A Small Notebook Demonstrating Castor's Meridian Contrast Overlay Creation

In [3]:

```python
import os

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd

from preprocess import *
from preprocess import _normalize_array

from analysis import *
```

Here we started with a previously packed nifti file. The first step is to correct the monkey position to sphinx. We'll als create a new directory for this preprocessing and copy the files over

In [4]:

```python
root = '/Users/loggiasr/Projects/fmri/monkey_fmri/castor_test'
functional_dir = os.path.join(root, 'functional/20100131/')
SOURCE = [os.path.join(functional_dir, f) for f in os.listdir(functional_dir) if f[0] != '
print(SOURCE)
```

```
['/Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/functional/20100131/015', '/Users/
loggiasr/Projects/fmri/monkey_fmri/castor_test/functional/20100131/012', '/Users/loggiasr/
Projects/fmri/monkey_fmri/castor_test/functional/20100131/013', '/Users/loggiasr/Projects/
fmri/monkey_fmri/castor_test/functional/20100131/014', '/Users/loggiasr/Projects/fmri/monk
ey_fmri/castor_test/functional/20100131/007', '/Users/loggiasr/Projects/fmri/monkey_fmri/c
astor_test/functional/20100131/009', '/Users/loggiasr/Projects/fmri/monkey_fmri/castor_tes
t/functional/20100131/008', '/Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/functio
nal/20100131/006', '/Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/functional/20100
131/011', '/Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/functional/20100131/016',
'/Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/functional/20100131/017', '/Users/l
oggiasr/Projects/fmri/monkey_fmri/castor_test/functional/20100131/010', '/Users/loggiasr/P
rojects/fmri/monkey_fmri/castor_test/functional/20100131/005']
```

In [3]:

```python
convert_to_sphinx(SOURCE, fname='f.nii')
```

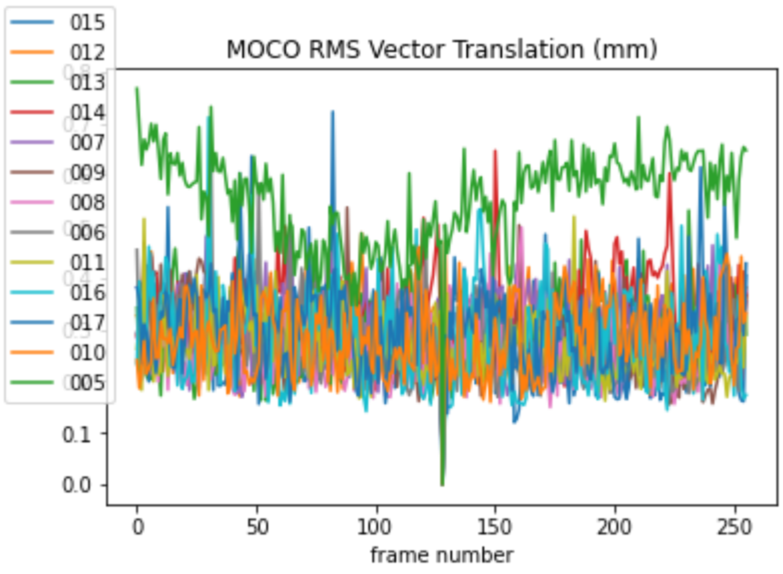
```
210915-12:15:23,511 nipype.interface INFO:
         stdout 2021-09-15T12:15:23.510964:mri_convert --sphinx --out_type nii --input_vol
ume /Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/functional/20100131/013/f.nii --
output_volume /Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/functional/20100131/01
3/f_sphinx.nii
210915-12:15:23,560 nipype.interface INFO:
         stdout 2021-09-15T12:15:23.549284:mri_convert --sphinx --out_type nii --input_vol
ume /Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/functional/20100131/012/f.nii --
output_volume /Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/functional/20100131/01
2/f_sphinx.nii
210915-12:15:23,794 nipype.interface INFO:
         stdout 2021-09-15T12:15:23.794342:mri_convert --sphinx --out_type nii --input_vol
ume /Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/functional/20100131/010/f.nii --
output_volume /Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/functional/20100131/01
0/f_sphinx.nii
210915-12:15:23,796 nipype.interface INFO:
         stdout 2021-09-15T12:15:23.795967:mri_convert --sphinx --out_type nii --input_vol
ume /Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/functional/20100131/016/f.nii --
output_volume /Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/functional/20100131/01
6/f_sphinx.nii
210915-12:15:23,799 nipype.interface INFO:
         stdout 2021-09-15T12:15:23.798108:mri_convert --sphinx --out_type nii --input_vol
ume /Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/functional/20100131/014/f.nii --
```

Run FSL Motion Correction

In [4]:
```
motion_correction(SOURCE, fname='f_sphinx.nii')
```

```
/Users/loggiasr/Projects/fmri/monkey_fmri/preprocess.py:160: UserWarning: Matplotlib is cu
rrently using module://matplotlib_inline.backend_inline, which is a non-GUI backend, so ca
nnot show the figure.
  fig.show()
The PostScript backend does not support transparency; partially transparent artists will b
e rendered opaque.
The PostScript backend does not support transparency; partially transparent artists will b
e rendered opaque.
```

Out[4]:
```
[<nipype.interfaces.base.support.InterfaceResult at 0x7ff470521040>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7ff481360fa0>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7ff4705216a0>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7ff4705210d0>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7ff470521070>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7ff470521790>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7ff470521760>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7ff470521490>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7ff470521370>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7ff470521910>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7ff470521520>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7ff470521430>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7ff4705211f0>]
```



In [5]:
```
# pad the funcitonal data
functional_to_cube(SOURCE, fname='moco.nii.gz')
```

```
96
96
96
96
96
96
96
96
96
96
96
96
96
```

Now we will downsample our high resolution anatomical for the use in registration. Working with the full size scan is computationally intensive for no reason, and can actually reduce registration accuracy.

In [3]:
```
# center_nifti(source_dir=os.path.join(root, 'mri'), fname='orig.mgz')
create_low_res_anatomical(source_dir=os.path.join(root, 'mri'), fname='orig.mgz')
```

```
mri_convert /Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/mri/orig.mgz -vs 3 3 3 /
Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/mri/low_res.nii
reading from /Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/mri/orig.mgz...
TR=2500.00, TE=2.99, TI=1100.00, flip angle=8.00
i_ras = (-1, 0, 0)
j_ras = (0, 0, -1)
k_ras = (0, 1, 0)
Reslicing using trilinear interpolation
writing to /Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/mri/low_res.nii...
mri_convert /Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/mri/low_res.nii -iis 1 -
ijs 1 -iks 1 /Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/mri/low_res.nii
reading from /Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/mri/low_res.nii...
TR=2500.00, TE=0.00, TI=0.00, flip angle=0.00
i_ras = (-1, 0, 0)
j_ras = (0, 0, -1)
k_ras = (0, 1, 0)
writing to /Users/loggiasr/Projects/fmri/monkey_fmri/castor_test/mri/low_res.nii...
```

Skull Strip the functional and anatomical data. This is done before registration to avoid weird skull artifacts in the functional from effecting the registration optimization process.

In [4]:
```
ana_dir = os.path.join(root, 'mri')
skull_strip(SOURCE, fname='f_cubed.nii', is_time_series=True)
skull_strip([ana_dir], fname='low_res.nii', is_time_series=False)
```

Now we will register the functional data to a detailed anatomical scan using FSL flirt and nirt

This is accomplished by first minimizing the mutual information between the functional and anatomical scan via only linear transforms (rotation, sheer, translation, dilation)

Next the functional and anatomical data are projected into a high dimensional space using a kernel function (default kernel is radial basis) and mutual information is minimized via linear transforms with respect to the kernel space. This results in fine grain deformations to the functional data to better match the anatomical structures.

In [5]:
```
linear_affine_registration(functional_input_dirs=SOURCE, fname='stripped.nii.gz',
                           template_file=os.path.join(ana_dir, 'stripped.nii.gz'))
nonlinear_registration(functional_input_dirs=SOURCE, source_fname='stripped.nii.gz',
                       template_file=os.path.join(ana_dir, 'stripped.nii.gz'),
                       transform_input_dir=SOURCE, affine_fname='stripped_flirt.mat', )
```

```
211006-13:46:47,452 nipype.interface INFO:
         stderr 2021-10-06T13:46:47.451667:Warning: An input intended to be a single 3D vo
lume has multiple timepoints. Input will be truncated to first volume, but this functional
ity is deprecated and will be removed in a future release.
211006-13:46:47,502 nipype.interface INFO:
         stderr 2021-10-06T13:46:47.501635:Warning: An input intended to be a single 3D vo
lume has multiple timepoints. Input will be truncated to first volume, but this functional
ity is deprecated and will be removed in a future release.
211006-13:46:47,535 nipype.interface INFO:
         stderr 2021-10-06T13:46:47.535352:Warning: An input intended to be a single 3D vo
lume has multiple timepoints. Input will be truncated to first volume, but this functional
ity is deprecated and will be removed in a future release.
211006-13:46:47,644 nipype.interface INFO:
         stderr 2021-10-06T13:46:47.644508:Warning: An input intended to be a single 3D vo
lume has multiple timepoints. Input will be truncated to first volume, but this functional
ity is deprecated and will be removed in a future release.
```

```
211006-13:47:15,358 nipype.interface INFO:
         stderr 2021-10-06T13:47:15.344931:Warning: An input intended to be a single 3D vo
lume has multiple timepoints. Input will be truncated to first volume, but this functional
ity is deprecated and will be removed in a future release.
211006-13:47:15,367 nipype.interface INFO:
         stderr 2021-10-06T13:47:15.169021:Warning: An input intended to be a single 3D vo
lume has multiple timepoints. Input will be truncated to first volume, but this functional
ity is deprecated and will be removed in a future release.
```

Out[5]:

```
[<nipype.interfaces.base.support.InterfaceResult at 0x7fc200befd00>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c00700>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c00610>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c00220>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c00490>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c00640>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c00400>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c00520>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c002b0>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c001f0>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c00460>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c006a0>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c003a0>]
```

Nonlinear registration will have generated a file of warp-coefficients that describe both the affine linear tranformation and the nonlinear local warps to the functional data. In order to apply this to the 4d time-series, we must use preform_nifti_registration, which wraps FSL.ApplyWarp. We pass the template anatomical again so we can preform a sanity check between the produced registered functional nifti and the anatomical.

In [6]:

```
preform_nifti_registration(functional_input_dirs=SOURCE, transform_input_dir=SOURCE, templ
                              source_fname='stripped.nii.gz', transform_fname='reg_tensor.ni
```

Out[6]:

```
[<nipype.interfaces.base.support.InterfaceResult at 0x7fc200c00250>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc221d21c70>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c00850>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c00a00>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c00880>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c005b0>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c00790>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc210519580>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c00940>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c009a0>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c007f0>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc210519220>,
 <nipype.interfaces.base.support.InterfaceResult at 0x7fc200c008e0>]
```

Let's Confirm the registration by visualizing it on top of the anatomical:

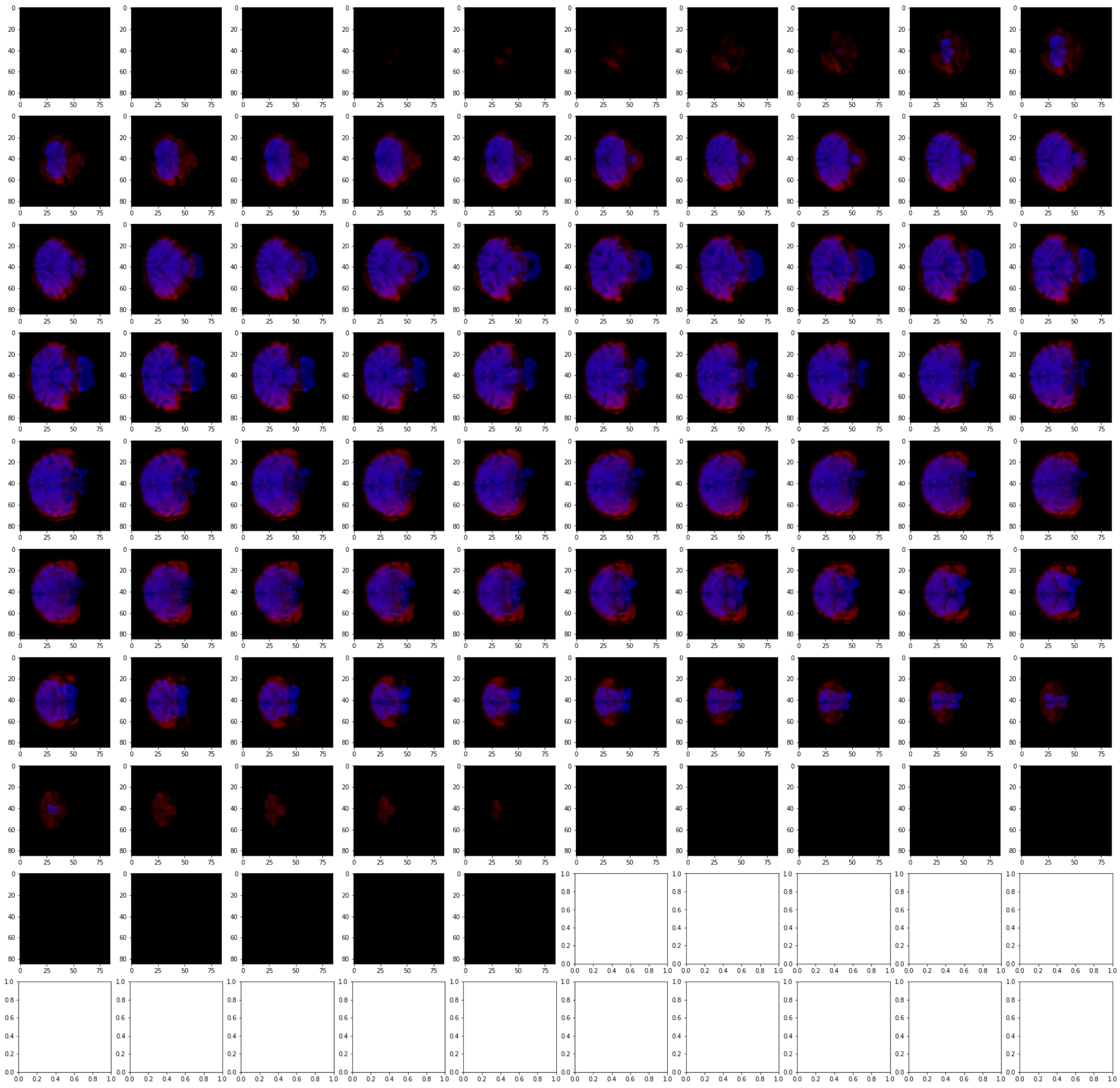
In [10]:

```
import nibabel as nib
func_nii = nib.load(os.path.join(SOURCE[0], 'registered.nii.gz'))
func_data = np.array(func_nii.get_fdata())
func_data = np.mean(func_data, axis=3) # average across time
num_plots = int(np.ceil(np.sqrt(func_data.shape[2])))

func_data = np.stack([func_data, np.zeros(func_data.shape), np.zeros(func_data.shape)], ax
anat_data = nib.load(os.path.join(ana_dir, 'stripped.nii.gz'))
anat_data = np.array(anat_data.get_fdata())
anat_data = np.stack([np.zeros(anat_data.shape), np.zeros(anat_data.shape), anat_data], ax

overlay = _normalize_array(func_data) * .5 + _normalize_array(anat_data)
overlay = (overlay / max(overlay.flatten()))
fig, ax = plt.subplots(num_plots, num_plots, figsize=(32, 32))
for i in range(num_plots):
    for j in range(num_plots):
        try:
            ax[i, j].imshow(overlay[:, :, i*num_plots + j])
        except IndexError:
            continue
fig.show()
```

```
/var/folders/89/dd1svxwn3f942px6dg4v1twx15nm77/T/ipykernel_55744/1882994868.py:21: UserWar
ning: Matplotlib is currently using module://matplotlib_inline.backend_inline, which is a
non-GUI backend, so cannot show the figure.
  fig.show()
```



Normalize the functional data:

In [11]:
```
normalize(SOURCE, output=None, fname='registered.nii.gz')
```

```
<class 'nibabel.nifti1.Nifti1Header'> object, endian='<'
sizeof_hdr      : 348
data_type       : b''
db_name         : b''
extents         : 16384
session_error   : 0
```

```
regular         : b'r'
dim_info        : 0
dim             : [  4  85  85  85 256   1   1   1]
intent_p1       : 0.0
intent_p2       : 0.0
intent_p3       : 0.0
intent_code     : none
datatype        : float32
bitpix          : 32
slice_start     : 0
pixdim          : [-1.  1.  1.  1.  2.  1.  1.  1.]
vox_offset      : 0.0
scl_slope       : nan
scl_inter       : nan
slice_end       : 0
slice_code      : unknown
xyzt_units      : 10
cal_max         : 0.0
cal_min         : 0.0
slice_duration  : 0.0
toffset         : 0.0
glmax           : 0
glmin           : 0
descrip         : b'6.0.5:9e026117'
aux_file        : b''
qform_code      : unknown
sform_code      : aligned
quatern_b       : 0.0
quatern_c       : 0.70710677
quatern_d       : -0.70710677
qoffset_x       : 116.09509
qoffset_y       : -115.18523
qoffset_z       : 143.20944
srow_x          : [ -1.         0.         0.       116.09509]
srow_y          : [   0.          0.          1.      -115.18523]
srow_z          : [   0.       -1.         0.       143.20944]
intent_name     : b''
magic           : b'n+1'
```

Begin Analysis

The contrast based analysis follows the following steps.

- first the contrast matrix is specified. Each row is an independent contrast. The values indicate the wieght on the different conditions (as listed in the paradigm file). In this paradigm file, index 0 is background, 1 is horizontal, and 2 is vertical. Thus the contrast specified in row one compares verticle and horizontal.
- the intras_subject_contrast function creates a design matrix and preforms LinearRegression between the design matrix and the functional data.
- The design matrix is created by taking the stimul sequence, one-hot encoding it and convolving it with estimated hemodynmic response functions in order to obtain expected bold signal along each independent channel
- A GLM can then take the whole design matrix as input, and fit to the observed MRI data. The coefficients of the fit GLM will represent the correlation between a voxels activity and the expectred activity for a stimuli along each channel. sklearns highly optimized GLM estimator is used here.
- The final contrasts can be obtained by simple matrix multiplication of the coefficient matrix with the contrast matrix, since this equates to summing across conditions with our desired weights.

```
In [ ]:
```

```
desired_contrast_mat = np.array([[0, -1, 1],
                                 [-1, .5, .5]]).T
contrast_desc = ['vertical_vs_horizontal', 'null_vs_avg_vert_horiz']
res = intra_subject_contrast(run_dirs=SOURCE,
                             paradigm_file=os.path.join(root, 'stimuli/meridian_mapper_orc
                             contrast_matrix=desired_contrast_mat,
                             contrast_descriptors=contrast_desc,
                             output_dir=os.path.join(root, 'analysis_out'),
                             fname='normalized.nii',)
```

creating contrast overlay

create_contrast_surface reshapes the contrast volumes to match the original high resolution anatomical scan, them projects it onto the white matter surface computed in the anatomical pipline. This generates and overlay in the analysis_out folder that can be applied to any of the surfaces.

In [3]:
```
create_contrast_surface(anatomical_white_surface='castor_test/surf/lh.white',
                        contrast_vol_path='castor_test/analysis_out/condition_vertical_vs_
                        orig_low_res_anatomical='castor_test/mri/stripped.nii.gz',
                        orig_high_res_anatomical='castor_test/mri/orig.mgz',
                        hemi='lh', subject_id='castor_test')
```

```
210915-14:22:13,819 nipype.interface WARNING:
         Creating nii file with extension .nii: upsampled_contrast_surface_castor_test.nii
210915-14:22:13,975 nipype.interface INFO:
         stdout 2021-09-15T14:22:13.975222:srcvol = /Users/loggiasr/Projects/fmri/monkey_f
mri/castor_test/analysis_out/upsampled_contrast.nii
210915-14:22:13,976 nipype.interface INFO:
         stdout 2021-09-15T14:22:13.975222:srcreg unspecified
210915-14:22:13,977 nipype.interface INFO:
         stdout 2021-09-15T14:22:13.975222:srcregold = 0
210915-14:22:13,977 nipype.interface INFO:
         stdout 2021-09-15T14:22:13.975222:srcwarp unspecified
210915-14:22:13,978 nipype.interface INFO:
         stdout 2021-09-15T14:22:13.975222:surf = inflated
210915-14:22:13,978 nipype.interface INFO:
         stdout 2021-09-15T14:22:13.975222:hemi = lh
210915-14:22:13,979 nipype.interface INFO:
         stdout 2021-09-15T14:22:13.975222:ProjFrac = 0.5
210915-14:22:13,980 nipype.interface INFO:
         stdout 2021-09-15T14:22:13.975222:thickness = thickness
210915-14:22:13,980 nipype.interface INFO:
         stdout 2021-09-15T14:22:13.975222:reshape = 0
210915-14:22:13,981 nipype.interface INFO:
         stdout 2021-09-15T14:22:13.975222:interp = nearest
210915-14:22:13,981 nipype.interface INFO:
         stdout 2021-09-15T14:22:13.975222:float2int = round
210915-14:22:13,982 nipype.interface INFO:
         stdout 2021-09-15T14:22:13.975222:GetProjMax = 0
210915-14:22:13,982 nipype.interface INFO:
         stdout 2021-09-15T14:22:13.975222:INFO: float2int code = 0
210915-14:22:13,983 nipype.interface INFO:
         stdout 2021-09-15T14:22:13.975222:niiRead(): NIFTI_UNITS_UNKNOWN, assuming mm
210915-14:22:13,983 nipype.interface INFO:
         stdout 2021-09-15T14:22:13.975222:Done loading volume
210915-14:22:13,984 nipype.interface INFO:
         stdout 2021-09-15T14:22:13.975222:Computing registration from header.
210915-14:22:13,984 nipype.interface INFO:
         stdout 2021-09-15T14:22:13.975222:  Using /Users/loggiasr/Projects/fmri/monkey_fm
ri/.//mri/orig.mgz as target reference.
210915-14:22:13,984 nipype.interface INFO:
         stdout 2021-09-15T14:22:13.975222:-------- original matrix -----------
```

```
210915-14:22:13,985 nipype.interface INFO:
	 stdout 2021-09-15T14:22:13.975222: 1.00000   0.00000   0.00000   0.00000;
210915-14:22:13,986 nipype.interface INFO:
	 stdout 2021-09-15T14:22:13.975222: 0.00000   1.00000   0.00000   0.00000;
210915-14:22:13,986 nipype.interface INFO:
	 stdout 2021-09-15T14:22:13.975222: 0.00000   0.00000   1.00000   0.00000;
210915-14:22:13,987 nipype.interface INFO:
	 stdout 2021-09-15T14:22:13.975222: 0.00000   0.00000   0.00000   1.00000;
210915-14:22:14,505 nipype.interface INFO:
	 stdout 2021-09-15T14:22:14.505818:-------- original matrix -----------
210915-14:22:14,506 nipype.interface INFO:
	 stdout 2021-09-15T14:22:14.505818:Reading surface /Users/loggiasr/Projects/fmri/m
onkey_fmri/.//surf/lh.inflated
210915-14:22:14,506 nipype.interface INFO:
	 stdout 2021-09-15T14:22:14.505818:Done reading source surface
210915-14:22:14,507 nipype.interface INFO:
	 stdout 2021-09-15T14:22:14.507405:Reading thickness /Users/loggiasr/Projects/fmr
i/monkey_fmri/.//surf/lh.thickness
210915-14:22:14,521 nipype.interface INFO:
	 stdout 2021-09-15T14:22:14.521131:Done
210915-14:22:14,521 nipype.interface INFO:
	 stdout 2021-09-15T14:22:14.521131:Mapping Source Volume onto Source Subject Surfa
ce
210915-14:22:14,579 nipype.interface INFO:
	 stdout 2021-09-15T14:22:14.579361: 1 1 1 1
210915-14:22:14,579 nipype.interface INFO:
	 stdout 2021-09-15T14:22:14.579361:using old
210915-14:22:14,581 nipype.interface INFO:
	 stdout 2021-09-15T14:22:14.581774:Done mapping volume to surface
210915-14:22:14,810 nipype.interface INFO:
	 stdout 2021-09-15T14:22:14.810305:Number of source voxels hit = 79239
210915-14:22:14,811 nipype.interface INFO:
	 stdout 2021-09-15T14:22:14.810305:Writing to /Users/loggiasr/Projects/fmri/monkey
_fmri/castor_test/analysis_out/upsampled_contrast_surface_castor_test.nii
210915-14:22:14,811 nipype.interface INFO:
	 stdout 2021-09-15T14:22:14.810305:Dim: 134181 1 1
210915-14:22:14,896 nipype.interface WARNING:
	 Creating nii file with extension .nii: upsampled_contrast_surface_castor_test.nii
```

```
In [ ]:
```