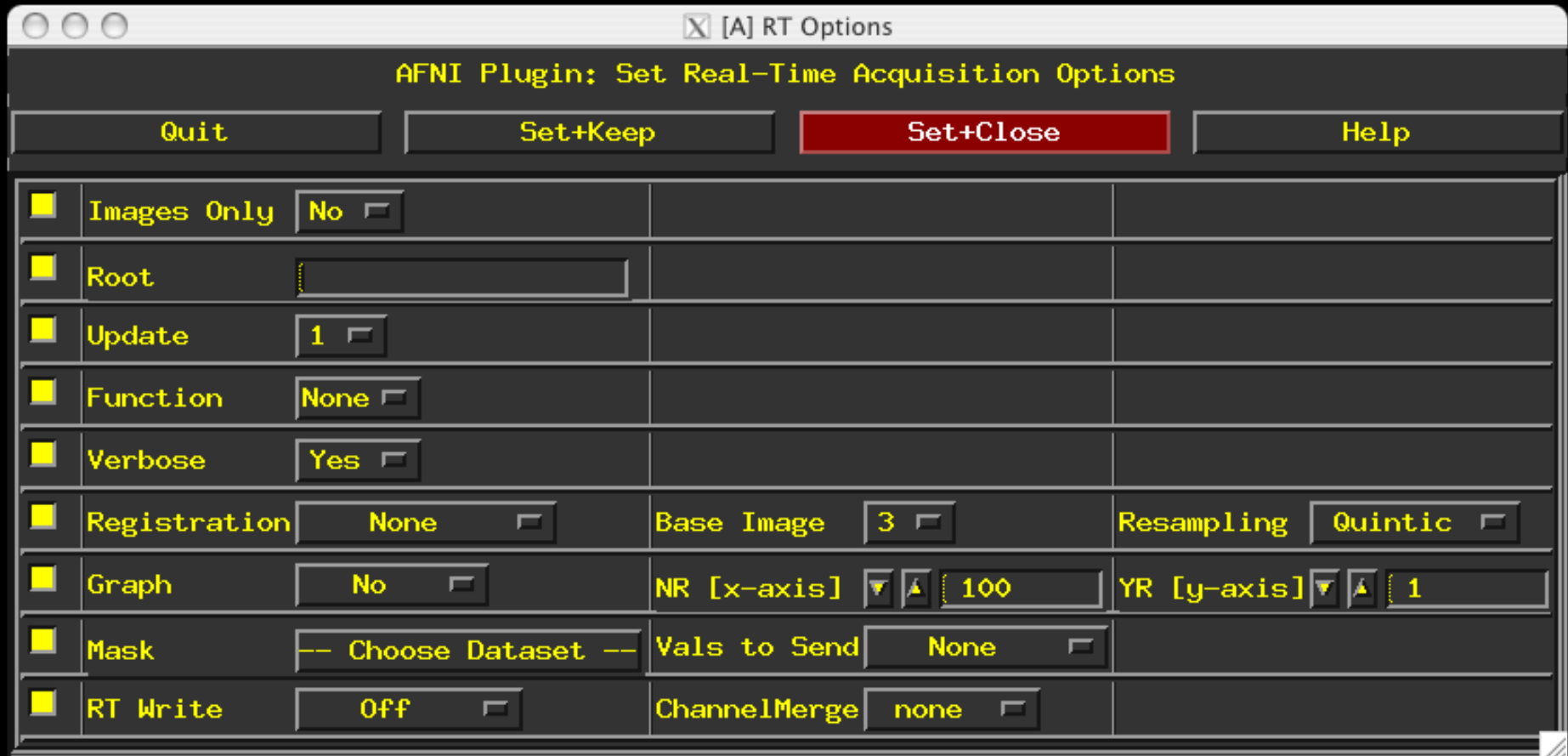


- Feedback run (6 minutes)



Setting up AFNI's RT plugin

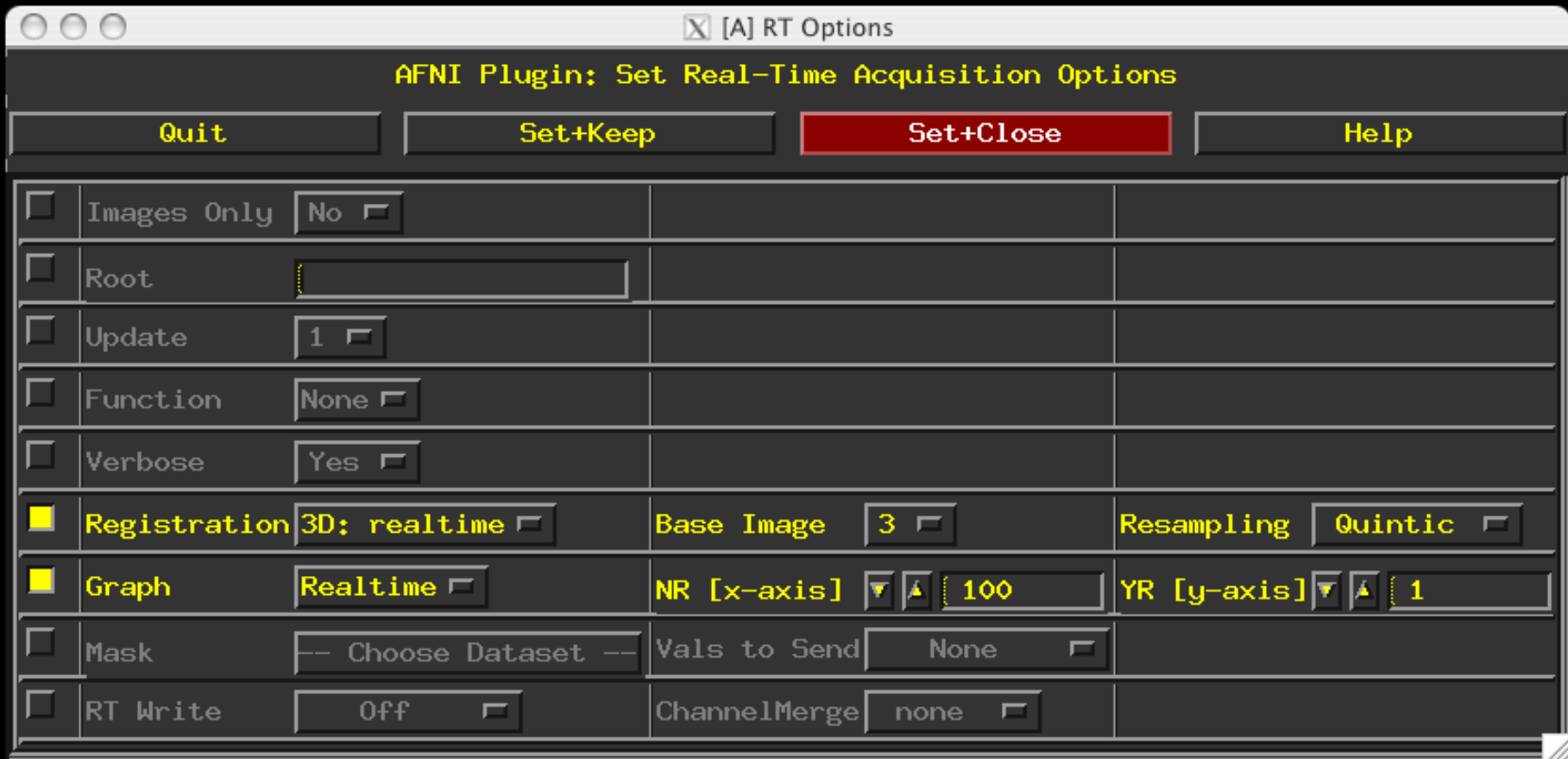
- Manually
 - Good for learning and demo



Setting up AFNI's RT plugin

- Via Environment Variables

```
setenv AFNI_REALTIME_Registration 3D:_realtime
setenv AFNI_REALTIME_Graph Realtime
```



AFNI help resources

- Readme files
 - README.driver
 - README.environment
 - README.realtime
- Demo material available on:
<http://afni.nimh.nih.gov>
- Automation
 - @DriveAfni script
 - @DriveSuma script
 - @DO.examples
- Sample programs
 - rtfeedme.c
 - Dimon.c
 - serial_helper.c
 - realtime_receiver.py

Multivoxel pattern-based real-time fMRI

- Conceptual overview
 - Supervised learning for fMRI
 - (classification and regression)
 - Integration with MRI and real-time platforms
- Experimental flexibility
 - Whole-brain models trained to the distributed patterns of each individual for neurofeedback, BCIs, adaptive fMRI paradigms, etc.
 - Feedback improves classifications
 - Employs frontoparietal networks
- Practicalities and additional resources
 - 3dsvm and AFNI's realtime plugin