A tutorial for fMRI analysis using ANTsR

Introduction

Overview

fMRI issues

- Nuisance signal from CSF and WM [1]
- Bandpass filtering
- Motion correction [2, 3]
- Global signal [4]

ANTsR implementation

The main fMRI-specific functions are:

- `fMRINormalization` — This function leverages structural image processing based on ANTs cortical thickness to implement standard functional image processing recommendations. The function will crop out the first k time frames, do motion correction and produce a variety of nuisance regressors. It will also do spatial and temporal filtering as well as interpolation between time frames that exceed a given framewise displacement. Finally, we return maps to the common coordinate system. Output may be trimmed in the future but currently provides access at different stages: merging versus filtering and normalized, fused BOLD images in both subject and template space.
- `preprocessRestingBOLD` — Preprocess resting fMRI by performing compcor/motion correction, nuisance regression, band-pass filtering, and spatial smoothing.
- `antsMotionCalculation` — Corrects 4D time-series data for motion and returns useful information, e.g., nuisance variables such as framewise displacement and DVARS. Uses `antsMotionCorr` for performing motion correction.
- `timeseries2matrix/matrix2timeseries` — Converts from/to a 4-D `antsImage` object to/from a $n \times p$ matrix where n is the number of time points and p is the number of voxels in the specified mask.
- `antsSpatialICAfMRI` — Perform spatial ICA on group or individual fMRI data.
- `frequencyFilterfMRI` — performs band-pass filtering on BOLD time series data.
- `getfMRInuisanceVariables` — Older, Black box-type function which performs motion correction, runs CompCor and estimates global signal (similar to `preprocessRestingBOLD`). The output is a list with the time series data matrix (time by voxels), motion and other nuisance variables, global signal, the mask and the average time series image.
- `filterfMRIforNetworkAnalysis` — to be deprecated.
- `preprocessfMRI` — to be deprecated (redundant functionality overlap with `preprocessRestingBOLD`).

Helper functions include:

- `matrixToImages/imagesToMatrix` — Converts to/from a $n \times p$ matrix from/to a list of `antsImage` where n corresponds to the number of images and n is equal to the number of voxels inside the specified mask.
- `icaWhiten` — Performs just the ica whitening step. This is useful for testing other decomposition algorithms performance when they use the same prewhitening step as ICA.
- `makePointsImage` – Creates spherical points in the coordinate space of the target image based on the n-dimensional matrix of points and a specified radius. Used with data such as `powers_areal_mni_itk`.

Data:

- `powers_areal_mni_itk` — Defines the set of nodes describing the functional organization of the brain as detailed in [5] and made publicly available.[1]

fMRI papers which use ANTsR

- *The pediatric template of brain perfusion* [6]
- *Subject-specific functional parcellation via prior based eigenanatomy* [7]
- *Unexpected role of interferon-γ in regulating neuronal connectivity and social behaviour* [8]

[1] http://www.nil.wustl.edu/labs/petersen/Resources.html

Tutorial

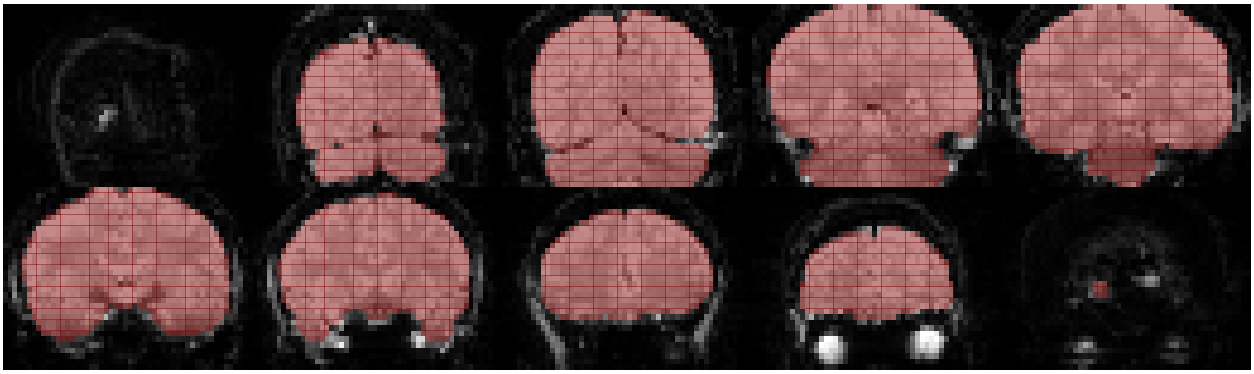
Initialization

```
# We include all the necessary R package dependencies.  We assume that the user
# is running this script (stitchTutorialDocument.R) in the repo directory.  The
# following R packages are required:
#   * ANTsR: image I/O, fMRI processing
#   * ggplot2: plot generation
#   * igraph:  network connectivity measures
#   * psych:  network connectivity measures
#   * corrplot:  correlation plot generation

invisible( suppressMessages( library( ANTsR ) ) )
library( pander )
library( ggplot2 )
library( igraph )
library( psych )
library( corrplot )

rootDirectory <- "./"
knitr::opts_knit$set( root.dir = rootDirectory )
knitr::opts_chunk$set( comment = "" )

figuresDirectory <- paste0( rootDirectory, "Figures/" )
if( ! dir.exists( figuresDirectory ) )
  {
  dir.create( paste0( rootDirectory, "Figures/" ) )
  }
dataDirectory <- paste0( rootDirectory, "Data/" );
```

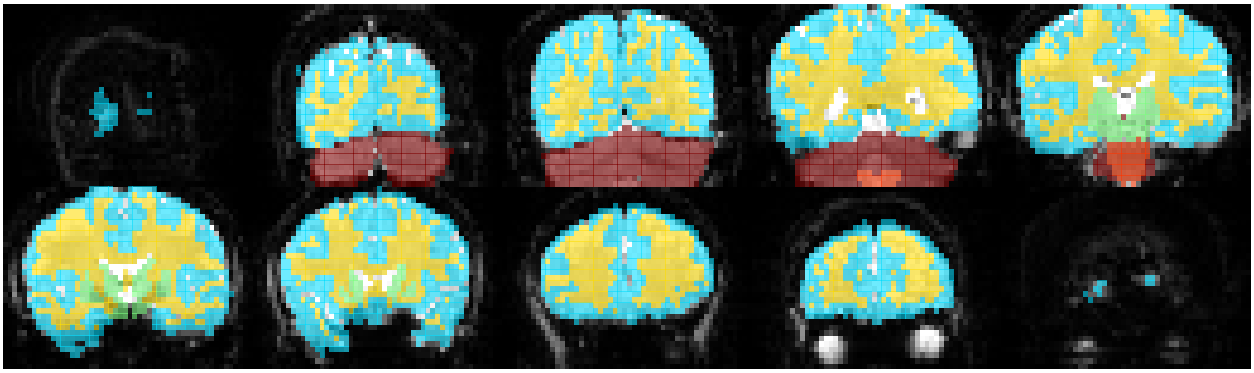
Read in input data

```
# Load the AAL (Automated Anatomical Labeling) data table and the AAL label image.
# These data define the AAL atlas of the human brain often used in fMRI for obtaining
# the anatomical regions of interest:
#    N. Tzourio-Mazoyer; B. Landeau; D. Papathanassiou; F. Crivello; O. Etard; N.
#    Delcroix; Bernard Mazoyer & M. Joliot (January 2002). "Automated Anatomical
#    Labeling of activations in SPM using a Macroscopic Anatomical Parcellation
#    of the MNI MRI single-subject brain". NeuroImage. 15 (1): 273-289.
#
# Also load the individual subject resting state BOLD images:  4-D bold, 3-D bold
# mask image, and 3-D segmentation (csf, gm, wm, etc.) image.

data( aal, package = 'ANTsR' )
aalLabelTable <- aal
aalFileName <- paste0( dataDirectory, "aal.nii.gz" )
aalImage <- antsImageRead( filename = aalFileName, dimension = 3 )

restingStateBoldFile <- paste0( dataDirectory, "rsbold.nii.gz" )
```

```
restingStateBoldImage <- antsImageRead( restingStateBoldFile, dimension = 4 )

restingStateBoldMaskFile <- paste0( dataDirectory, "rsboldmask.nii.gz" )
restingStateBoldMaskImage <- antsImageRead( restingStateBoldMaskFile, dimension = 3 )

restingStateBoldSegFile <- paste0( dataDirectory, "rsboldseg.nii.gz" )
restingStateBoldSegImage <- antsImageRead( restingStateBoldSegFile, dimension = 3 )

# Let's look at the images to make sure things make sense, e.g. masks are aligned.
#  Average of 4-D bold with mask superimposed

restingStateBoldAverage <- getAverageOfTimeSeries( restingStateBoldImage )
invisible( plot.antsImage( restingStateBoldAverage, restingStateBoldMaskImage,
   alpha = 0.75, ncolumns = 5 ) )
```

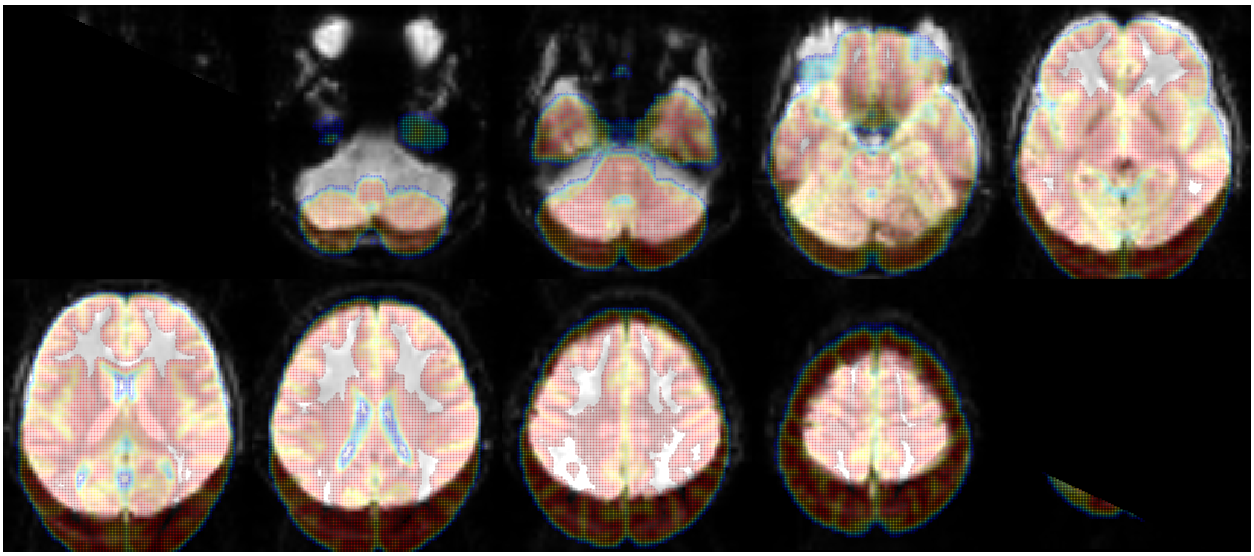


```
# Average of 4-D bold with segmentation mask superimposed
invisible( plot.antsImage( restingStateBoldAverage, restingStateBoldSegImage,
   alpha = 0.9, ncolumns = 5 ) )
```



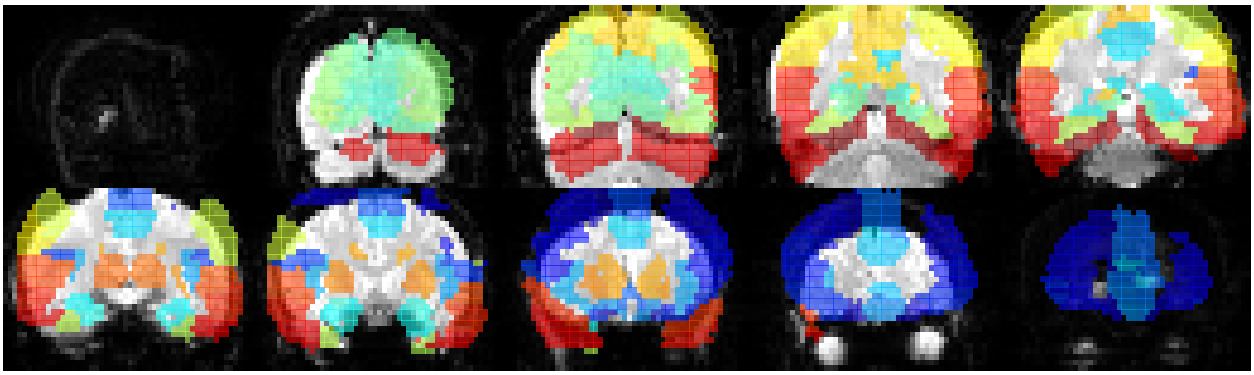
```
# Another possible use of ``plot.antsImage`` is to map the input image to another
# image of known/standard orientation, e.g., MNI space.  To do this, we read in the
# MNI template (see http://www.lead-dbs.org/?p=1241) and do a quick registration.

mniFile <- paste0( dataDirectory, "/mni.nii.gz" )
mniImage <- antsImageRead( mniFile, dimension = 3 )

mniRegistration <- antsRegistration( fixed = mniImage,
                      moving = restingStateBoldAverage, typeofTransform = "QuickRigid",
```

```
                        outprefix = paste0( dataDirectory, "mnixrs" ) )

invisible( plot( restingStateBoldAverage, mniImage, ncolumns = 5, axis = 3, alpha = 0.25,
   domainImageMap = c( mniImage, mniRegistration$fwdtransforms ) ) )
```

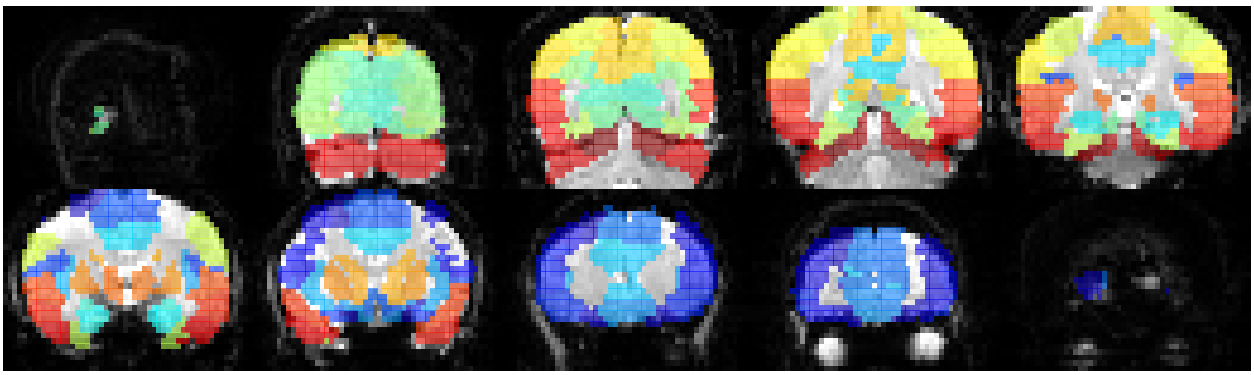


Spatially normalize AAL image

```
# The AAL image (which will be used later in the tutorial) is not in the space of the
# BOLD image so we do a quick registration of the AAL labels to the bold mask.  We
# first do an "AffineFast" transform to see if that has sufficient degrees of freedom.

aalRegistration <- antsRegistration( fixed = restingStateBoldMaskImage,
                        moving = aalImage, typeofTransform = "AffineFast",
                        outprefix = paste0( dataDirectory, "rsboldxaal" ) )

aalWarpedImage <- antsApplyTransforms( fixed = restingStateBoldMaskImage,
                        moving = aalImage, interpolator = 'genericLabel',
                        transformlist = aalRegistration$fwdtransforms )

invisible( plot.antsImage( restingStateBoldAverage, aalWarpedImage,
   alpha = 0.9, ncolumns = 5 ) )
```



```
# Clearly the alignment is not as good as we would like so we redo the registration
# using an "ElasticSyN" transform which looks much better.

aalRegistration <- antsRegistration( fixed = restingStateBoldMaskImage,
                      moving = aalImage, typeofTransform = "ElasticSyN",
                      outprefix = paste0( dataDirectory, "rsboldxaal" ) )

aalWarpedImage <- antsApplyTransforms( fixed = restingStateBoldMaskImage,
                      moving = aalImage, interpolator = 'genericLabel',
                      transformlist = aalRegistration$fwdtransforms )

invisible( plot.antsImage( restingStateBoldAverage, aalWarpedImage,
   alpha = 0.9, ncolumns = 5 ) )
```



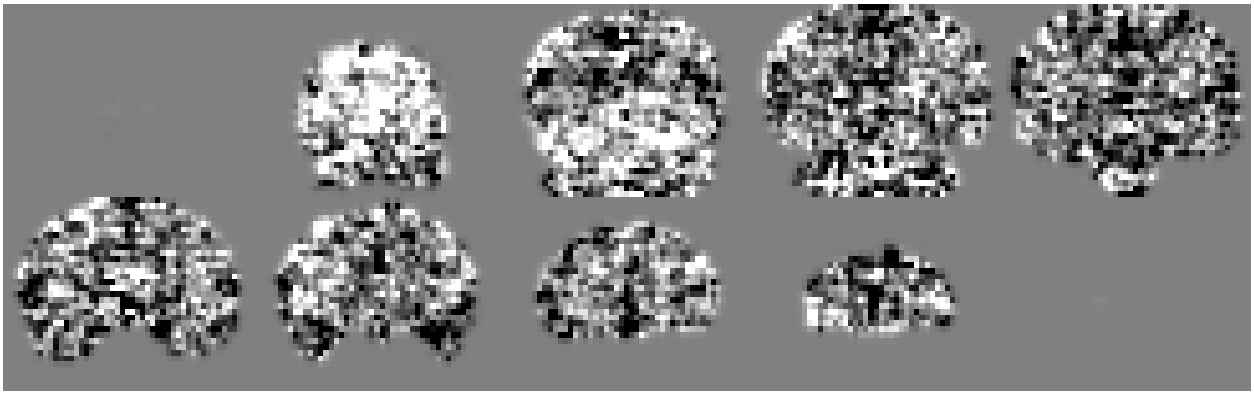
Preprocessing the resting state fMRI data

```
# The evolution of fMRI functionality in ANTsR is still ongoing. We process our
# current subject with ``preprocessRestingBOLD`` and plot the average of the
# resulting processed fMRI.  Note that we're doing motion correction on the lowest
# accuracy level for tutorial purposes.  For actual data, one would probably
# want to increase the accuracy level.

preprocessedRestingState <-
  preprocessRestingBOLD( restingStateBoldImage,
                         maskImage = restingStateBoldMaskImage,
                         denseFramewise = FALSE, numberOfCompCorComponents = 6,
                         doMotionCorrection = TRUE, motionCorrectionAccuracyLevel = 0,
                         motionCorrectionIterations = 1, frequencyLowThreshold = 0.01,
                         frequencyHighThreshold = 0.1,
                         spatialSmoothingType = "gaussian",
                         spatialSmoothingParameters = 2 )

pander( summary( preprocessedRestingState ), style = "rmarkdown",
  caption = "Returned values from the function preprocessRestingBOLD." )
```

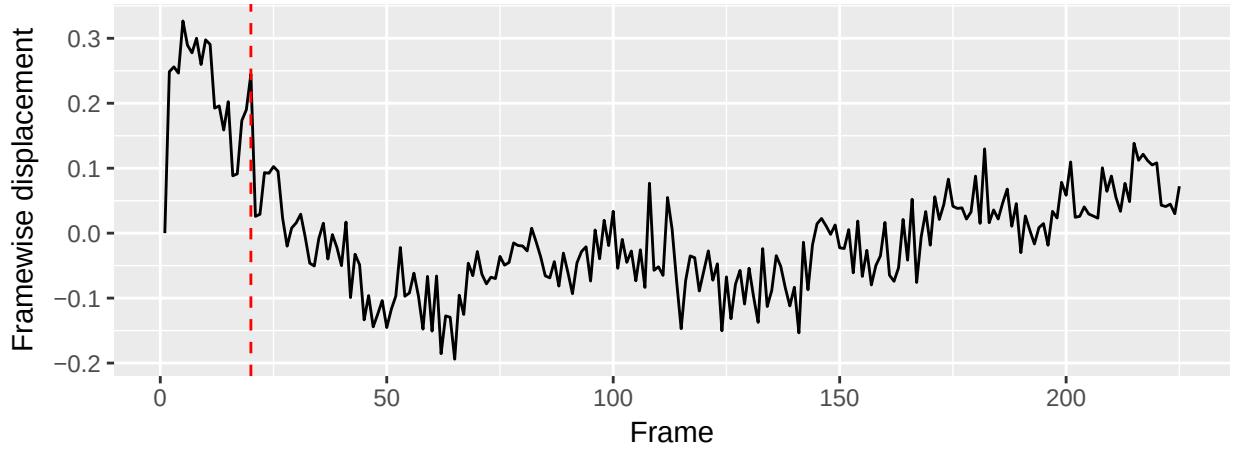
Table 1: Returned values from the function preprocessRestingBOLD.

	Length	Class	Mode
cleanBoldImage	1	antsImage	S4
maskImage	1	antsImage	S4
DVARS	225	-none-	numeric
DVARSpostCleaning	225	-none-	numeric
FD	225	-none-	numeric
globalSignal	225	-none-	numeric
nuisanceVariables	1350	-none-	numeric

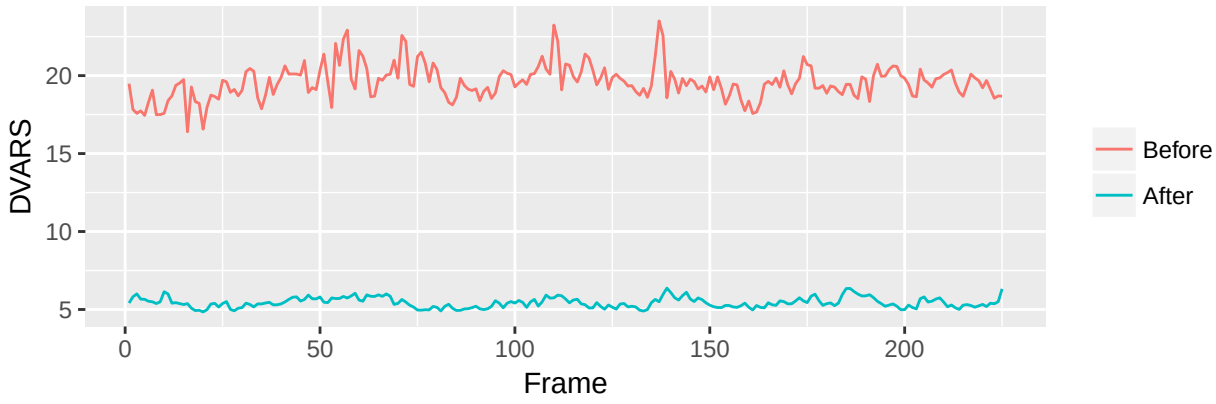
```
invisible( plot.antsImage(
   getAverageOfTimeSeries( preprocessedRestingState$cleanBoldImage ), ncolumns = 5 ) )
```



```
# We continue to check the preprocessing by plotting:
#   1. the framewise displacement (FD)
#   2. the global signal before and after regression (globalSignal)
#   3. comparing the DVARS of the original data (DVARS) and the processed
#        data (DVARSpostCleaning)
#   4. Plot the CompCor nuisance variables

numberOfTimeFrames <- dim( restingStateBoldImage )[4]

# Plot the framewise displacement.  Note the extreme displacements during the
# initial acquisition.  This is common and preprocessing often involves discarding
# the first N frames.

fdDataFrame <- data.frame( Frame = 1:numberOfTimeFrames,
   FD = preprocessedRestingState$FD - mean( preprocessedRestingState$FD ) )

ggplot( fdDataFrame ) +
   geom_line( aes( x = Frame, y = FD ), size = 0.5 ) +
   geom_vline( xintercept = 20, linetype = "dashed", size = 0.5, color = "red" ) +
   xlab( "Frame" ) + ylab( "Framewise displacement" ) +
   theme( legend.title = element_blank() ) + theme( aspect.ratio=1/3 )
```

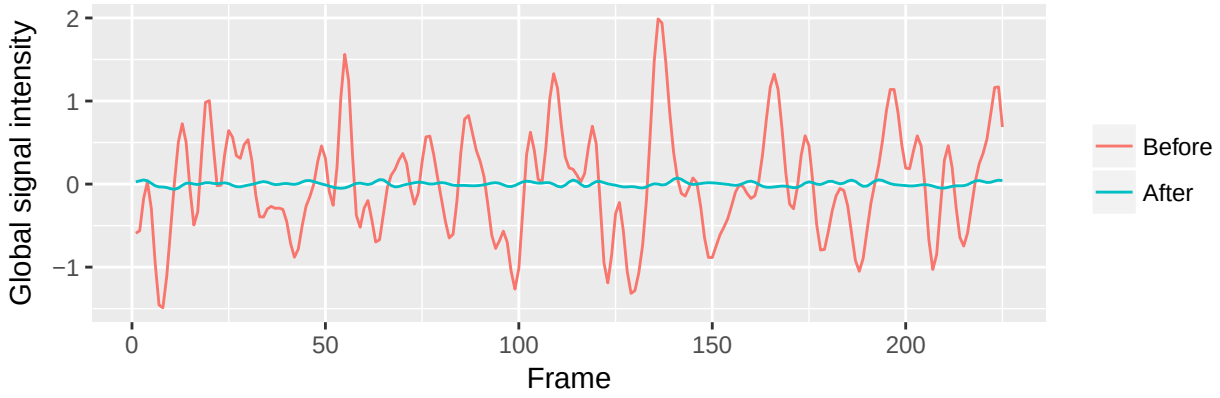



```
# Plot the DVARS.  Defined as the framewise backwards RMS voxelwise difference averaged
# over each time frame.

dvarsDataFrame <- data.frame( Frame = rep( 1:numberOfTimeFrames, 2 ),
       DVARS = c( preprocessedRestingState$DVARS,
                  preprocessedRestingState$DVARSpostCleaning ),
       Type = factor( c( rep( "Before", numberOfTimeFrames ),
              rep( "After", numberOfTimeFrames ) ), levels = c( "Before", "After" ) ) )

ggplot( dvarsDataFrame ) +
   geom_line( aes( x = Frame, y = DVARS, colour = Type ), size = 0.5 ) +
     xlab( "Frame" ) + ylab( "DVARS" ) +
     theme( legend.title = element_blank() ) + theme( aspect.ratio = 1/3 )
```

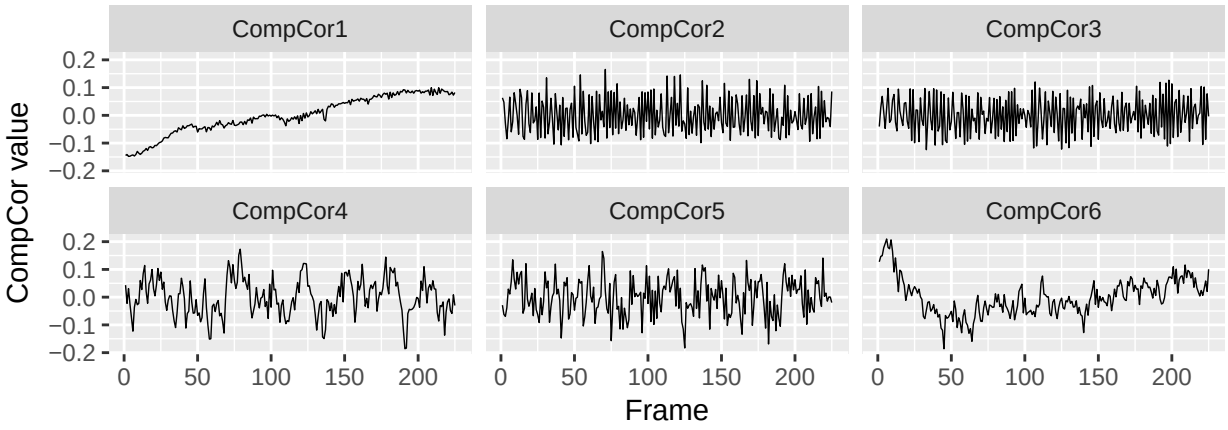



```
# Plot the global signal.  Do we regress out the global signal? Still an open issue.
# Let's just explore the approach to regressing it out afterwards.  A better way would
# be to include it as an ``initialNuisanceVariable`` in ``preprocessRestingBOLD()``.

boldMatrix <- timeseries2matrix(
  preprocessedRestingState$cleanBoldImage, restingStateBoldMaskImage )
boldMatrixGlobalSignalRegressedOut <-
  residuals( lm( boldMatrix ~ scale( preprocessedRestingState$globalSignal ) ) )

globalSignalDataFrame <- data.frame( Frame = rep( 1:numberOfTimeFrames, 2 ),
       GlobalSignal = c( preprocessedRestingState$globalSignal,
```

```
          apply( boldMatrixGlobalSignalRegressedOut, mean, MARGIN = 1 ) ),
       Type = factor( c( rep( "Before", numberOfTimeFrames ),
              rep( "After", numberOfTimeFrames ) ), levels = c( "Before", "After" ) ) )

ggplot( globalSignalDataFrame ) +
   geom_line( aes( x = Frame, y = GlobalSignal, colour = Type ), size = 0.5 ) +
     xlab( "Frame" ) + ylab( "Global signal intensity" ) +
     theme( legend.title = element_blank() ) + theme( aspect.ratio = 1/3 )
```

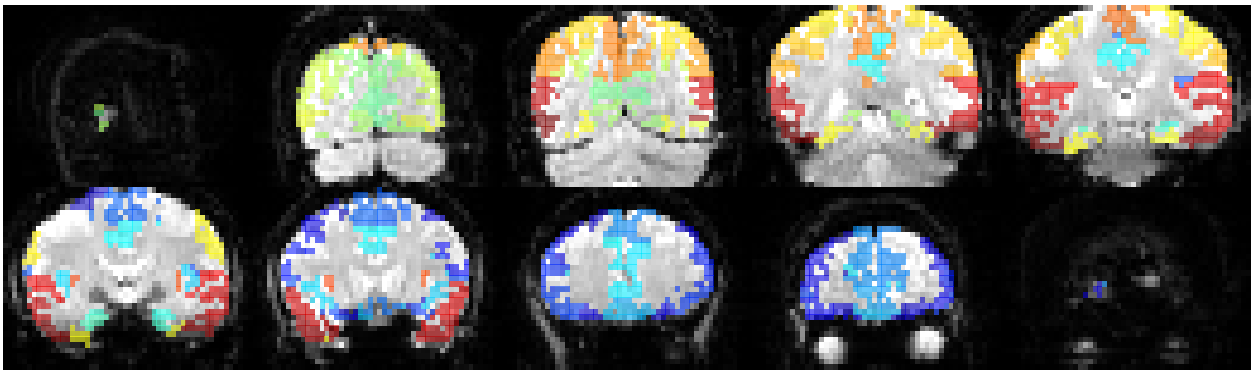



```
# Plot the CompCor nuisance signals.  Defined in terms of the PCA decomposition of
# the high frequency components of the BOLD signal.

numberOfCompCorComponents <- ncol( preprocessedRestingState$nuisanceVariables )
whichComponentLevels <- paste0( "CompCor", 1:numberOfCompCorComponents )
whichComponent <- factor( as.vector(
  matrix( rep( whichComponentLevels, numberOfTimeFrames ),
    nrow = numberOfTimeFrames, byrow = TRUE ) ), levels = whichComponentLevels )

compCorDataFrame <- data.frame(
  Frame = rep( 1:numberOfTimeFrames, numberOfCompCorComponents ),
  WhichComponent = whichComponent,
  Values = as.vector( preprocessedRestingState$nuisanceVariables ) )

ggplot( compCorDataFrame ) +
   geom_line( aes( x = Frame, y = Values ), size = 0.25 ) +
     facet_wrap( ~ WhichComponent, ncol = 3 ) +
     xlab( "Frame" ) + ylab( "CompCor value" ) +
     theme( legend.title = element_blank() ) + theme( aspect.ratio = 1/3 )
```



Calculate functional connectivity measures

```
# We use the AAL labels to calculate various functional connectivity measures for
# this subject.

# Mask out non-gray matter pixels from aalImage and plot image to ensure things look
# correct.

aalWarpedImage[restingStateBoldSegImage != 2] <- 0

invisible( plot.antsImage( restingStateBoldAverage, aalWarpedImage,
   alpha = 0.9, ncolumns = 5 ) )
```



```
# Determine the unique AAL labels and construct the correlation matrix

aalRoiLabelVector <- as.vector( as.array( aalWarpedImage[aalWarpedImage > 0] ) )
aalUniqueLabels <- sort( unique( aalRoiLabelVector ) )

boldMatrix <- timeseries2matrix(
  preprocessedRestingState$cleanBoldImage, restingStateBoldMaskImage )

boldLabelMatrix <- matrix( NA, nrow = nrow( boldMatrix ), ncol = length( aalUniqueLabels ) )
for( j in 1:length( aalUniqueLabels ) )
  {
  currentLabelIndices <- which( aalRoiLabelVector == aalUniqueLabels[j] )
```

```
  if( length( currentLabelIndices ) > 1 )
    {
    boldLabelMatrix[, j] <- rowMeans( boldMatrix[, currentLabelIndices] )
    }
  else
    {
    boldLabelMatrix[, j] <- mean( boldMatrix[, currentLabelIndices] )
    }
  }
correlationMatrix <- cor( boldLabelMatrix, boldLabelMatrix )
correlationMatrix[which( is.na( correlationMatrix ) )] <- 0
rownames( correlationMatrix ) <- colnames( correlationMatrix ) <-
  aalLabelTable$label_name[aalUniqueLabels]

# We calculate the significance for each entry.

cor.mtest <- function( mat, ... )
  {
  mat <- as.matrix( mat )
  n <- ncol( mat )
  p.mat <- matrix( NA, n, n )
  diag( p.mat ) <- 0

  for( i in 1:( n - 1 ) )
    {
    for( j in ( i + 1 ):n )
      {
      tmp <- cor.test( mat[, i], mat[, j], ... )
      p.mat[i, j] <- p.mat[j, i] <- tmp$p.value
      }
    }
  colnames( p.mat ) <- rownames( p.mat ) <- colnames( mat )
  p.mat
  }

p.mat <- cor.mtest( correlationMatrix )

uvaColors <- colorRampPalette( c( "#F59A2C", "#F1E5C7", "#E6E7E8", "#46A8C2", "#0D3268" ) )
correlationPlotFile <- paste0( figuresDirectory, '/CorrelationMatrix.pdf' );
pdf( height = 10, width = 10, file = correlationPlotFile )
corrplot( correlationMatrix, method = "circle", diag = FALSE, type = "upper",
  tl.col = "black", tl.cex = 0.48, tl.srt = 45, col = uvaColors( 200 ),
  p.mat = p.mat, sig.level = 0.01, insig = "blank" )
invisible( dev.off() )
```

```
# Instead of exploring the connectivity of the entire set of AAL labels, let's just
# look at specific networks (default mode and salience networks).  This information is in
# ``aalLabelTable``.

# Default mode network

aalDmnLabels <- aalLabelTable$label_num[which( aalLabelTable$isdmn > 0 )]
```

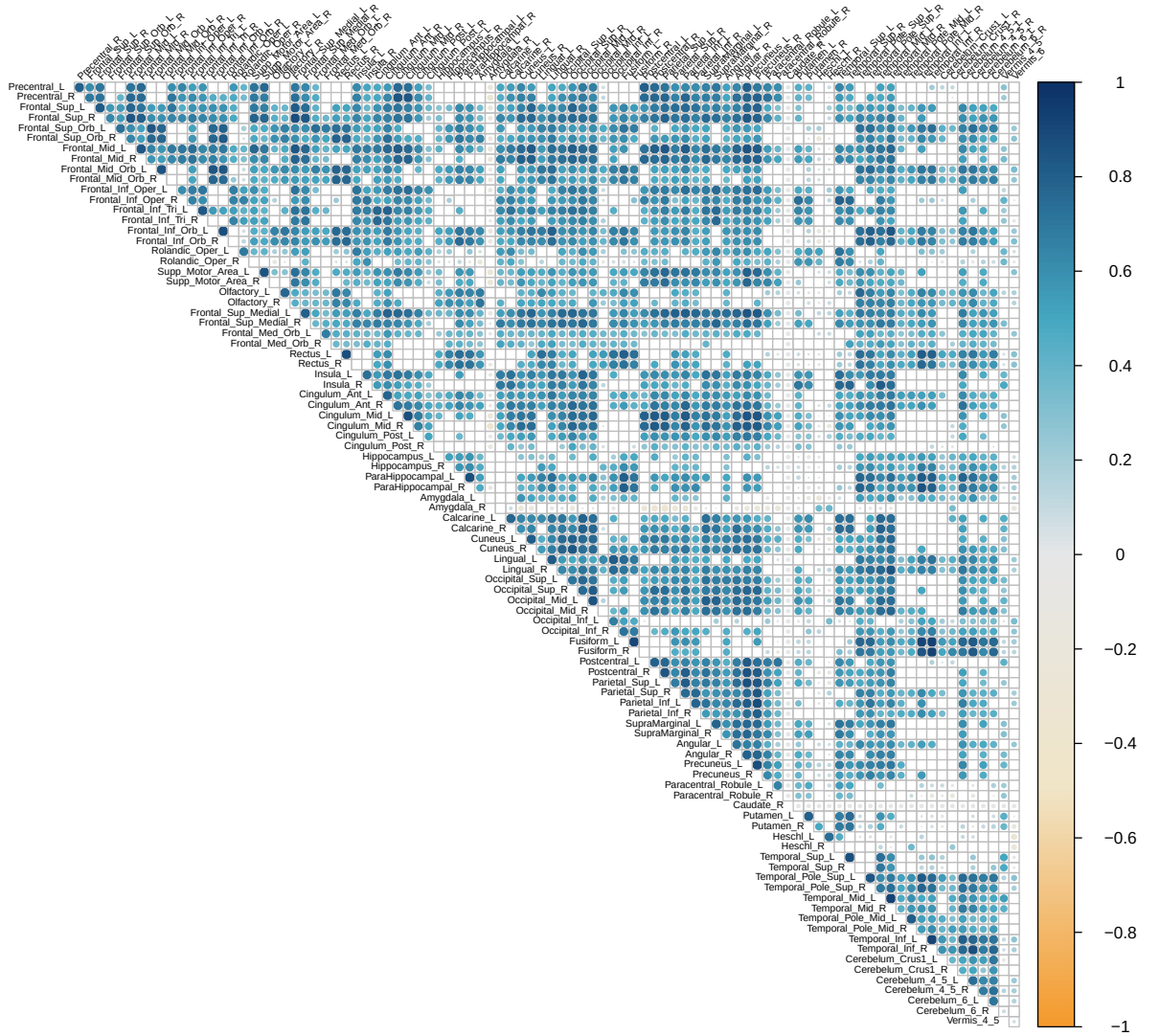


Figure 1: Correlation plot illustrating the functional connectivity relationships between AAL-defined regions.

```
dmnBoldLabelMatrix <- matrix( NA, nrow = nrow( boldMatrix ), ncol = length( aalDmnLabels ) )
for( j in 1:length( aalDmnLabels ) )
  {
  currentLabelIndices <- which( aalRoiLabelVector == aalDmnLabels[j] )
  if( length( currentLabelIndices ) > 1 )
    {
    dmnBoldLabelMatrix[, j] <- rowMeans( boldMatrix[, currentLabelIndices] )
    }
  else
    {
    dmnBoldLabelMatrix[, j] <- mean( boldMatrix[, currentLabelIndices] )
    }
  }
dmnCorrelationMatrix <- cor( dmnBoldLabelMatrix, dmnBoldLabelMatrix )
dmnCorrelationMatrix[which( is.na( dmnCorrelationMatrix ) )] <- 0
rownames( dmnCorrelationMatrix ) <- colnames( dmnCorrelationMatrix ) <-
  aalLabelTable$label_name[aalDmnLabels]

p.mat <- cor.mtest( dmnCorrelationMatrix )

dmnCorrelationPlotFile <- paste0( figuresDirectory, '/dmnCorrelationMatrix.pdf' )
pdf( height=5, width=5, file = dmnCorrelationPlotFile )
corrplot( dmnCorrelationMatrix, method = "circle", diag = FALSE, type = "upper",
  tl.col = "black", tl.cex = 0.75, tl.srt = 45, col = uvaColors( 200 ),
  p.mat = p.mat, sig.level = 0.01, insig = "blank" )
invisible( dev.off() )
```

```
# Now we calculate various graph-based measures from the connectivity relationships
# in the default mode network and plot the resulting graph.

networkGraph <- makeGraph( dmnCorrelationMatrix, graphdensity = 0.25, getEfficiency = T )

V( networkGraph$mygraph )$color <- "#F59A2C"
V( networkGraph$mygraph )$label.cex <- 1.0
E( networkGraph$mygraph )$width <- 1.5

dmnGraphPlotFile <- paste0( figuresDirectory, '/dmnGraph.pdf' )
pdf( height = 10, width = 10, file = dmnGraphPlotFile )
plot.igraph( networkGraph$mygraph, layout = layout.fruchterman.reingold )
invisible( dev.off() )
```

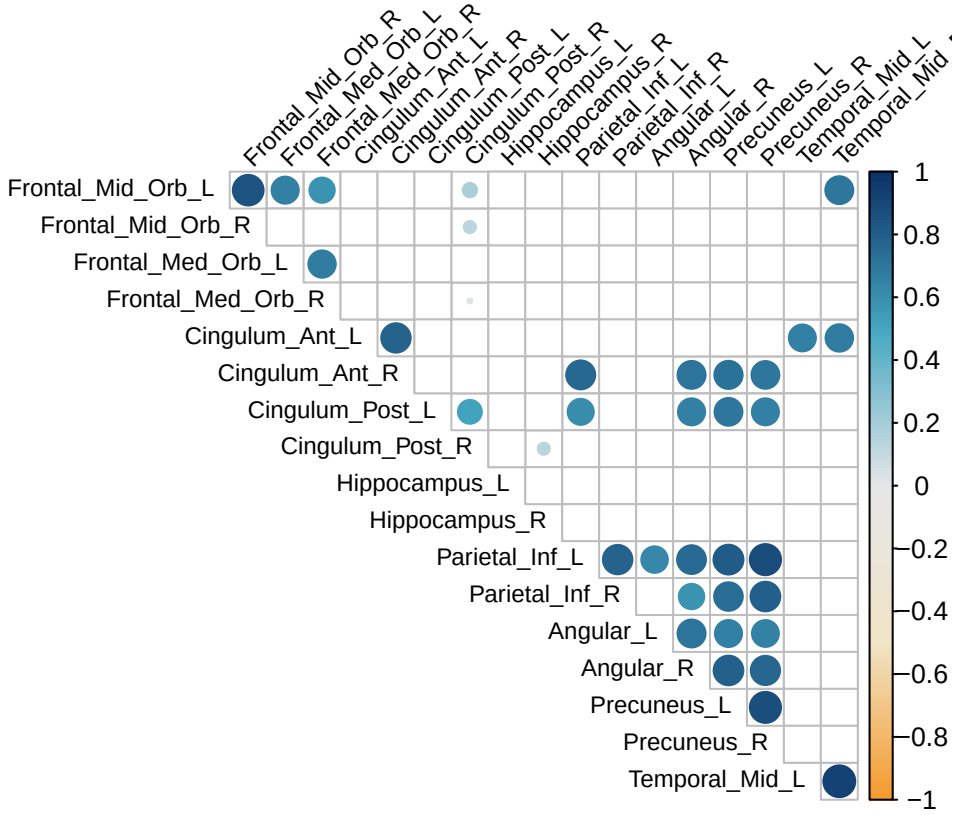


Figure 2: Correlation plot illustrating the functional connectivity relationships in the default mode network.

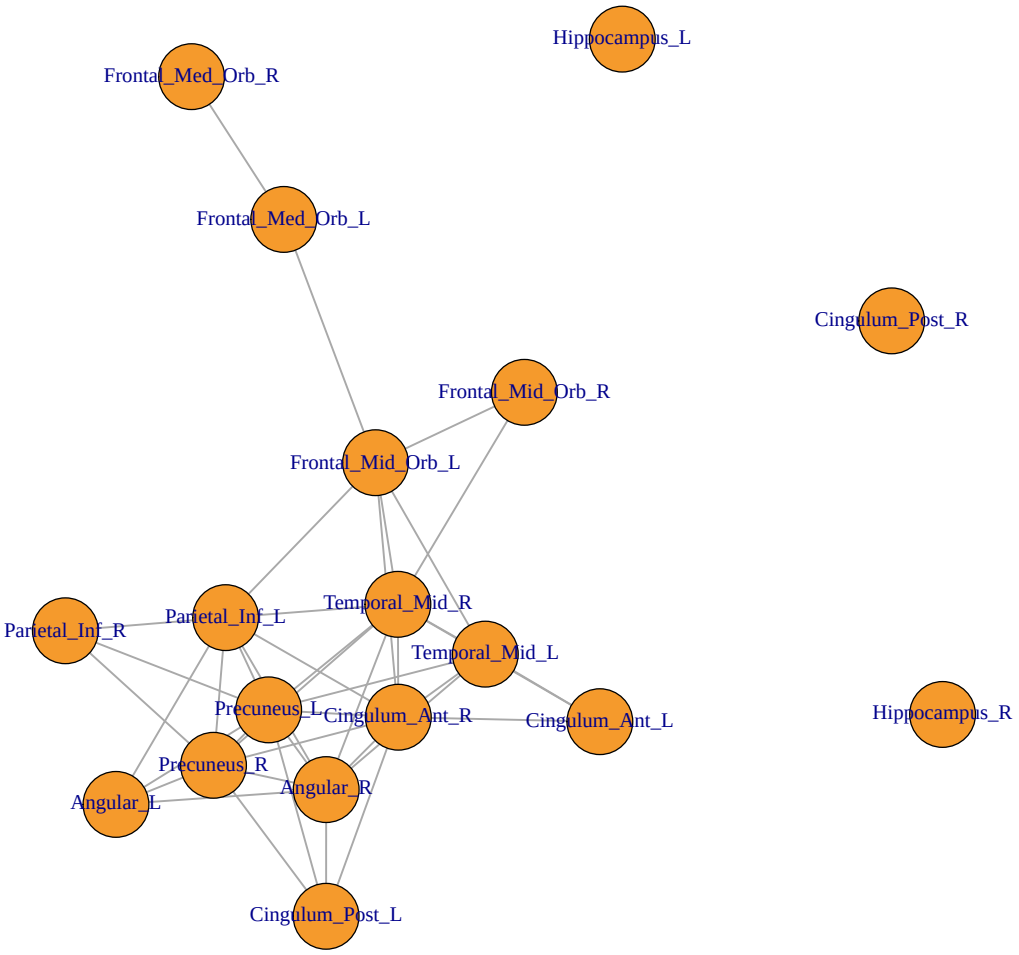


Figure 3: Graph of the functional connectivity of the default mode network.

Tutorial II: Brian's current strategy

Read in input data

```
# In this second tutorial, we expose more of the pre-processing aspects of the
# fMRI pipeline and use it to process a single subject from the MMRR (Kirby) data
# set:
#    B. A. Landman, et al. "Multi-Parametric Neuroimaging Reproducibility: A 3T
#    Resource Study", NeuroImage. (2010).
#
# We also explore functional connectivity using the nodal network described in
#    J. D. Power, et al. "Functional network organization of the human brain",
#    Neuron. (2011).
# instead of using the AAL image to define ROIs.

# Read in the Kirby subject 01 images

kirbyDirectory <- paste0( dataDirectory, "/KKI2009-01" )
kirbyT1File <- paste0( kirbyDirectory, "/KKI2009-01-MPRAGE.nii.gz" )
kirbySegFile <- paste0( kirbyDirectory, "/KKI2009-01-BrainSegmentation.nii.gz" )
kirbyBoldFile <- paste0( kirbyDirectory, "/KKI2009-01-fMRI.nii.gz" )

kirbyT1Image <- antsImageRead( kirbyT1File )
kirbySegImage <- antsImageRead( kirbyT1File )
kirbyBoldImage <- antsImageRead( kirbyT1File )

# Read in MNI template
```

References

1. Behzadi, Y., Restom, K., Liau, J., and Liu, T. T. "**A Component Based Noise Correction Method (CompCor) for BOLD and Perfusion Based FMRI**" *Neuroimage* 37, no. 1 (2007): 90–101. doi:10.1016/j.neuroimage.2007.04.042

2. Power, J. D., Barnes, K. A., Snyder, A. Z., Schlaggar, B. L., and Petersen, S. E. "**Spurious but Systematic Correlations in Functional Connectivity MRI Networks Arise from Subject Motion**" *Neuroimage* 59, no. 3 (2012): 2142–54. doi:10.1016/j.neuroimage.2011.10.018

3. Power, J. D., Mitra, A., Laumann, T. O., Snyder, A. Z., Schlaggar, B. L., and Petersen, S. E. "**Methods to Detect, Characterize, and Remove Motion Artifact in Resting State FMRI**" *Neuroimage* 84, (2014): 320–41. doi:10.1016/j.neuroimage.2013.08.048

4. Liu, T. T., Nalci, A., and Falahpour, M. "**The Global Signal in FMRI: Nuisance or Information?**" *Neuroimage* 150, (2017): 213–229. doi:10.1016/j.neuroimage.2017.02.036

5. Power, J. D., Cohen, A. L., Nelson, S. M., Wig, G. S., Barnes, K. A., Church, J. A., Vogel, A. C., Laumann, T. O., Miezin, F. M., Schlaggar, B. L., and Petersen, S. E. "**Functional Network Organization of the Human Brain**" *Neuron* 72, no. 4 (2011): 665–78. doi:10.1016/j.neuron.2011.09.006

6. Avants, B. B., Duda, J. T., Kilroy, E., Krasileva, K., Jann, K., Kandel, B. T., Tustison, N. J., Yan, L., Jog, M., Smith, R., Wang, Y., Dapretto, M., and Wang, D. J. J. "**The Pediatric Template of Brain Perfusion**" *Sci Data* 2, (2015): 150003. doi:10.1038/sdata.2015.3

7. Dhillon, P. S., Wolk, D. A., Das, S. R., Ungar, L. H., Gee, J. C., and Avants, B. B. "**Subject-Specific Functional Parcellation via Prior Based Eigenanatomy**" *Neuroimage* 99, (2014): 14–27. doi:10.1016/j.neuroimage.2014.05.026

8. Filiano, A. J., Xu, Y., Tustison, N. J., Marsh, R. L., Baker, W., Smirnov, I., Overall, C. C., Gadani, S. P., Turner, S. D., Weng, Z., Peerzade, S. N., Chen, H., Lee, K. S., Scott, M. M., Beenhakker, M. P., Litvak, V., and Kipnis, J. "**Unexpected Role of Interferon-γ in Regulating Neuronal Connectivity and Social Behaviour**" *Nature* 535, no. 7612 (2016): 425–9. doi:10.1038/nature18626