GCNeuro

Graph Convolutional Network for Multimodal Neuroimage Analysis

Introduction, Justification and Background

QRG Spring 2019 (1st) Presentation

Christian McDaniel

Presentation Roadmap:

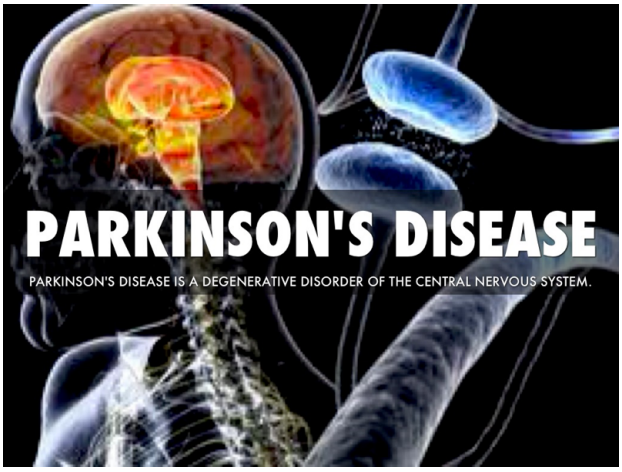
- Parkinson's Disease & Research
- Neuroimage Analysis
  - Basic MRI Physics
  - Modalities (MRI, dMRI, fMRI)
  - Artifacts
- Data Preparation

Presentation Goals:

- Provide an understanding for the basics of MRI Imaging and the resulting data
- Demonstrate the neuroimage preprocessing pipeline
- Introduce the Restricted Boltzmann Machine and report previous results
- Setup for the next presentation on the Graph Convolutional Network

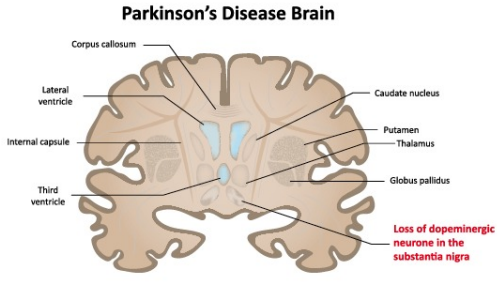
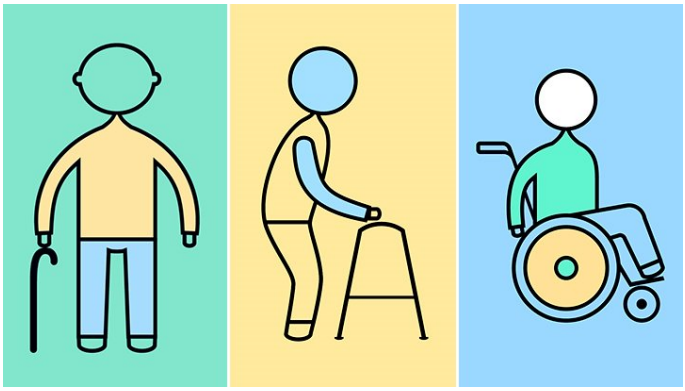
Parkinson's Disease

- One of the most prevalent neurodegenerative diseases
  - 500,000+ Americans
  - 14th highest cause of death in US



Parkinson's Disease - Pathology

- Predominantly affects the dopaminergic neurons in the substantia nigra
- Highly progressive
- Highly heterogeneous - clinical manifestations vary on an individual basis

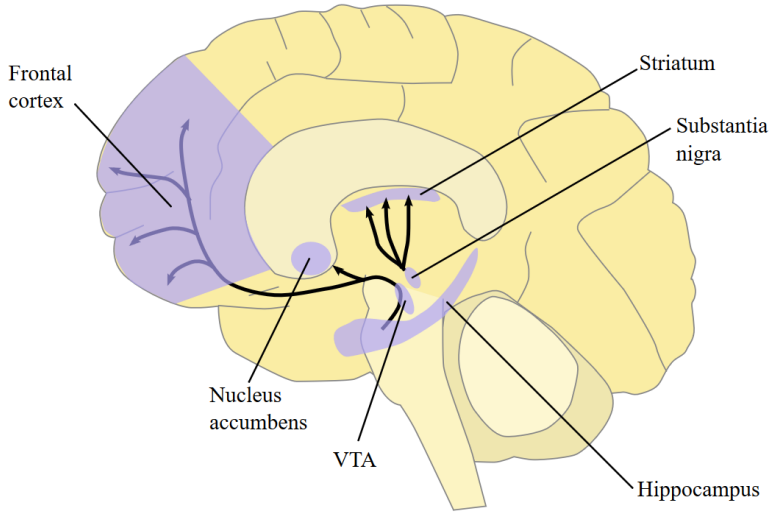
_	_
	

Parkinson's Disease - Pathology cont'd

Nigrostriatal Dopamine pathway

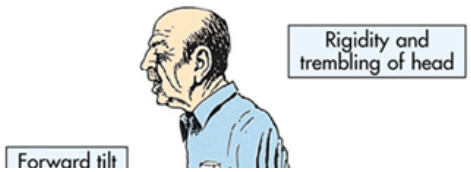
- Structural
- Functional
- Connectivity

Mesocorticolimbic and nigrostriatal dopamine pathways



Parkinson's Disease - Diagnosis

Diagnosis is largely clinically based:

–	–
+ Clinical assessments	

Parkinson's Disease - Diagnosis

+ Diagnosis has traditionally involved a degree of subjectivity, and its distinction between similar diseases can be somewhat arbitrary

Parkinson's Disease - Research

Parkinson's Disease - Research

+ There has been much effort to develop more rigorous, qualitative diagnostic tools

Parkinson's Disease - Research

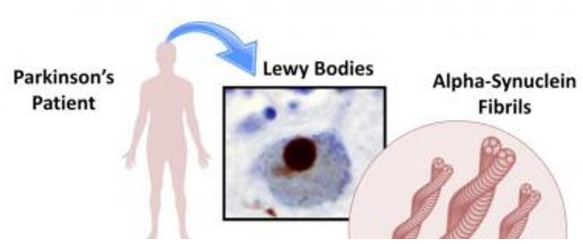
Some of the research is done using:

Clinical Data

Parkinson's Disease - Research

Other research is done using:

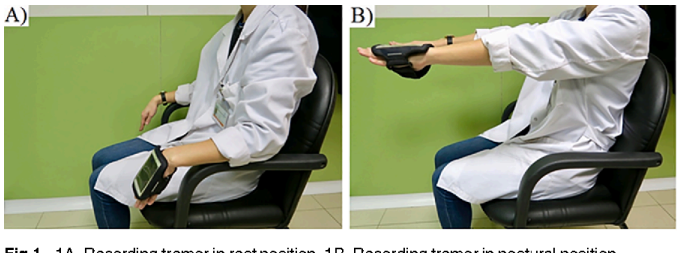
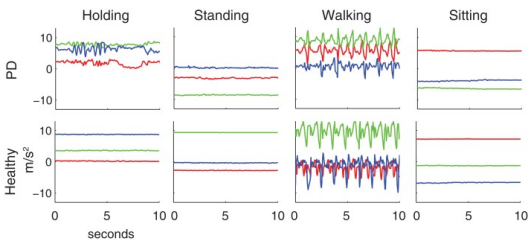
Genetic and Molecular Data



Parkinson's Disease - Research

Other research is done using:

Accelerometer Data

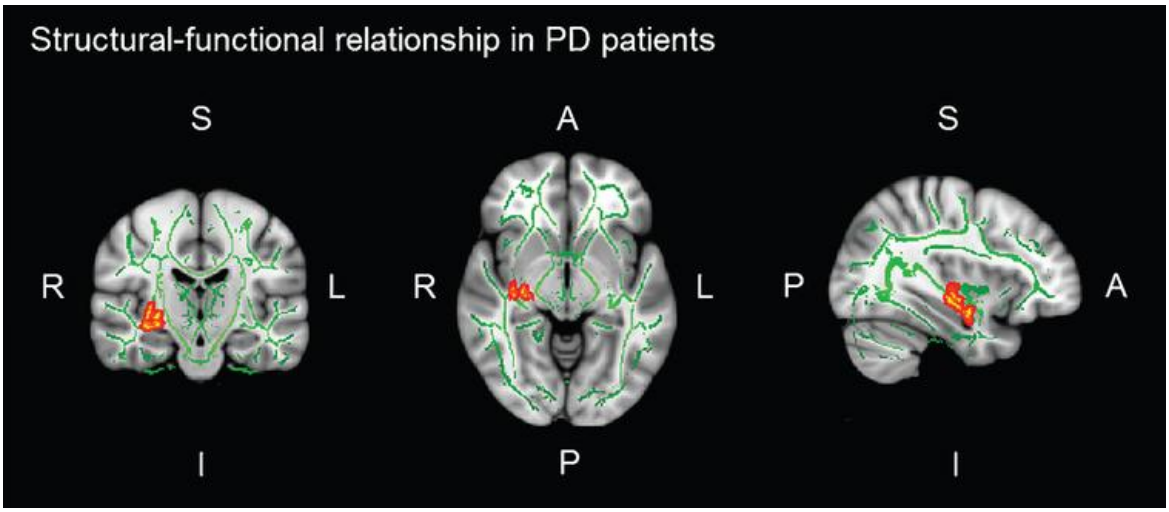
–	–
	

... Still waiting on that data (and paper) .... ¯_(ツ)_/¯

Parkinson's Disease - Research

And other research is done using:

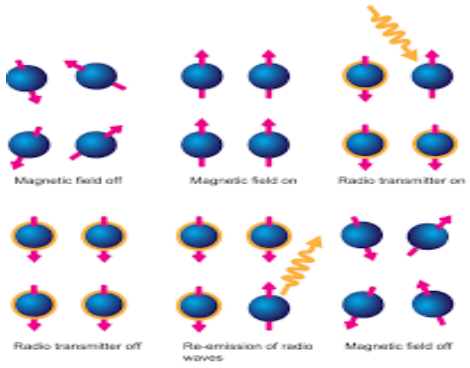
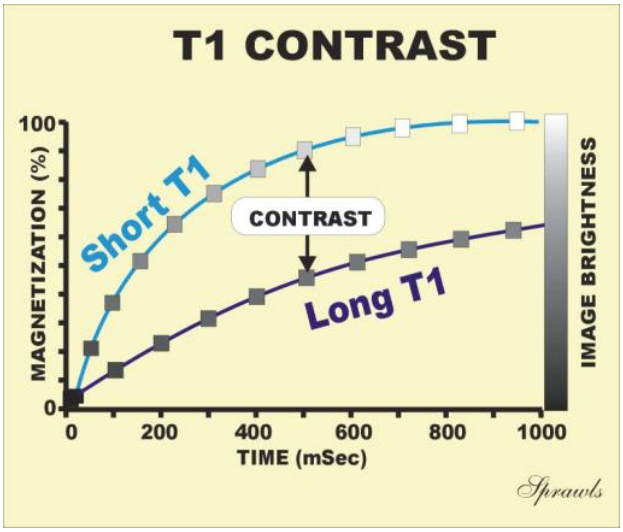
Imaging Data



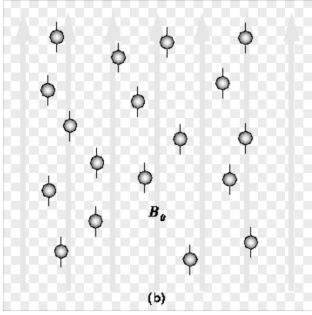
Magnetic-based Neuroimaging Review:

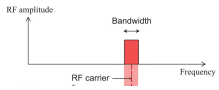
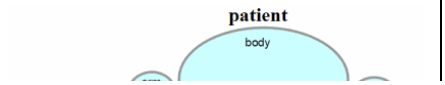
Optical Imaging	X-Ray Imaging	Magnetic Resonance Imaging
	2D X-ray detector plane	

Magnetic Resonance Imaging

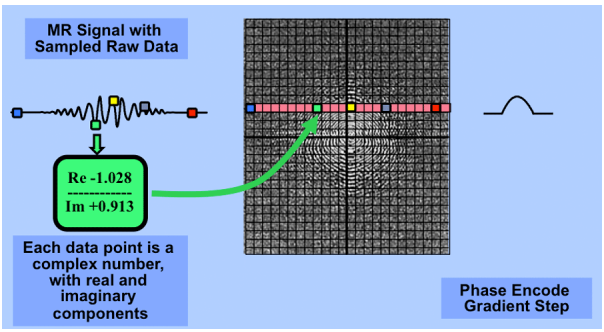
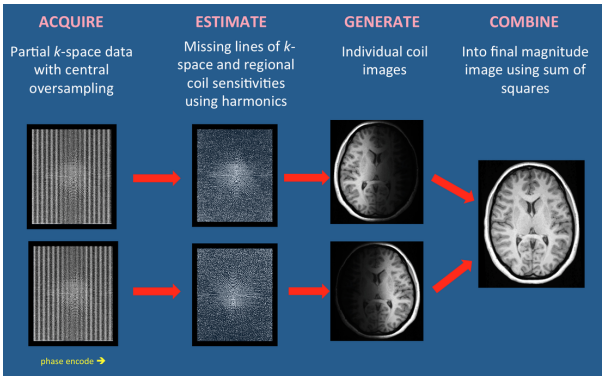
Signal Generation and Reception	Tissue Differentiation (T1 Relaxation)
	

Magnetic Resonance Imaging - Signal Localization via Gradients



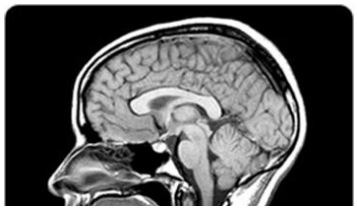
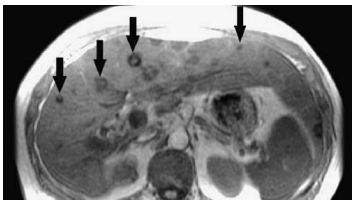
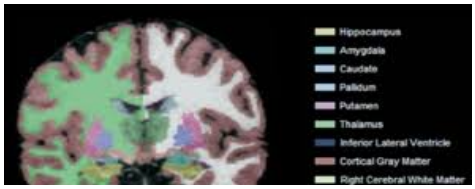
Slice selection (Z-axis)	Phase Encoding (Y-axis)	Frequency Encoding (X-axis)
		

Magnetic Resonance Imaging - Image Construction

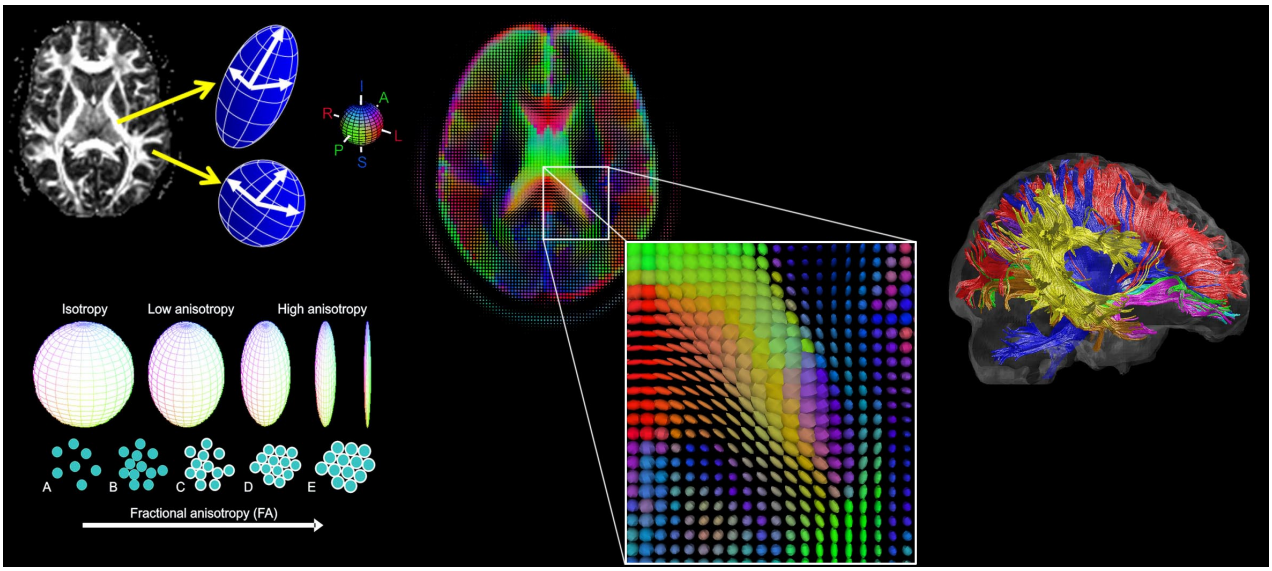
K-Space	Fourier Transform
	

MRI Modalities

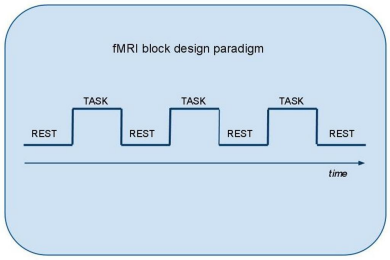
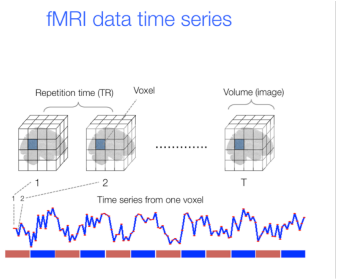
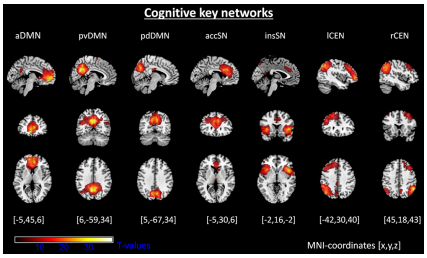
MRI: Anatomical Insights

	.	.
		

Diffusion Weighted MR Imaging: Connectivity Insights



function MR Imaging: Functional Insights

experiment paradigm	voxel-wise tracking	network inference
		

MRI combines...

MRI combines...

... a symphony of magnetic transmissions ...

MRI combines...

... a symphony of magnetic transmissions ...

... physical properties at the atomic level ...

MRI combines...

... a symphony of magnetic transmissions ...

... physical properties at the atomic level ...

... digitization and mathematical computation ...

MRI combines...

... a symphony of magnetic transmissions ...

... physical properties at the atomic level ...

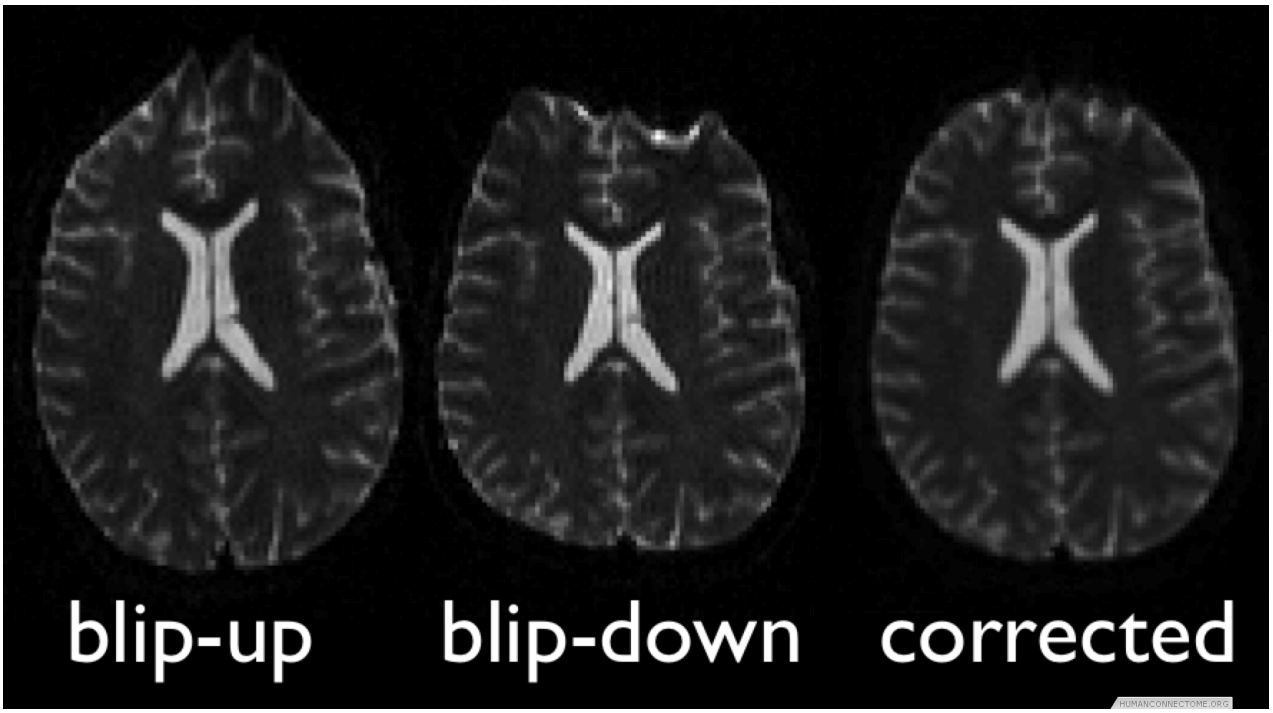
... digitization and mathematical computation ...

... and each introduces a source of noise!

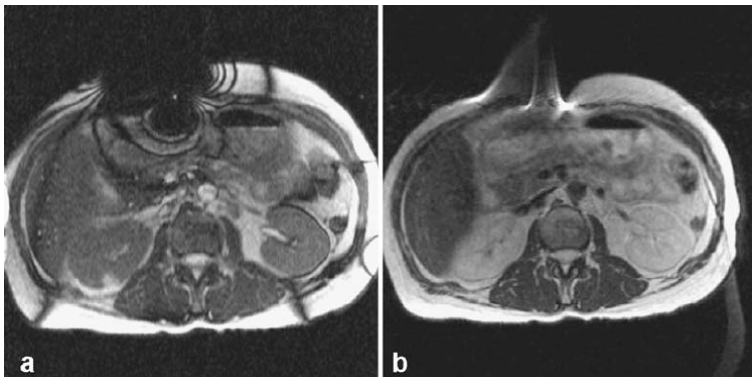
Noise: MRI Artifacts

Signal Transmission - Zipper Effect	Signal Acquisition - Aliasing

Noise: MRI Artifacts - Gradient-Related Artifacts

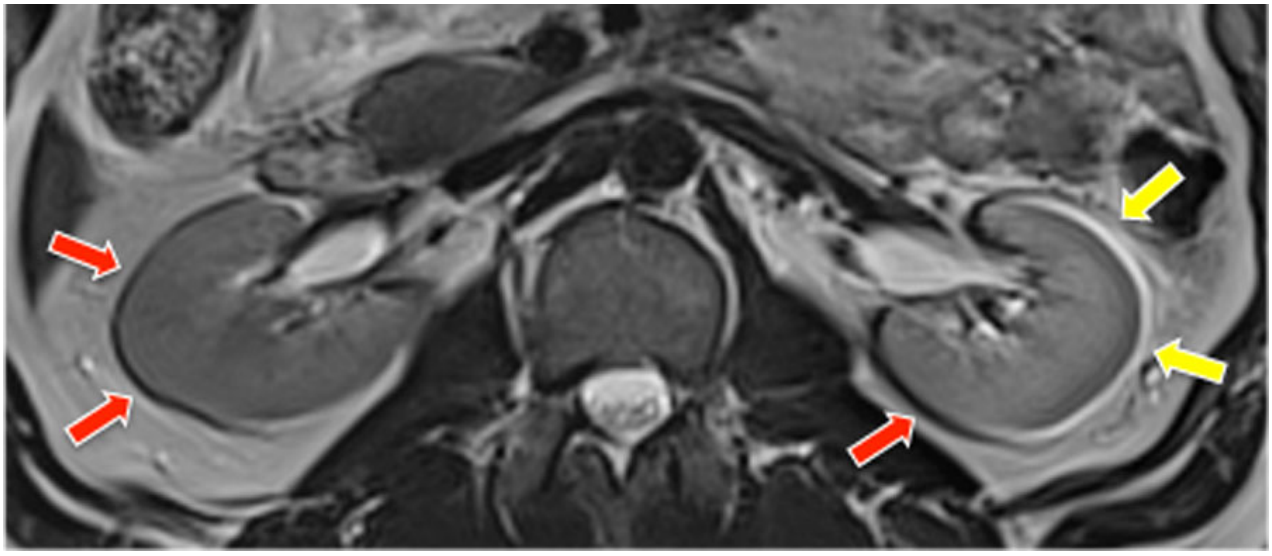


Noise: MRI Artifacts - Magnetization

Ghosting	Susceptibility
	

Noise: MRI Artifacts - Magnetization

Chemical Shift:



The acquisition physics are directly tied to the artifacts.....

E.g., *Hidden Cues: Deep Learning for Alzheimer's Disease* (2016)

_	tSNE
+ Used tri-axial RAW MRI data	

So with that in mind ...

let's get started!

First step, get some data!

The PPMI Dataset

- Large-scale multinational dataset
- Imaging, biological sampling, clinical/behavioral assessment data

Now..... Get to cleanin'

Preprocessing

First step, Formatting!

Let's look at what we get from the image lab...

Preprocessing

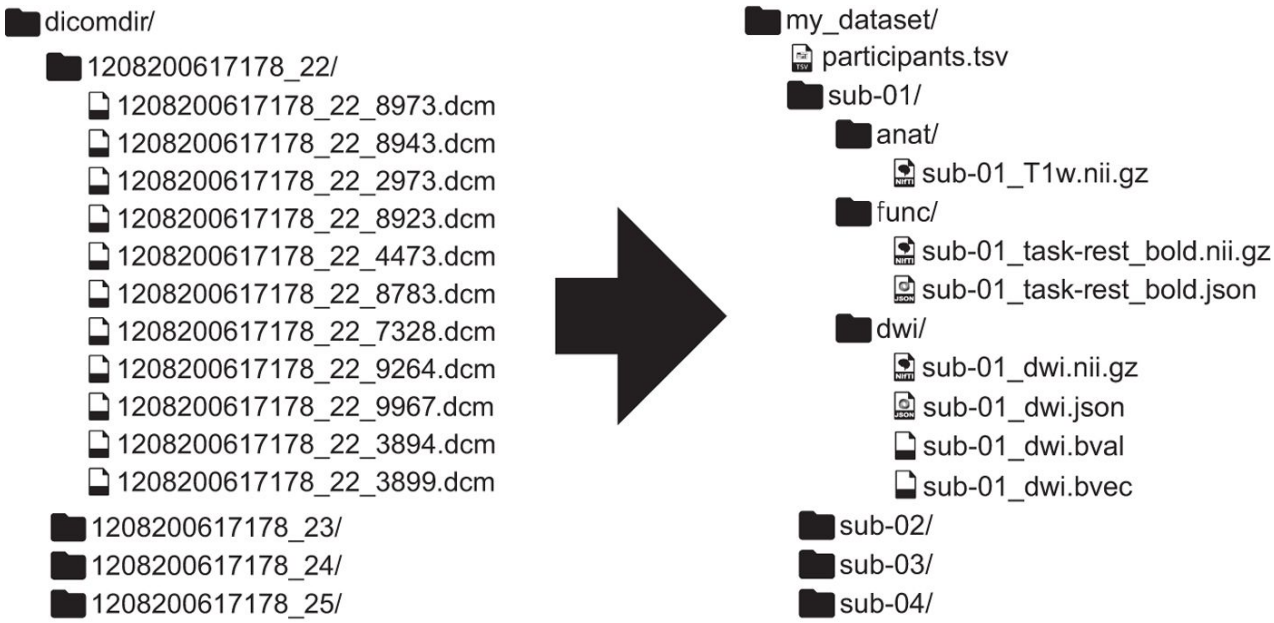
Step 1: Dicom to Nifti

DICOM	NIFTI
Digital Imaging and Communications in Medicine	Neuroimaging Informatics Technology Initiative
Many volumes are taken for one image	Processing is concerned with the

Preprocessing

Step 1: Dicom to Nifti

BIDS: Brain Imaging Data Structure



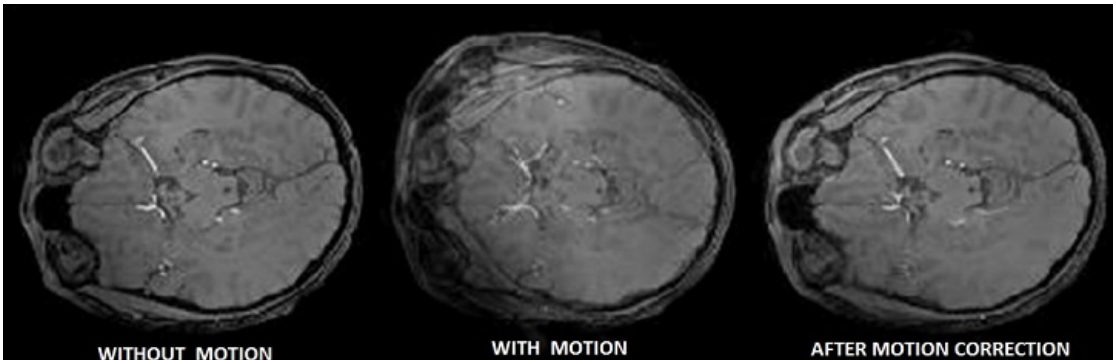
Preprocessing

Preprocessing - MRI Anatomical Images

In [ ]:
```
# bash_command('fsleyes sub-09_run-01_T1w.nii')
```

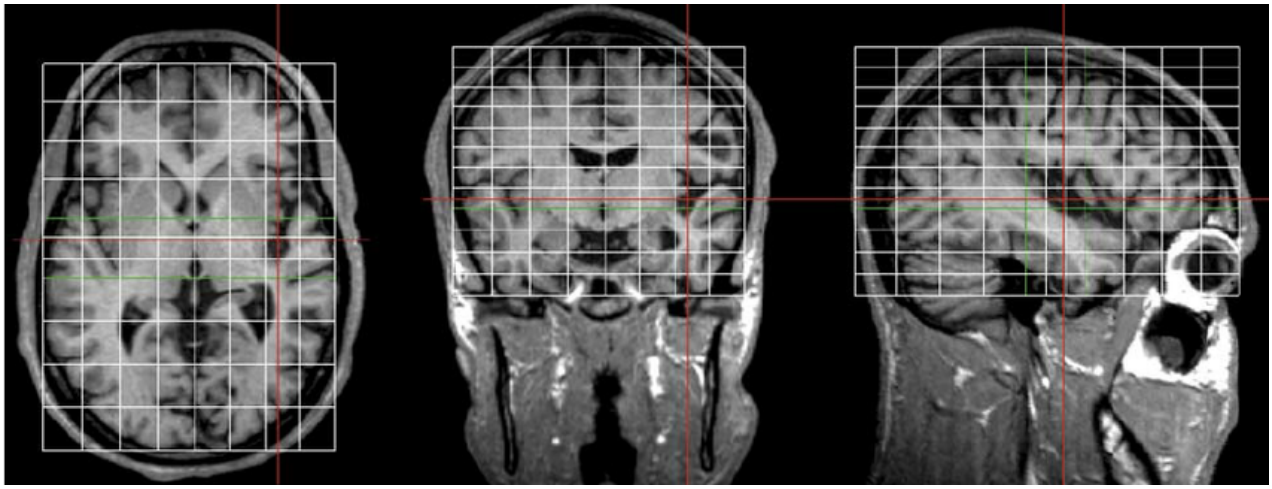
Motion Correction

```
#-----------------------------------------
#@# MotionCor Sat Nov  3 12:02:41 EDT 2018
```



Get it into a standard space -> use an Atlas

```
#-----------------------------------------
#@# Talairach Sat Nov  3 12:02:51 EDT 2018
```



Voxel Intensity Normalization

```
#-----------------------------------------
#@# Nu Intensity Correction Sat Nov  3 12:04:16 EDT 2018
mri_binarize --i ./tmp.mri_nu_correct.mni.14123/nu2.mnc --min -1 --o ./tmp.mri_
nu_correct.mni.14123/ones.mgz
#-----------------------------------------
#@# Intensity Normalization Sat Nov  3 12:06:05 EDT 2018
---------------------------------
3d normalization pass 1 of 2
white matter peak found at 110
white matter peak found at 105
gm peak at 56 (56), valley at 26 (26)
csf peak at 29, setting threshold to 47
building Voronoi diagram...
performing soap bubble smoothing, sigma = 8...
---------------------------------
```

Skull Stripping

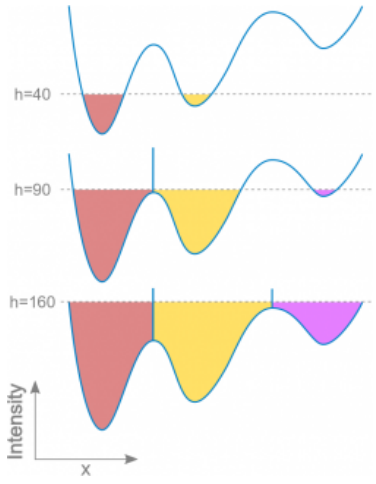
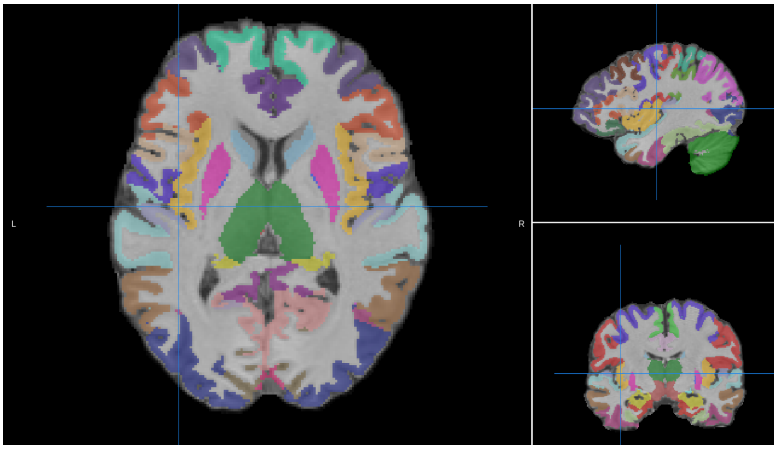
In [1]:
```
# Checkpoint: Lets check the progress
#bash_command('fsleyes sub-09_run-01_fs/mri/T1.mgz sub-09_run-01_fs/mri/brainmask.mgz')

# and look at brain.mgz in MRIcro for surface viz
```

Segmentation - Watershed algo

```
*********************************************
The input file is T1.mgz
The output file is brainmask.auto.mgz
Weighting the input with atlas information before watershed
************************WATERSHED*************************
Sorting...
      first estimation of the COG coord: x=122 y=124 z=105 r=95
      first estimation of the main basin volume: 3617151 voxels
      Looking for seedpoints
        2 found in the cerebellum
        15 found in the rest of the brain
      global maximum in x=146, y=113, z=80, Imax=255
      CSF=15, WM_intensity=110, WM_VARIANCE=5
      WM_MIN=110, WM_HALF_MIN=110, WM_HALF_MAX=110, WM_MAX=110
      preflooding height equal to 10 percent
done.
Analyze...

      main basin size=2383667146866273 voxels, voxel volume =1.000
```

a	b
	

mri_strip_skull: done peeling brain

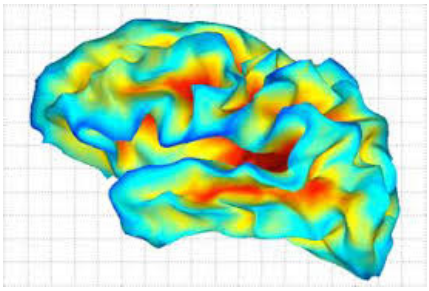
In [5]:
```
# Checkpoint
#bash_command('freeview -v aseg.presurf.mgz:colormap=lut:opacity=1.0')
# pretty good, but edges are not quite right
```

Surface Tessellation

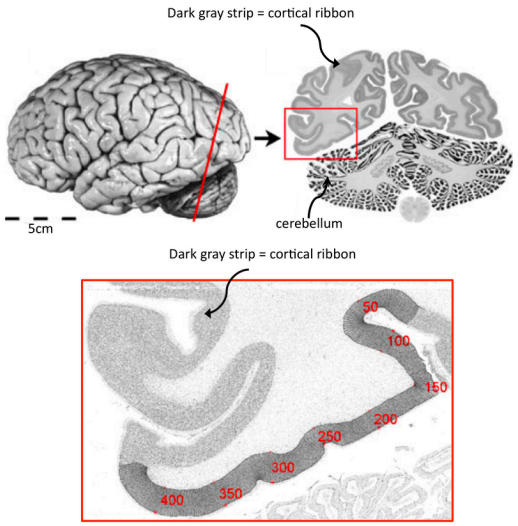
```
#--------------------------------------
#@# SubCort Seg Sat Nov  3 14:31:51 EDT 2018
#-----------------------------------------
#@# WM Segmentation Sat Nov  3 15:23:59 EDT 2018
#-----------------------------------------
#@# Tessellate lh Sat Nov  3 15:26:37 EDT 2018
#-----------------------------------------
```

Surface Smoothing

```
#@# Smooth1 rh Sat Nov  3 15:26:55 EDT 2018
#-----------------------------------------
#@# Inflation1 lh Sat Nov  3 15:27:02 EDT 2018
#-----------------------------------------
#@# Fix Topology Copy lh Sat Nov  3 15:33:20 EDT 2018
topology fixing took 103.9 minutes
```



In [ ]:
```
# see wm.seg.mgz in MRIcro!
```

_	_
#@# Make White Surf lh Sat Nov 3 23:57:50 EDT 2018 #@# Surf Reg rh Sun Nov 4 01:51:36 EDT 2018 #@# Jacobian white lh Sun Nov 4 01:45:45 EST 2018 #@# AvgCurv lh Sun Nov 4 01:45:48 EST 2018 #@# Cortical Parc lh Sun Nov 4 01:45:51 EST 2018 #@# Make Pial Surf lh Sun Nov 4 01:46:16 EST 2018 #@# Surf Volume lh Sun Nov 4 02:10:26 EST 2018 #@# Cortical ribbon mask Sun Nov 4 02:10:32 EST 2018	

In [6]:
```
#bash_command('fsleyes rh.ribbon.mgz')
```

This takes a long time!

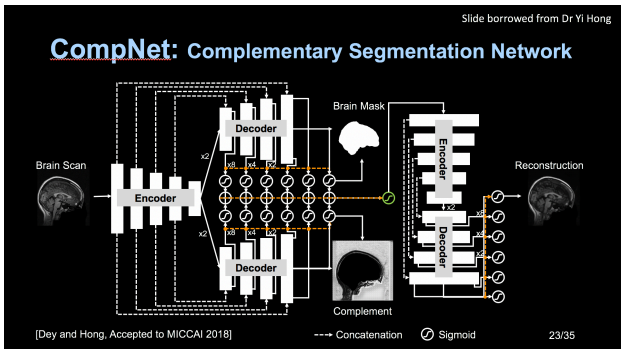
```
Started at Sat Nov 3 12:02:31 EDT 2018
Ended   at Sun Nov 4 02:55:07 EST 2018
#@#%# recon-all-run-time-hours 15.877
recon-all -s sub-02_run_01_fs finished without error at Sun Nov  4 02:55:07 EST
2018
done
```

In [7]:
```
# But yields a nice result: Let's take a look!

#bash_command('freeview -v sub-09_run-01_fs/mri/T1.mgz sub-09_run-01_fs/mri/wm.m
gz sub-09_run-01_fs/mri/brainmask.mgz sub-09_run-01_fs/mri/aseg.mgz:colormap=lut
:opacity=0.5 -f sub-09_run-01_fs/surf/lh.white:edgecolor=blue sub-09_run-01_fs/s
urf/lh.pial:edgecolor=red sub-09_run-01_fs/surf/rh.white:edgecolor=blue sub-09_r
un-01_fs/surf/rh.pial:edgecolor=red')
```

The time requirement and limitations of traditional registration/segmentation of MRI data has led to DL/ML-based approaches...

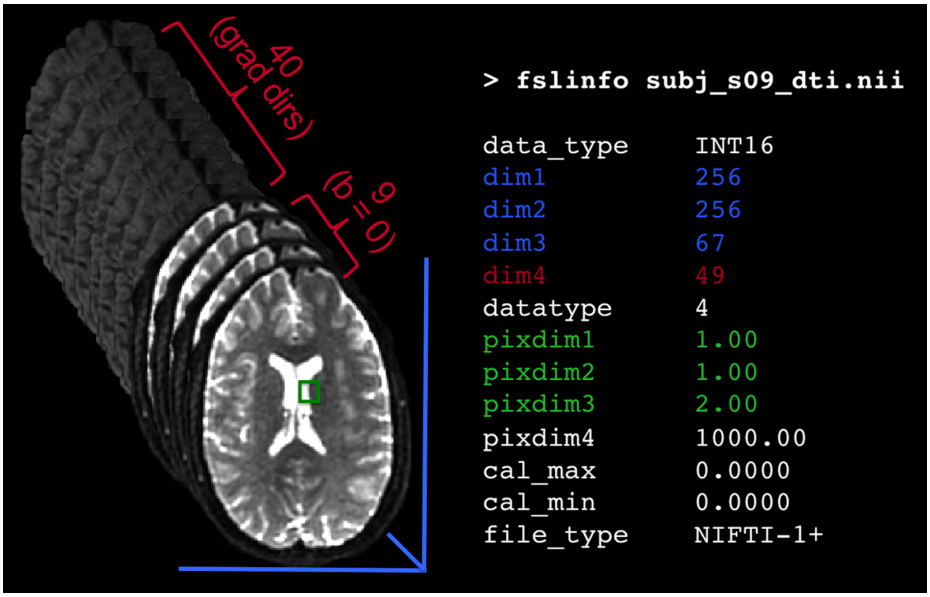
- Raj --> Matrix Decomposition
- Raunak



Encoder with outputs at each level –> split into two decoders (one for brain, one for background), also with outputs at each level

Preprocessing - DWI (Connectivity)

The key to dwi preprocessing: the b0



In [9]:
```
# Find the b0 image in our dwi nifti file
#bash_command('fsleyes ./sub-05_run-01_dwi.nii')
```

In [10]:
```
# Produce b0's for 2 or more acquisitions
# bash_command('fslroi sub-05_run-01_dwi sub-05_run-01_b0 0 1')
# bash_command('fslroi sub-05_run-02_dwi sub-05_run-02_b0 0 1')
# bash_command('fslroi sub-05_run-03_dwi sub-05_run-03_b0 0 1')
```

In [11]:
```
# merge the b0's
# bash_command('fslmerge -t sub-05_run-01-02-03_b0 sub-05_run-01_b0 sub-05_run-0
2_b0 sub-05_run-03_b0')
# bash_command('fsleyes ./sub-05_run-01-02-03_b0.nii.gz')
```

The acquisition parameters:

- Each acquisition (image) has a phase encode direction and a readout time (image acquisition)

```
"TotalReadoutTime": 0.0419756,
"PhaseEncodingDirection": "j-",
```

- Need at least two files that have opposing phase encode directions (along same axis) or different readout times

In [12]:
```
# Write file containing acquisition parameters
# bash_command('printf "0 -1 0 0.0419756\n0 -1 0 0.0419756\n 0 -1 0 0.0408244" > acq.txt')
```

FSL - topup

(Echo Planar Imaging (EPI)-Induced) Susceptibility Artifacts

External magnetic fields induce local fields!

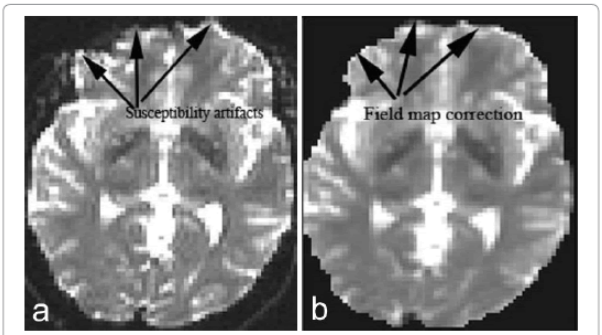


Figure 3: a) Shows susceptibility artifact in DTI data collected from one of our

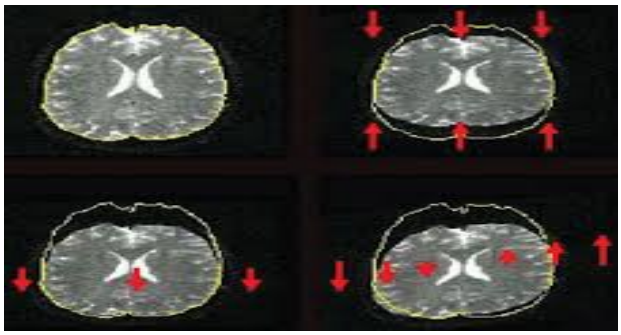
In [18]:
```
# run topup
# bash_command('topup --imain=sub-05_run-01-02-03_b0 --datain=acq.txt --config=b
02b0.cnf --out=sub-05_run-01-02-03_topup --iout=sub-05_run-01-02-03_b0_topup')

# bash_command('fsleyes sub-05_run-01-02-03_b0.nii.gz sub-05_run-01-02-03_b0_top
up.nii.gz')
```

In [17]:
```
# create mask
# bash_command('fslmaths sub-05_run-01-02-03_b0_topup -Tmean sub-05_run-01-02-03
_b0_topup')
# bash_command('bet sub-05_run-01-02-03_b0_topup sub-05_run-01-02-03_b0_brain -m
')
# bash_command('fsleyes sub-05_run-01-02-03_b0_brain.nii.gz sub-05_run-01-02-03_
b0_brain_mask.nii.gz sub-05_run-01-02-03_b0_topup.nii.gz')
```

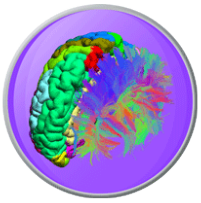
In [16]:
```
# instruct fsl which slices belong to which acquisition parameters
# bash_command('indx=""')
# bash_command('for ((i=1; i<=65; i+=1)); do indx="$indx 1"; done')
# bash_command('echo $indx > index.txt')
```

FSL - eddy

eddy currents	dwi
	

Relating ~~apples~~ to ~~oranges~~

Relating anatomy to diffusion: BrainSuite

_	_
	+ Distortion correction (remember the EPI Susceptibility artifacts we tried to fix with Topup earlier) + Elastic co-registration of dwi space to anatomical space (and vice versa)

fMRI preprocessing - not yet performed

... but this modality hasn't been completely left out ...

... more on this later ...

Now that the data is cleaned, let's prepare it for our DL algorithm ...

Forming Graphs from MR data

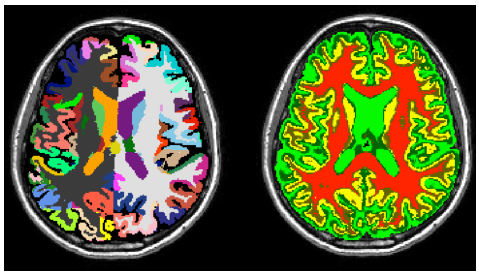
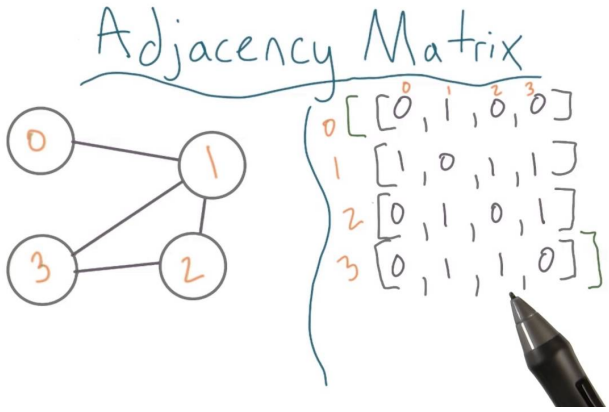
Feature representation ... Why Graphs?

The case for graph convolutional networks:

- dice coefficient / IoU as loss function (used for image segmentation) is **not sufficient** for internal structure/function representation

Structural MRI –> Brain Geometry Graph

Adjacency Matrix

–	–
	

Form an average BGG shared by all images

Structural MRI –> Brain Geometry Graph

K-Nearest Neighbor Graph

- determine ROI's across all **MRI** images
- each ROI becomes a vertex on the **Brain Geometry Graph** (BGG)
  - K-Nearest Neighbor graph
  - edges are weighted by Gaussian similarity function of Euclidean distances

$$k(x_i, x_j) = e^{\frac{||x_i - x_j||_2^2}{2\sigma^2}}$$

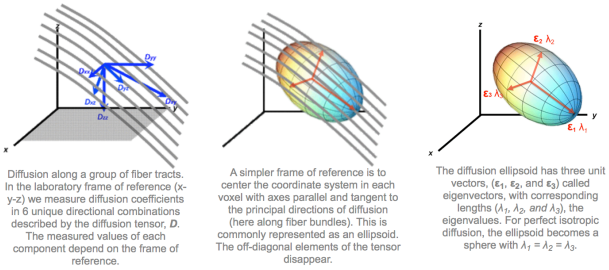
  - This yields an adjecency matrix **A** representing the similarity to nearest similar ROIs
- All subjects share the same BGG

Diffusion MRI –> Brain Connectivity Graph

First Step: Tractography

Diffusion MRI –> Brain Connectivity Graph

Basic Tractography (Deterministic) - Calculating the Diffusion Tensor, Fractional Anisotropy

_	_
$\mathcal{D} = \begin{bmatrix} D_{xx} & D_{xy} & D_{xz} \\ D_{yx} & D_{yy} & D_{yz} \\ D_{zx} & D_{zy} & D_{zz} \end{bmatrix}$ Diffusion Tensor	 Diffusion along a group of fiber tracts. In the laboratory frame of reference (x-y-z) we measure diffusion coefficients in 6 unique directional combinations described by the diffusion tensor, *D*. The measured values of each component depend on the frame of reference. A simpler frame of reference is to center the coordinate system in each voxel with axes parallel and tangent to the principal directions of diffusion (here along fiber bundles). This is commonly represented as an ellipsoid. The off-diagonal elements of the tensor disappear. The diffusion ellipsoid has three unit vectors, (ε_1, ε_2, and ε_3) called eigenvectors, with corresponding lengths (λ_1, λ_2, *and* λ_3), the eigenvalues. For perfect isotropic diffusion, the ellipsoid becomes a sphere with $\lambda_1 = \lambda_2 = \lambda_3$.

$$\mathrm{FA} = \sqrt{\frac{1}{2}} \frac{\sqrt{(\lambda_1 - \lambda_2)^2 + (\lambda_2 - \lambda_3)^2 + (\lambda_3 - \lambda_1)^2}}{\sqrt{\lambda_1^2 + \lambda_2^2 + \lambda_3^2}}.$$

Limitations: crossing/touching fibers Solution: Probabilistic Tractography --> ODF, HARDI

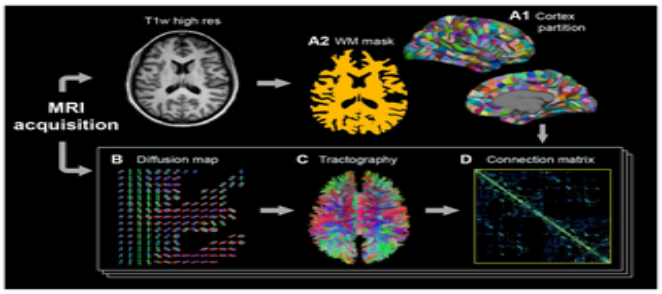
Diffusion MRI –> Brain Connectivity Graph (BCG)

Basic Tractography - "Multiple Views"

- TrackVis Diffusion Toolkit
  - Fiber Assignment by Continuous Tracking (FACT):
    - One of the first algo's (1999); still widely popular
    - Trace the principal eigenvector accross voxels until a threshold orientation transition is exceeded FA goes below a threshold value (e.g., 30 degrees, FA < 0.2)
  - Second-order Runge Kutta (RK2):
    - Differential equation: equate the tangent vector to the principal eigenvector

Diffusion MRI –> Brain Connectivity Graph (BCG)

Now we can form the graph

–	–
	+ n x n similarity matrix with all ROI's along top and side + values represent the strength of fiber connectivity between two ROI's + each tractography algorithm yields a unique matrix

Functional MRI –> Brain Network Graph (BNG)

First Step: Identifying Networks

Functional MRI –> Brain Network Graph (BNG)

First Step: Identifying Networks

Traditional Method: Independent Component Analysis

Functional MRI –> Brain Network Graph (BNG)

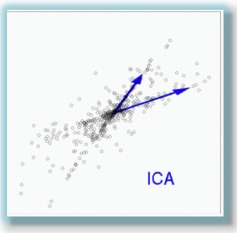
First Step: Identifying Networks

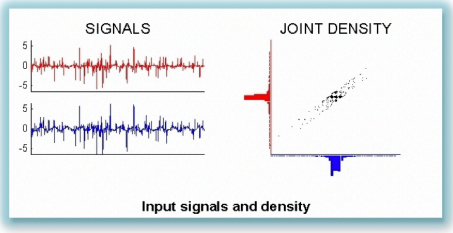
Traditional Method: Independent Component Analysis

Background
Key Concepts

Independent Component Analysis (ICA):

- Represent 2D data as linear combinations of independent source components
  - Information theoretic principles drive the estimation of the joint distributions
- Seeks to solve the Blind Source Problem (BSS)
- $Y(t) = A\,S(t)$
  - S(t) are the source signals, A is the mixing matrix
  - Find linear transformation L (ideally A^{-1}) s.t. $L\,Y(t) = S(t)$
- Popular algorithms include Infomax and FastICA





Functional MRI –> Brain Network Graph (BNG)

First Step: Identifying Networks

Traditional Method: Independent Component Analysis

Background
Key Concepts

- Spatial ICA for fMRI:
  - fMRI data has low SNR resulting in a mixture of signals
  - Follows from the localized paradigm of classical neuroscience
  - Assumes areas of interest are independently distributed
  - Research aided by software tools: Brainvoyager, FSL, DTU, GIFT

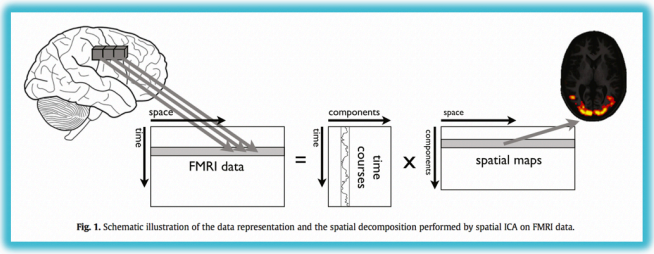


Fig. 1. Schematic illustration of the data representation and the spatial decomposition performed by spatial ICA on FMRI data.

Functional MRI –> Brain Network Graph (BNG)

First Step: Identifying Networks

Traditional Method: Independent Component Analysis

_	_

Functional MRI –> Brain Network Graph (BNG)

First Step: Identifying Networks

Newer Method: Restricted Boltzmann Machine

Functional MRI –> Brain Network Graph (BNG)

First Step: Identifying Networks

Newer Method: Restricted Boltzmann Machine, History and Background:

- Ludwig Boltzmann
- Boltzmann Distribution (1868) aka Gibb's Distribution

$$p_i \propto e^{-\frac{\varepsilon_i}{kT}}$$

- Models the probability a given "particle" is in a given state
  - probability of a Random Variable taking a given state
  - the probability distribution for a given state and sample

$$p_i = \frac{1}{Q} e^{-\varepsilon_i/kT} = \frac{e^{-\varepsilon_i/kT}}{\sum_{j=1}^{M} e^{-\varepsilon_j/kT}}$$

- Maximizes Entropy

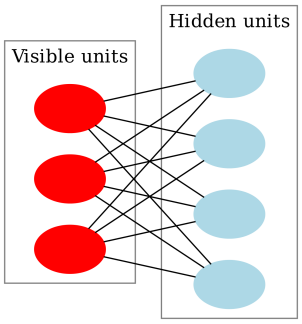
Functional MRI –> Brain Network Graph (BNG)

Functional MRI –> Brain Network Graph (BNG)

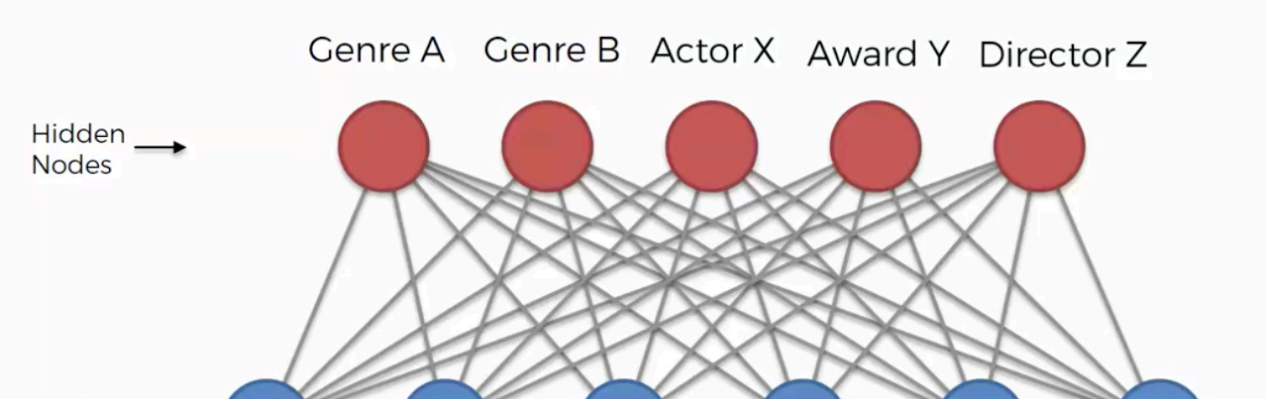
First Step: Identifying Networks

Newer Method: Restricted Boltzmann Machine

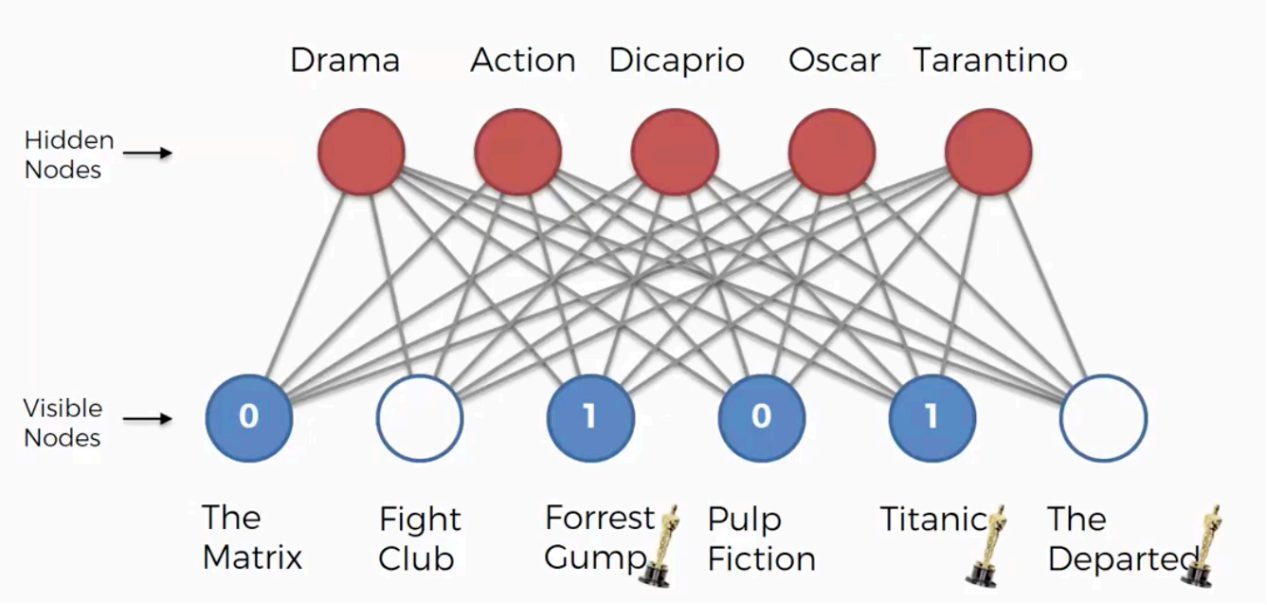
The architecture:

–	–
	+ Boltzmann Machine (fully connected, Hinton 1985) impractible/untrainable (edges grow exponentially with nodes) + RBM: Paul Smolensky in 1986 (originally termed Harmonium) + Saw larger adoption in mid-2000s after Geoff Hinton made them more easily and quickly trainable +Undirected Bipartite Graph +They are a special case of Boltzmann Machines and Markov Random Fields

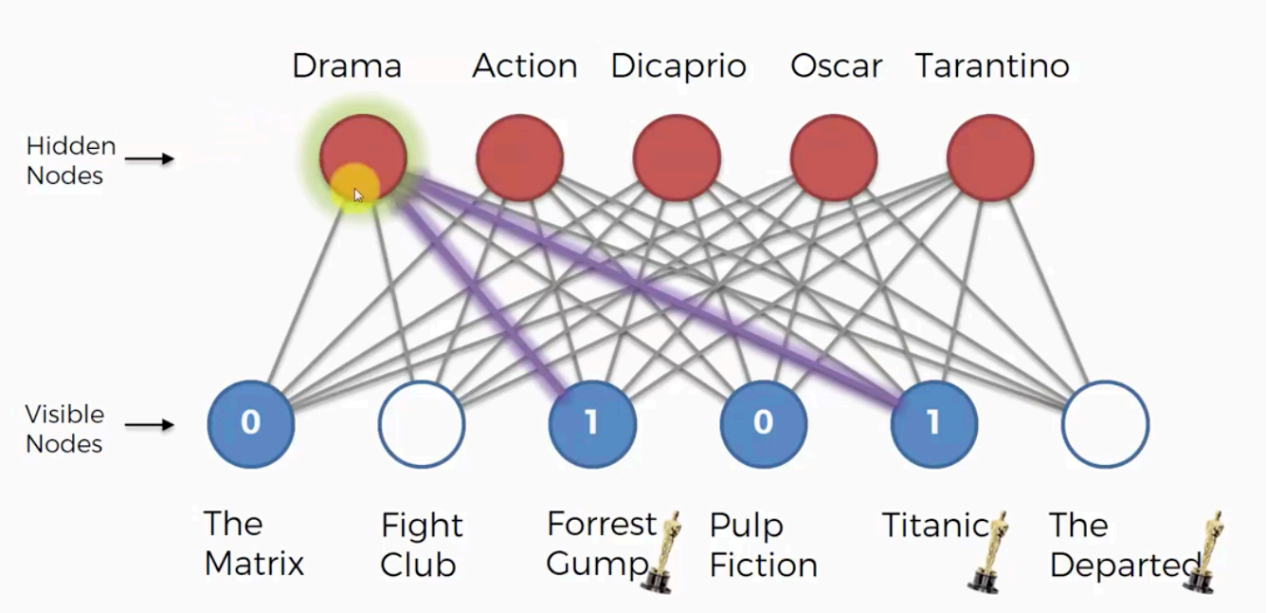
RBM Example: Trained RBM



RBM Example: New Input, 2 missing values

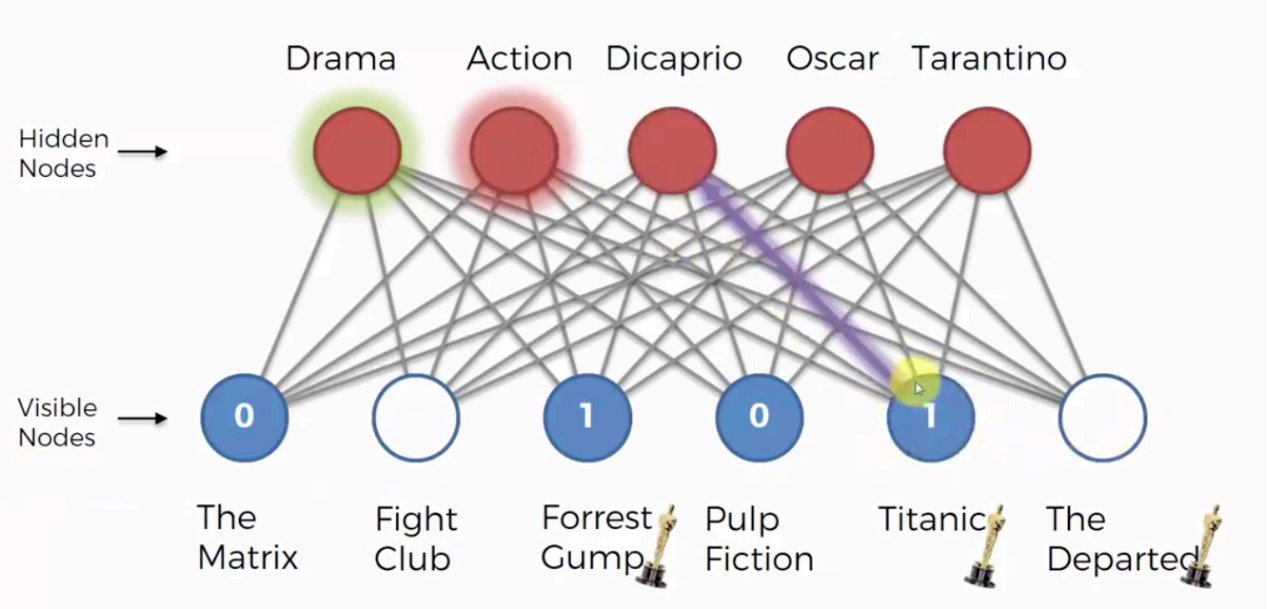


RBM Example: Compute each hidden node, Pr[h(1) | v]

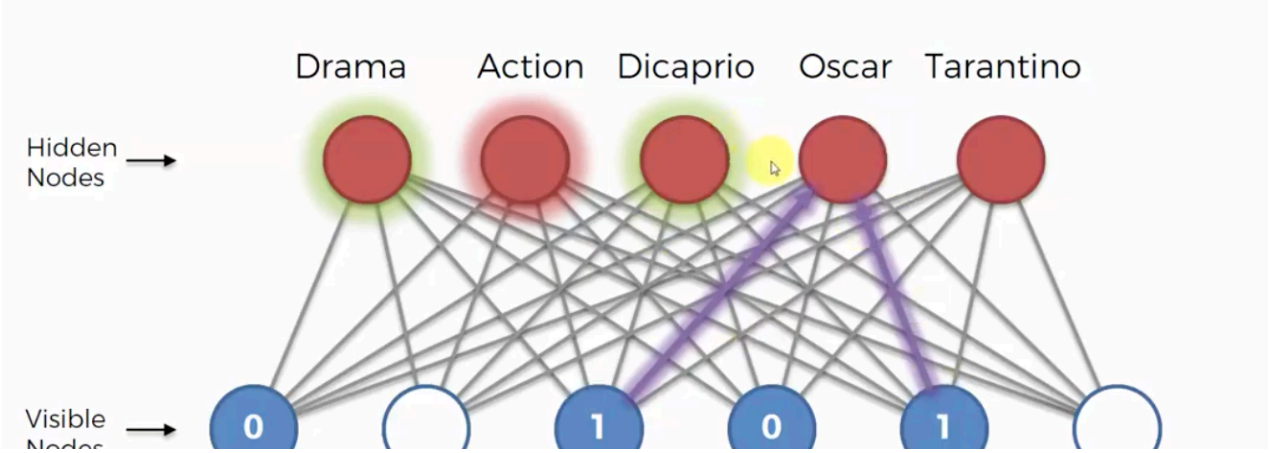


RBM Example: Compute each hidden node, Pr[h(2) | v]

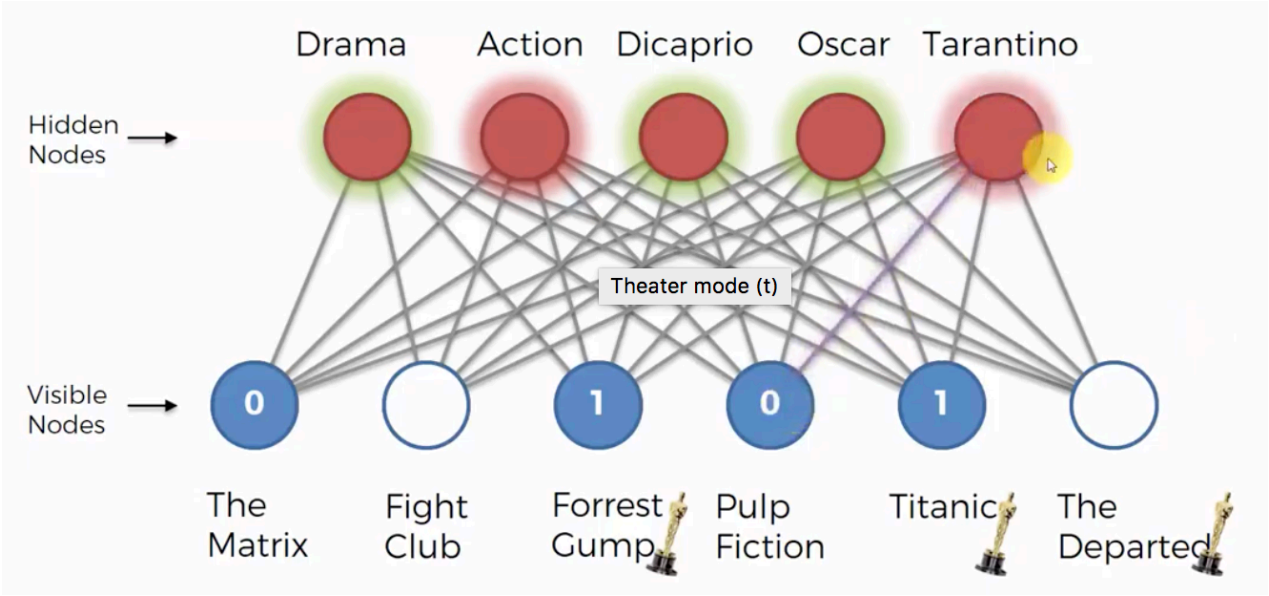
RBM Example: Compute each hidden node, Pr[h(3) | v]



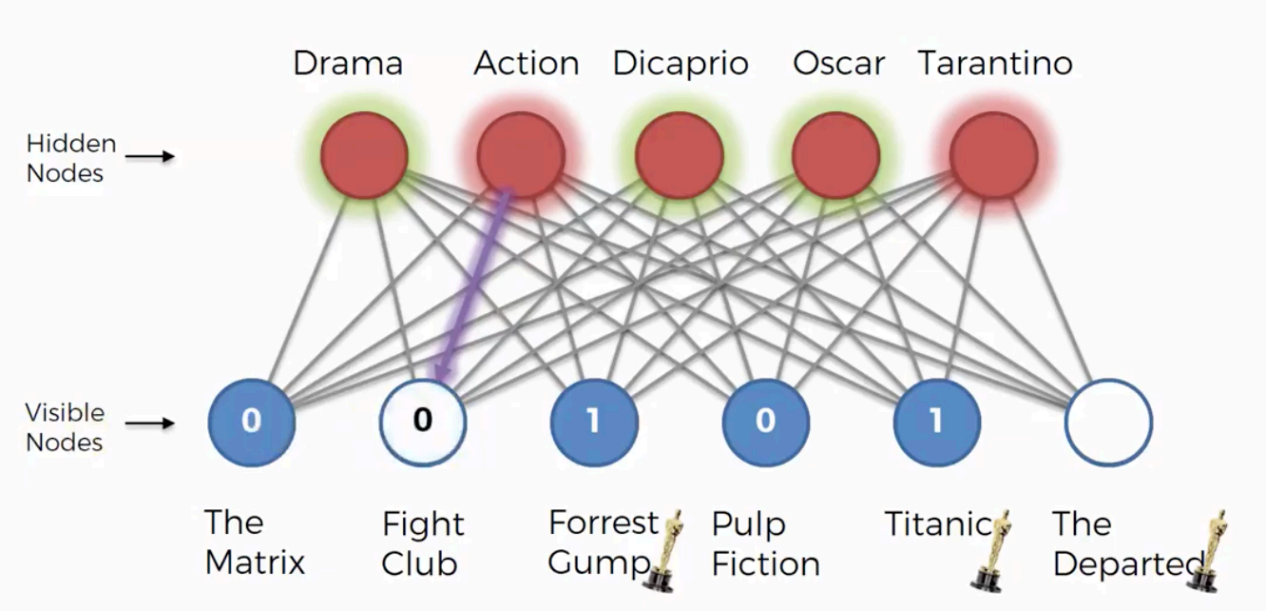
RBM Example: Compute each hidden node, Pr[h(4) | v]



RBM Example: Compute each hidden node, Pr[h(5) | v]



RBM Example: Reconstruct the input with predictions for missing values, Pr[v(2) | h]

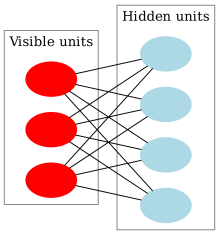


RBM Example: Reconstruct the input with predictions for missing values, Pr[v(6) | h]

Functional MRI –> Brain Network Graph (BNG)

First Step: Identifying Networks

Newer Method: Restricted Boltzmann Machine, Training:

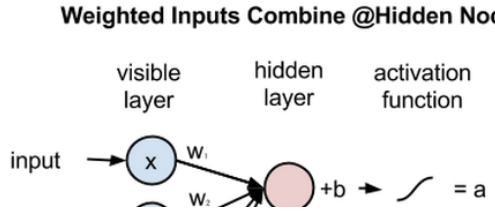
–	–
	+ The weights in the middle dictate the energy of the system + Each side becomes a Random Variable, assuming a probability distribution governed by the energy of the system w.r.t. the weights + The probabilities are formed via the Boltzmann distribution $p_i \propto e^{-\frac{\varepsilon_i}{kT}}$ $E(\vec{v},\vec{h}) = -\sum_{i=1}^{m} a_i v_i - \sum_{j=1}^{n} b_j h_j - \sum_{i=1}^{m}\sum_{j=1}^{n} v_i h_j w_{ij}$ $P(v) = \frac{1}{Z}\sum_{h} e^{-E(v,h)} \quad P(v,h) = \frac{1}{Z} e^{-E(v,h)}$ $p(v_i = 1 \mid \vec{h}) = \text{sig}\left(a_i + \sum_j h_j w_{i,j}\right)$ $p(h_j = 1 \mid \vec{v}) = \text{sig}\left(b_j + \sum_i v_i w_{ij}\right)$

Functional MRI –> Brain Network Graph (BNG)

First Step: Identifying Networks

Newer Method: Restricted Boltzmann Machine

Training: Compute Hidden State

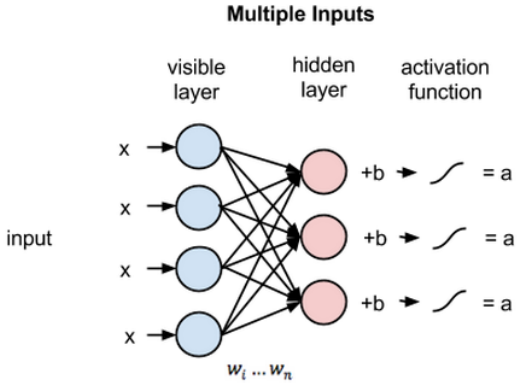
–	–
	$p(h_j = 1\|\vec{v}) = \text{sig}\left(b_j + \sum v_i w_{ij}\right)$

Functional MRI –> Brain Network Graph (BNG)

First Step: Identifying Networks

Newer Method: Restricted Boltzmann Machine

Training: Compute Hidden State

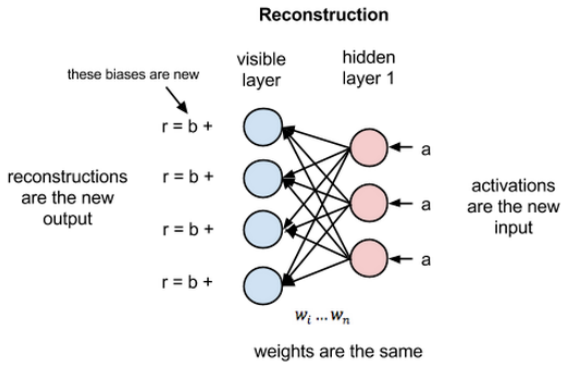
–	–
	$p(h_j = 1\|\vec{v}) = \text{sig}\left(b_j + \sum_i v_i w_{ij}\right)$ $p(h\|v) = \prod_i p(h_i\|v)$

Functional MRI –> Brain Network Graph (BNG)

First Step: Identifying Networks

Newer Method: Restricted Boltzmann Machine

Training: Compute Visible State (Reconstruction) –> Uses same wights as prev. step

–	–
	$$p(v_i = 1\|\vec{h}) = \text{sig}\left(a_i + \sum_j h_j w_{i,j}\right)$$ $$p(v\|h) = \prod_j p(v_j\|h)$$

Functional MRI –> Brain Network Graph (BNG)

First Step: Identifying Networks

Newer Method: Restricted Boltzmann Machine

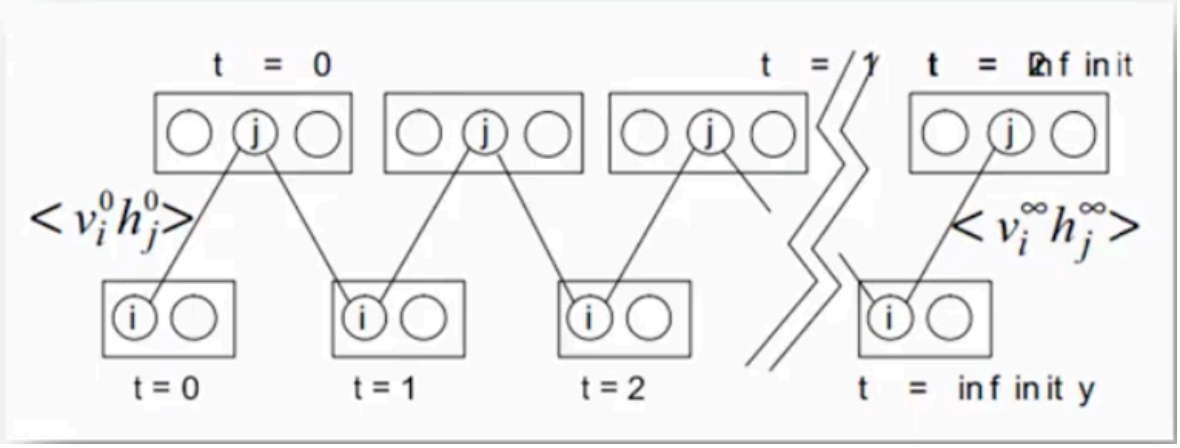
Optimization: Reconstruction-based?

Functional MRI –> Brain Network Graph (BNG)

First Step: Identifying Networks

Newer Method: Restricted Boltzmann Machine

Optimization: Reconstruction-based?

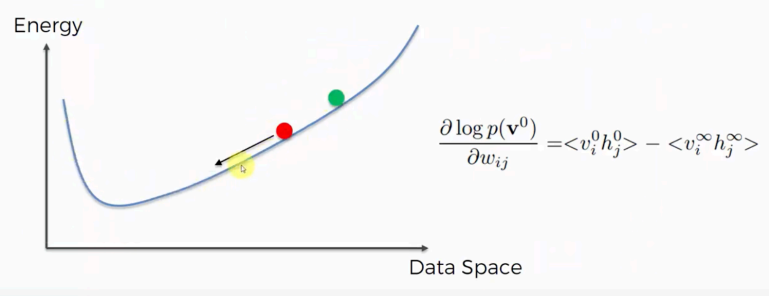


Functional MRI –> Brain Network Graph (BNG)

First Step: Identifying Networks

Newer Method: Restricted Boltzmann Machine

Optimization: Reconstruction-based?



Optimizes the energy function with the WEIGHTS fixed

Functional MRI –> Brain Network Graph (BNG)

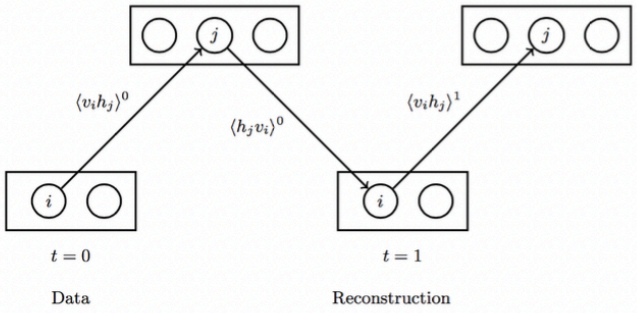
Functional MRI –> Brain Network Graph (BNG)

First Step: Identifying Networks

Newer Method: Restricted Boltzmann Machine

Training: Gibbs Sampling & Parameter Update

$$\Delta w_{ij} = \varepsilon \left(\langle v_i h_j \rangle^0 - \langle v_i h_j \rangle^1 \right);$$
$$\Delta a_i = \varepsilon \left(v_i^0 - v_i^1 \right);$$
$$\Delta b_j = \varepsilon \left(h_j^0 - h_j^1 \right);$$

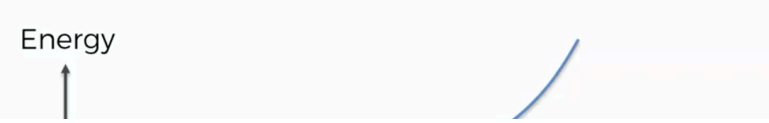


Functional MRI –> Brain Network Graph (BNG)

First Step: Identifying Networks

Newer Method: Restricted Boltzmann Machine

Optimization: Contrastive Divergence (Hinton)

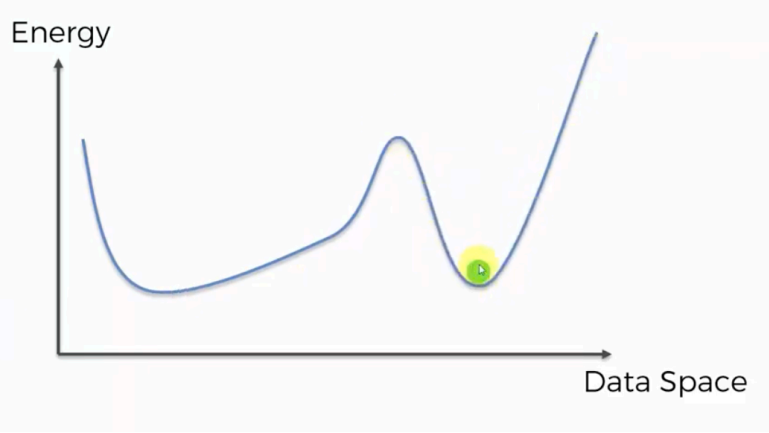


Functional MRI –> Brain Network Graph (BNG)

First Step: Identifying Networks

Newer Method: Restricted Boltzmann Machine

Optimization: Contrastive Divergence (Hinton)



So... how does this related to functional networks in the brain?

So... how does this related to functional networks in the brain?

Back to ICA –> fMRI signal is a linear combination of network signals (+ noise and background)

So... how does this related to functional networks in the brain?

Back to ICA –> fMRI signal is a linear combination of network signals (+ noise and background)

ICA is a form of Single Matrix Factorization

$$S \approx DX,$$

Back to ICA –> fMRI signal is a linear combination of network signals (+ noise and background)

ICA is a form of Single Matrix Factorization:

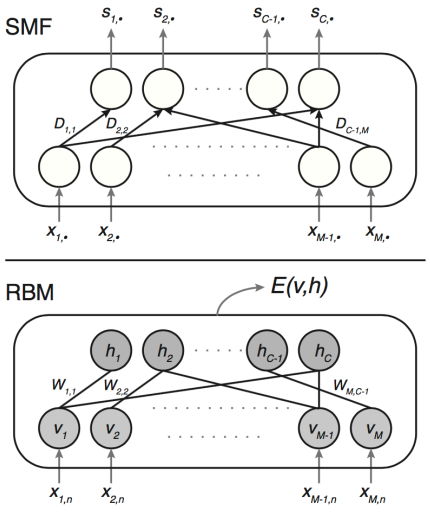
$$\mathbf{S} \approx \mathbf{D}\mathbf{X},$$

Similarly, the weights in an RBM can be seen as demixing matrices for component signals from the timeseries data:

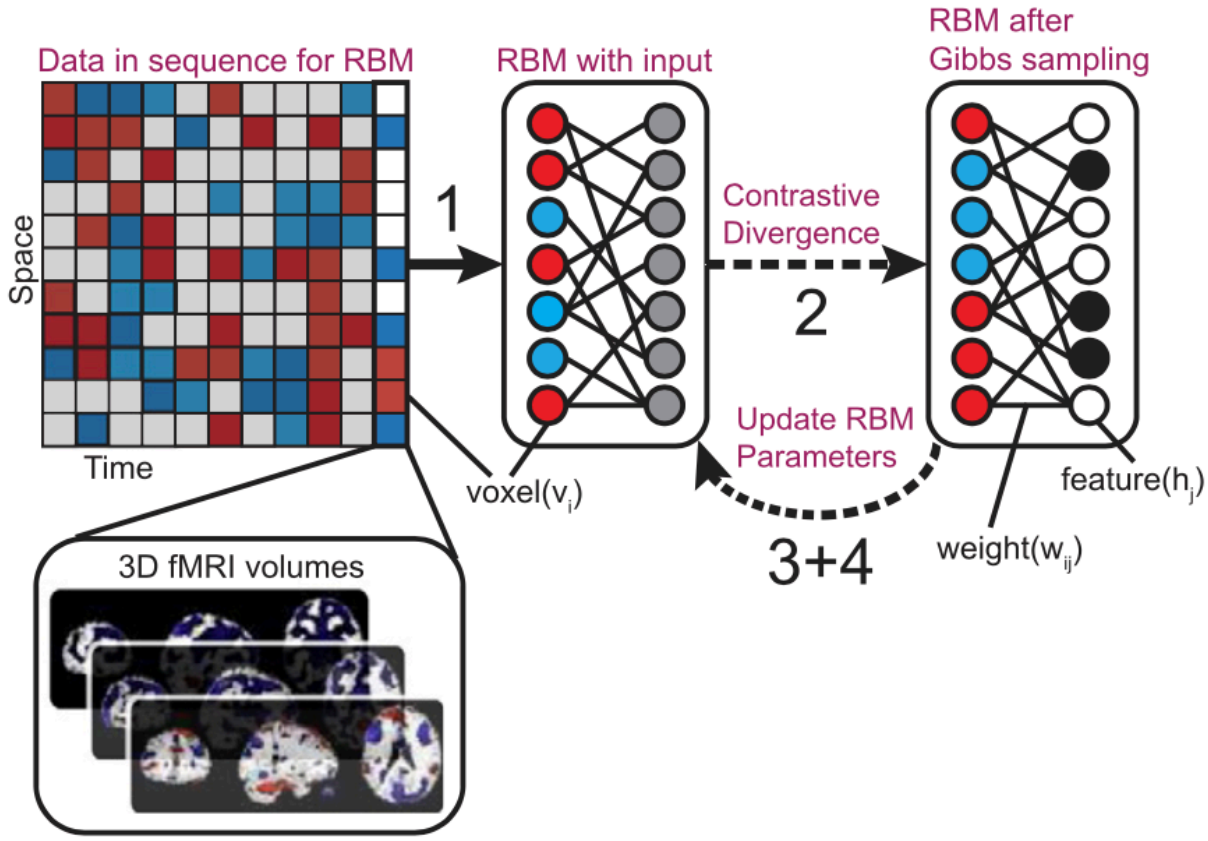
$$S = W^{\top} X,$$

So by training the RBM and extracting the weights, we can (hopefully) generate the functional networks of the data!

And can compare it to the traditional method, ICA

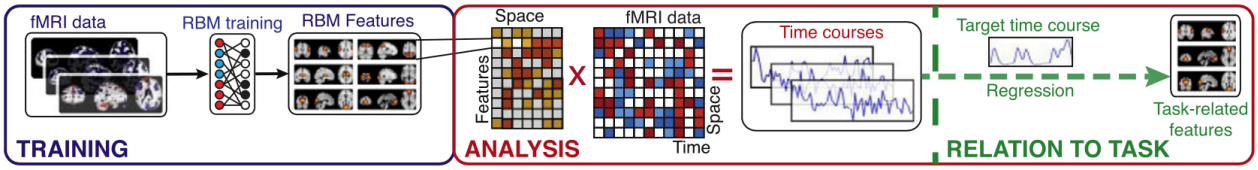


Training the RBM on fMRI data



Training the RBM on fMRI data

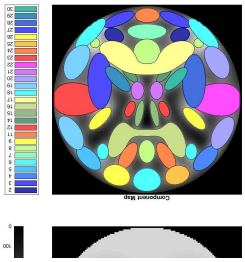
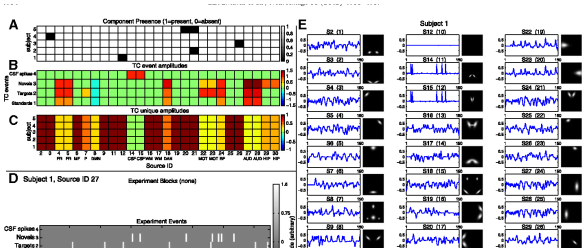
- tanh is used as the activation function instead of sigmoid
- weight normalization had the greatest influence on reconstruction performance



Training the RBM on fMRI data

First Step: Use simulation data

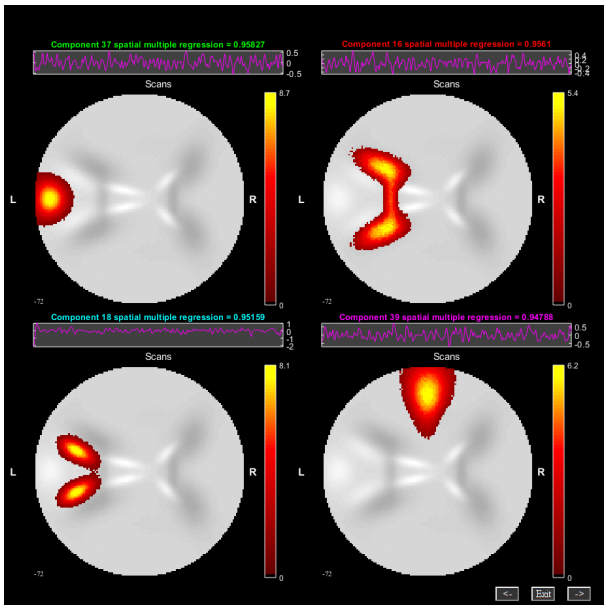
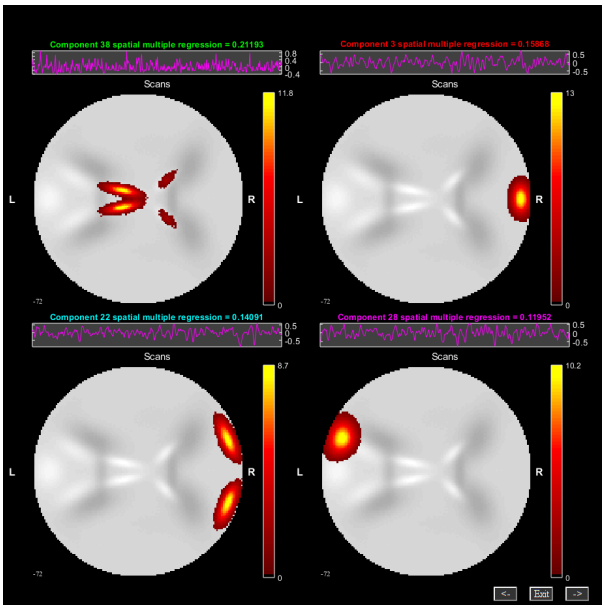
SimTB Software, MATLAB

Subject-specific component-wise simulations	Time series data
	

Training the RBM on fMRI data

First Step: Use simulation data

ICA (GIFT Toolbox, MATLAB)

–	–
	

Training the RBM on fMRI data

First Step: Use simulation data

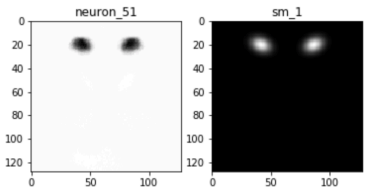
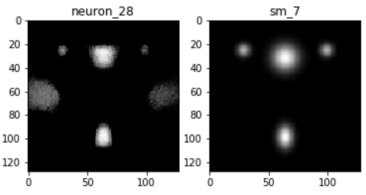
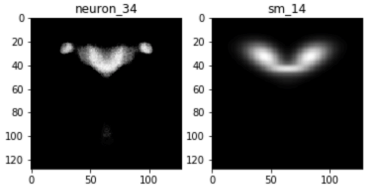
RBM Activation Maps (weights) –> Spatial correlation

–	–	–
neuron_35__sm_7 corr=0.9110710816063504	neuron_45__sm_8 corr=0.9753544236245922	neuron_54__sm_17 corr=0.9470167340201475

Training the RBM on fMRI data

First Step: Use simulation data

RBM Activation Maps (weights) –> Spatial correlation

–	–	–
neuron_51__sm_1 corr=-0.9520506120687706 	neuron_28__sm_7 corr=0.7574517598824874 	neuron_34__sm_14 corr=0.6867895643020699 

Training the RBM on fMRI data

First Step: Use simulation data

RBM Activation Maps (weights) –> Spatial correlation

26 of 27 (96.3%) of the ground truth components were captured by the RBM

(the 27th (SM 11) visually has matches, but was not captured by pearson corr.)

Training the RBM on fMRI data

26 of 27 (96.3%) correct matches are made

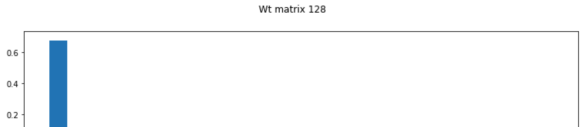
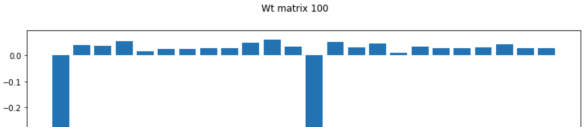
Matching Criteria:

- First,
- First, temporal correlation is used to match each single-subject activation map time course to a ground truth component time course
- Next, if the subjects for a single activation map match with different ground truth components, the ground truth component whose spatial correlation is highest with that activation map is taken as the single best match

Training the RBM on fMRI data

First Step: Use simulation data

RBM Activation Maps (weights) –> Functional Network Connectivity

–	–
	

–	–
Number of Weight matrices with spatial correlations >0.35 for two or more spatial components	22
Number of selected weights with at least partially correct FNC's	20 (91%)
Number of selected weights with fully correct FNC's	10 (45%)
Number of unique fully inaccurate FNC's	1
Number of accurately selected components	10 (62.5%)
Number of inaccurate FNC components	6 (37.5%)
Number of connected components represented	10 of 14 (71%)

Training the RBM on fMRI data

Second Step: Repeat for Piglet Data –> Piglet RSNs

Images in FSLEyes

Training the RBM on fMRI data

Second Step: Repeat for Piglet Data –> Piglet RSNs

Training the RBM on fMRI data

Next: Repeat for preprocessed fMRI PPMI data

Graphs will be formed from the resulting activation maps between ROI's and fed into the GCN

So...

What is all this for?

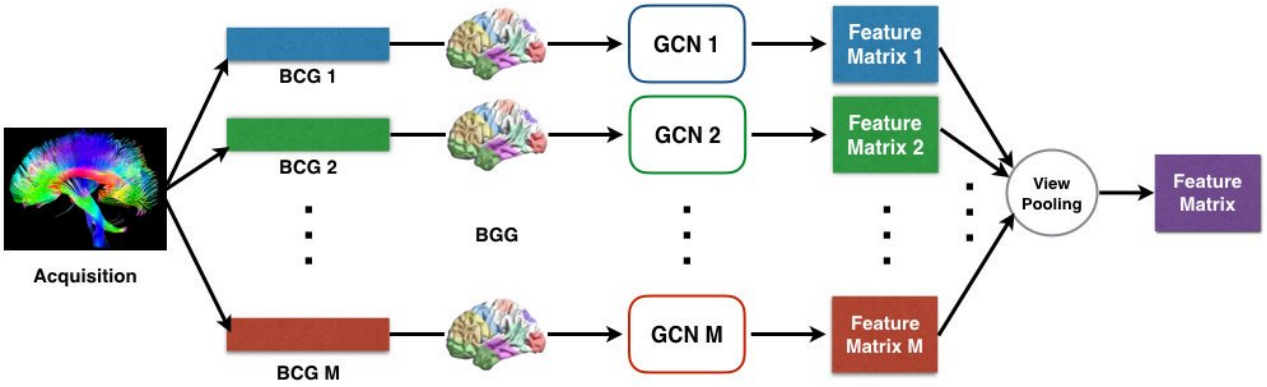
So...

What is all this for?

The Graph Convolutional Network

Feature representation:

The multi-view graph convolutional network:



Hypothesis: it would be nice to borrow from *signal processing* techniques, which have centuries' of academic rigor behind them

Hypothesis: it would be nice to borrow from *signal processing* techniques, which have centuries' of academic rigor behind them

Enter: Shuman, *et. al.* (2012)

The Emerging Field of Signal Processing on Graphs

Extending High-Dimensional Data Analysis to Networks and Other Irregular Domains

Shuman *et. al.*

The Emerging Field of Signal Processing on Graphs

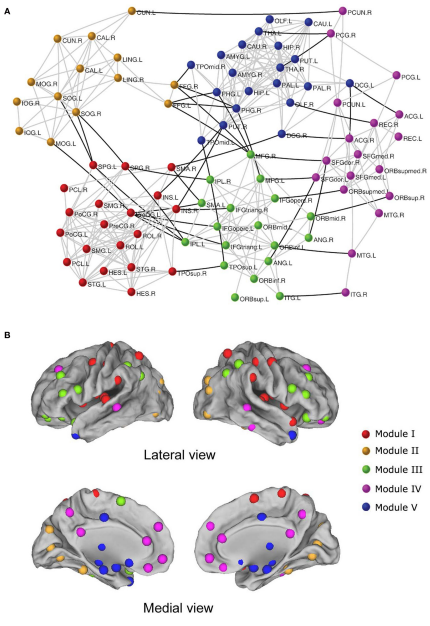
Extending High-Dimensional Data Analysis to Networks and Other Irregular Domains

Shuman *et. al.*

The goal was devised to process a graph signal as if it were a discrete-time signal

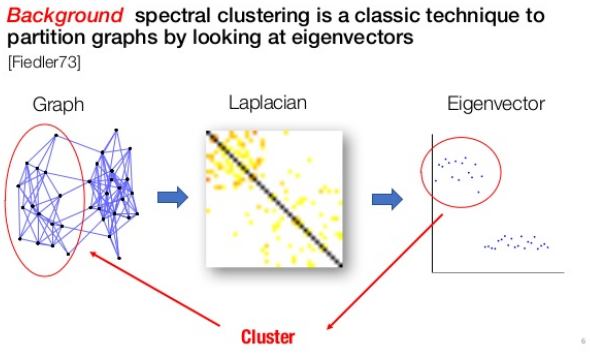
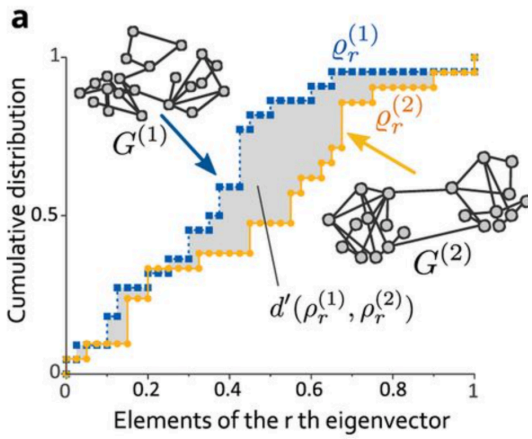
- Both a signal on a graph with *N* vertices and a classical discrete-time signal with *N* vertices can be viewed as vectors in **R**^N
- So, noisy fMRI images can be approximated as signals on weighted graphs

Defining the graph - The graph Laplacian

–	–
	**Laplacian Matrix** • Given a graph *G* with *n* vertices, its n x n Laplacian matrix L is defined as: L = D - A • L is the difference of the degree matrix *D* and the adjacency matrix *A* of the graph • Spectral graph theory studies the properties of graphs via the eigenvalues and eigenvectors of their associated graph matrices: adjacency matrix and the graph Laplacian and its variants • The most important application of the Laplacian is spectral clustering that corresponds to a computationally tractable solution to the graph partitioning problem

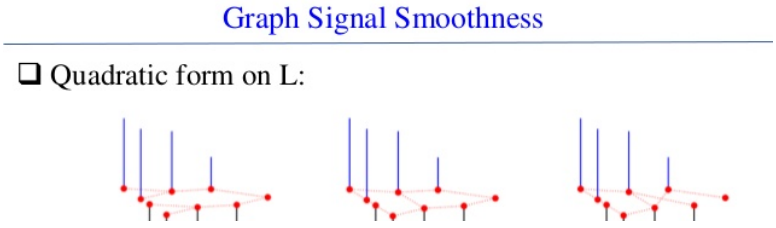
Spectral Graph Domain:

The graph Laplacian

_	_
	

Defining the graph signal

The graph signal is defined as the eigenvectors associated with the graph Laplacian

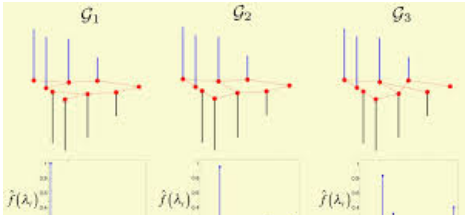


Connecting the Laplacian to the Fourier Transform

- Fourier Transform:
  - Expansion of a function in terms of the complex exponentials (i.e., cos(e) + *i* sin(e))
  - offers a discretized function for the flux of a system
- Laplace Operator (1-D):
  - The divergence of the gradient of a function (scalar notion of the gradient's flux)
  - Those same complex exponentials are the eigenfunctions of the one-dimensional Laplace operator
  - The eigenvalues 2(pi) *f* (t) carry a notion of *frequency*

Connecting the Laplacian to the Fourier Transform

- The graph FT can be defined on the vertices of G as the expansion of any function (signal) *f* in terms of the eigenvectors of the graph Laplacian
  - **L** = **U** **Λ** **U**^T
  - Let x be the signal defined on the vertices of the graph
    - The Graph Fourier Transform (GFT) is defined as x^ = **U**^T * x
    - This converts the signal x to the spectral domain spanned by the Fourier basis **U**
- Magnitude of the eigenvalue is proportional to the frequency
  - Eigenvectors associated with larger eigenvalues oscillate more rapidly between connected vertices



Chebyshev Polynomial

Chebyshev polynomials - interpolation

Chebyshev Polynomial

The coefficients c(k) *are* the Fourier coefficients a(k)

THUS, ...

... if we can learn the Chebyshev coefficients associated with the GFT of each image's signal ...

... (comprised of the Laplacian eigenvectors across connected ROI's), ...

... this can govern the convolutional operator of our CNN.

i.e., the Graph Convolutional Network

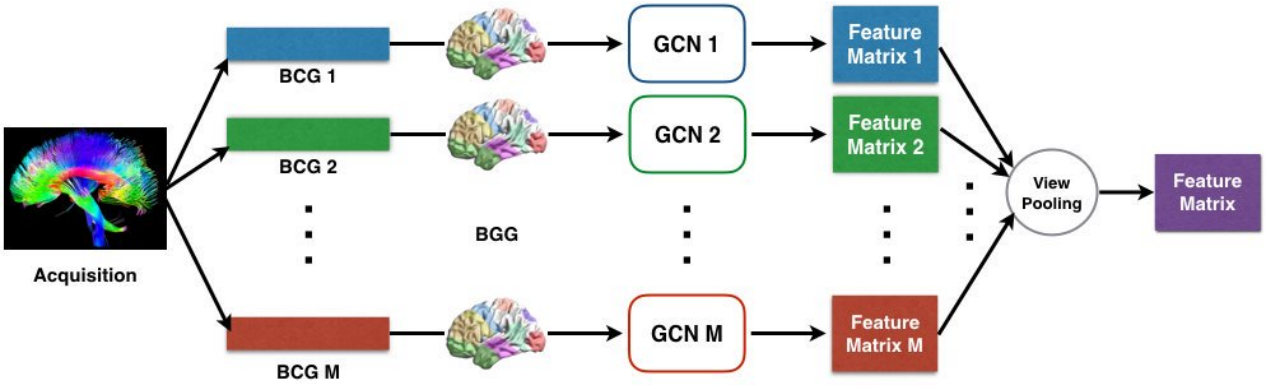
Now let's apply this to neuroimaging analysis for PD:

Current Paper:

Three Main Components	–
MV-GCN: Incorporate multiple modalities to learn a feature	

Feature representation:

The multi-view graph convolutional network:



Relationship Prediction - Graph Convolutional Networks

- Learn a feature matrix from each BCG and their shared BGG
  - Each BCG can be represented by a **vector of Chebyshev coefficients**
  - This captures the **local** traits of each individual and the **global** traits of the population
- Aggregate the feature matrices of a given image set via element-wise *view pooling*
- Train a final softmax classifier on the binary relationship between each *acquisition pair*

Graph Convolutional Network

- Conveniently, convolutions in the vertex domain become multiplication operations in the graph spectral domain
- Remembering our equations for the graph Laplacian: **L** = **U Λ U**^T
- and the GFT: x^ = **U**^T * x
- we can define the graph convolution as such:

$$\mathbf{y} = g_\theta(\mathbf{L})\mathbf{x} = g_\theta(\mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathrm{T}})\mathbf{x} = \mathbf{U}g_\theta(\mathbf{\Lambda})\mathbf{U}^{\mathrm{T}}\mathbf{x},$$

  - where θ is a vector of Fourier coefficients
  - g(θ) is the filter, a function of Λ such that, taking from our Chebyshev equation

$$f(x) \approx g_n(x) = \frac{1}{2}c_0 T_0(x) + \sum_{k=1}^{n} c_k T_k(x)$$

  we can define

$$g_\theta(\mathbf{\Lambda}) = \sum_{p=0}^{s-1} \theta_p T_p(\tilde{\mathbf{\Lambda}})$$

    - where θ(p) is a vector of Chebyshev coefficients
    - T(Λ˜) is the Chebyshev polynomial of order p evaluated at Λ˜ = 2Λ/λ(max) - **I** (diag. matrix of scaled eigenvalues)

Graph Convolutional Network

This substitution yields:

$$\mathbf{y} = g_\theta(\mathbf{L})\mathbf{x} = \sum_{p=0}^{s-1} \theta_p T_p(\tilde{\mathbf{L}})\mathbf{x},$$

$$\text{where } \tilde{\mathbf{L}} = \frac{2\mathbf{\Lambda}}{\lambda_{\max}}\mathbf{L} - \mathbf{I}$$

Graph Convolutional Network

Further:

- If we define

  $\tilde{\mathbf{x}}_p = T_p(\tilde{\mathbf{L}})\mathbf{x}$

- we see that

  $\tilde{\mathbf{x}}_i = 2\,\tilde{\mathbf{L}}\,\tilde{\mathbf{x}}_{p-1} - \tilde{\mathbf{x}}_{p-2}$

  with $\tilde{\mathbf{x}}_0 = \mathbf{x}$ and $\tilde{\mathbf{x}}_1 = \tilde{\mathbf{L}}\mathbf{x}$

- i.e.,

  $\tilde{\mathbf{x}}_p = T_p(\tilde{\mathbf{L}})\mathbf{x}$

  can be defined recursively from the normalized graph Laplacian and only the coefficients need to be learned

Graph Convolutional Network

- the jth output feature map from a GCN is given by

$$\mathbf{y}_j = \sum_{i=1}^{F_{in}} g_{\theta_{i,j}}(\mathbf{L})x_i$$

  - where x(i) is the i-th row of the input connectivity matrix (BCG) **X**
    - there are *n* rows in **X** corresponding to *n* ROI's
  - yielding F(in) x F(out) vectors of trainable Chebyshev coefficients
- each subject has a GCN output of M feature matrices
  - one feature matrix per BCG (tractography view)

View pooling:

- Multiple tractographies are aggregated together
- An element-wise maximum operation is used across all M feature matrices for a given subject
  - maximum operation combines the views' more informative features
    - instead of averaging (weakening the strongest features)
- This produces a shared feature matrix **Z** for each subject
  - giving a combined vector of Chebyshev coefficients for each ROI

Pairwise Matching Strategy

1. Compile the dataset of all pairs of feature matrices (**Z**)
   - Normalize each matrix so that the sum of squares of each row = 1
2. Define a pairwise Similarity measure:
   - if two subjects are similar (re: PD v. HC)
     - they should have a high probability of having the same class label
   -

$$\mathrm{sim}\left(\mathbf{z}_p^i, \mathbf{z}_q^i\right) = \mathbf{z}_p^{i\,T}\mathbf{z}_q^i, \quad i = 1,2,\ldots,n$$

     - where $\mathbf{z}_p^i$ and $\mathbf{z}_q^i$ are the i-th row vectors of the normalized Z matrices

Relationship Prediction

- The pairwise matching layer yields a feature vector **r**
  - each element in **r** is a row-wise similarity
- **r** is passed to a fully-connected Softmax layer for classification
  -

$$p(y = j|r) = \frac{\exp(\mathbf{w}_j^T \mathbf{r})}{\sum_{c=1}^{C} \exp(\mathbf{w}_c^T \mathbf{r})}$$

  - where w(c) is the weight vector of the *c*-th class and **r** is the final abstract representation of the input example

IN SUMMARY:

- This network has three main components:
  - Multi-View Graph Convolutional Network (MVGCN)
    - Learn a feature representation for each subject across multiple BCG's (tractography views)
      - Features obtained via the normalized graph Laplacian matrix and the graph Fourier Transform
  - Pairwise Matching Strategy
  - Softmax Relationship Prediction
- Each component is trained using backpropogation and stochastic optimization

Results (1): GCN

- Compared GCN to raw-edges weights and PCA performance for each DTI tractography algorithm
- Used same matching component and softmax component for each method

Table 1: Results for classifying matching vs. non-matching brain networks in terms of AUC metric.

Methods	Views					
	FACT	**RK2**	**SL**	**TL**	**ODF-RK2**	**Hough**
Raw Edges	58.47±4.05	62.54±6.88	59.39±5.99	61.94±5.00	60.93±5.60	64.49±3.56
PCA	64.10±2.10	63.40±2.72	64.43±2.23	62.46±1.46	60.93±2.63	63.46±3.52
FCN	66.17±2.00	65.11±2.63	65.00±2.29	64.33±3.34	68.80±2.80	61.91±3.42
FCN_{2l}	82.36±1.87	81.02±4.28	81.68±2.49	81.99±3.44	82.53±4.74	81.77±3.74
GCN	**92.67±4.94**	**92.99±4.95**	**92.68±5.32**	**93.75±5.39**	**93.04±5.26**	**93.90±5.48**

Results (2): MVGCN

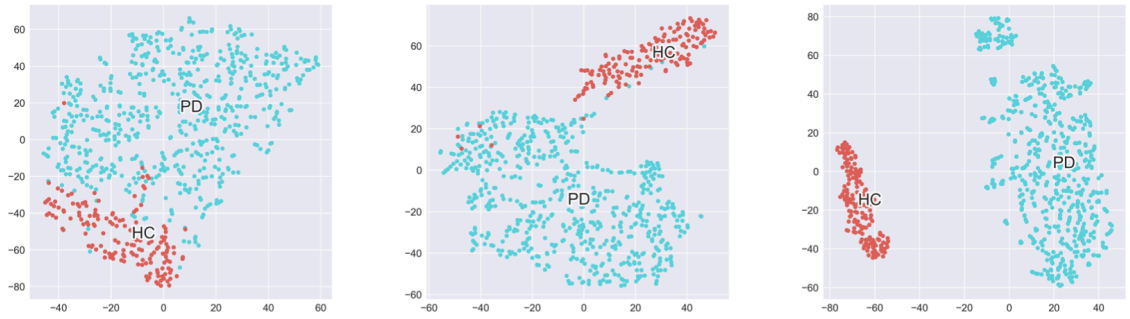
- Compared the clustering abilities of the MVGCN (re: PD vs. HC)
  - Clustering performance measured via Normalized Mutual Information (NMI)

Table 2: Comparison of binary classification (AUC) and acquisition clustering (NMI) results using both single-view and multi-view architectures.

Architectures	AUC	NMI
PCA100-M-S	64.43±2.23	0.39
FCN1024-M-FCN64-S	82.53±4.74	0.87
GCN128-M-S	93.75±5.39	0.98
MVGCN128-M-S_{mean}	94.74±5.62	1.00
MVGCN128-M-S_{max}	**95.37±5.87**	1.00

Results (3): Visualization - Binary Similarity

- used the relationship prediction generated from the various models to map all 754 DTI acquisitions distanced by their predicted similarity

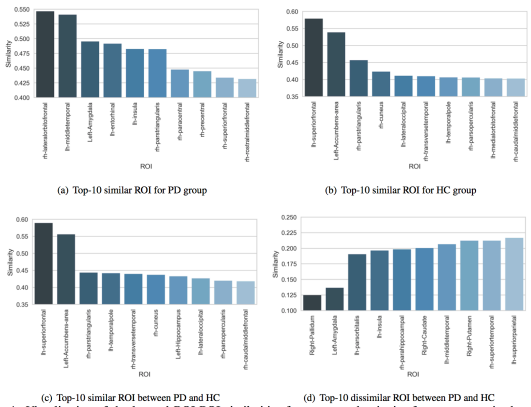


(a) PCA (b) FCN_{2l} (c) MVGCN

Figure 3: Visualization of the DTI acquisition clusters. The acquisitions are mapped to the 2D space using the t-SNE algorithm with the predicted values of pairwise relationship as input. Blue denotes PD, Red denotes HC.

Results (4): Visualisation - ROI Similarity

- The MVGCN output consists of ROI-wise pairwise similarity
- Able to visualize the 10 most similar or dissimilar ROIs for PD vs. HC



(a) Top-10 similar ROI for PD group

(b) Top-10 similar ROI for HC group

(c) Top-10 similar ROI between PD and HC

(d) Top-10 dissimilar ROI between PD and HC

Figure 4: Visualization of the learned ROI-ROI similarities from averaged pairwise feature vectors in the certain groups. Top-10 similar or dissimilar ROIs for PD and HC groups and the corresponding values are shown in (a)-(d) respectively.

- Key Findings:
  - Lateral orbitofrontal area, middle temporal and amygdala are three most similar ROIs for PD
  - Caudate and putamen are discriminative b/t Pd and HC

Discussion/ Conclusions:

- MVGCN allows modeling multiple brain connectivity networks (BCGs) and brain geometry graphs (BGG) based on common ROIs and tractography algorithms
  - BCGs are non-Euclidean, so standard convolution is not straightforward to use and must be specifically defined
- Multi-view approach allows the exploration of various aspects of the data
- Pairwise method increases the size of the dataset
- Straightforward interpretations of the networks were learned