Report - Daehyun Cho

In advance, every work about this assignment is uploaded on github.

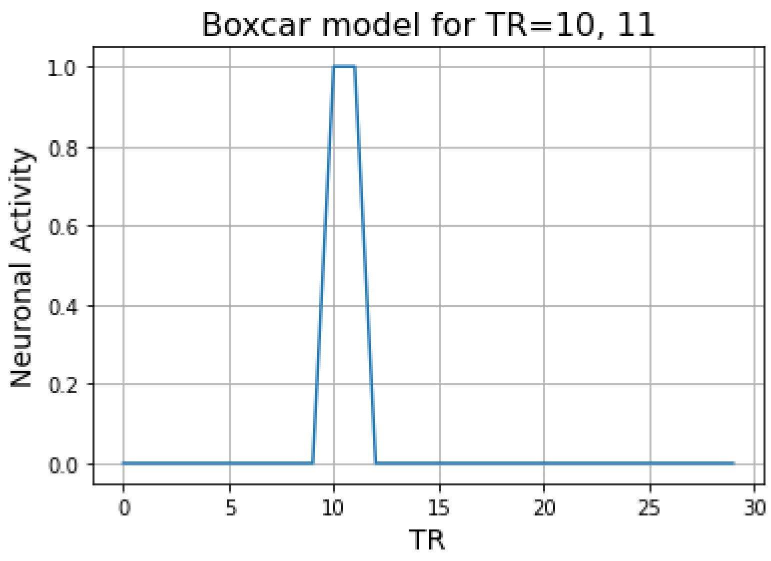
Also, if .pdf files are broken, follow this page.

[0] Data Preparation

Let's suppose the following for the measured BOLD signal (TR = 2 s).

(a) ***Each of the task was performed at 20 s during 4 s (i.e., 10th and 11th TRs). Please use the boxcar model for neural activity***

I made boxcar $N(t)$ for this



(b) ***Consider the 0 – 20 s (i.e., 0th – 9th TRs) and 40 s – 60 s (i.e., 20th – 29th TRs) as baseline period***

(c) ***Calculate the percentage (%) BOLD (pBOLD) signal based on (b)***

(d) ***Conduct the estimation of the HRFs using the pBOLD signals calculated in (c) by applying each of the following methods [1] - [4].***

When estimating HRFs, please use the pBOLD signal between 18 s – 52 s (i.e., 8th to 25th TRs) which includes a task-related period in between the baseline periods

---

[1] Deconvolution approach

(a) ***Set the reasonable number of Fast Fourier transform (FFT) points***

I couldn't understand *reasonable number* of FFT because there is no way of giving hyperparameters on FFT. I attach some of the research that I endeavored.

I set below intervals as number of points.

- TR=8 ~ 25
- TR=8 ~ 16
- TR=10 ~ 16

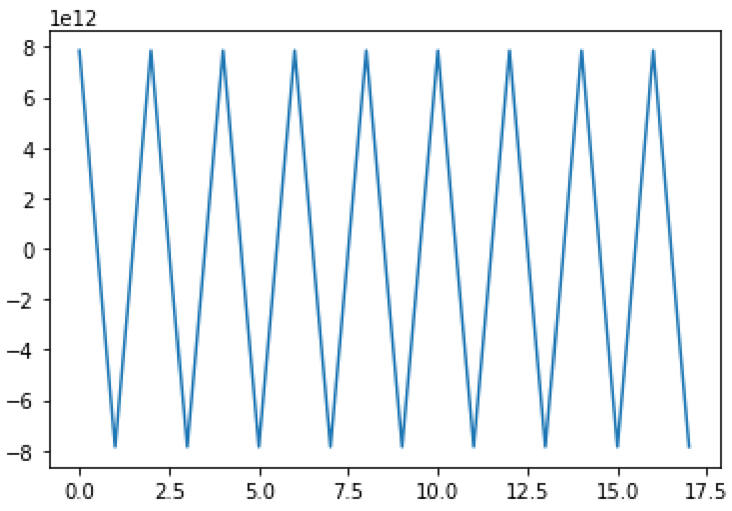
Re: Deconvolution

Is it a reasonable approach to remove a hemodynamic response function from a timeseries of EPI data with deconvolution? That is, converting both the HRF and the time series into frequency space, dividing the data by the HRF, and then coverting it back?

https://afni.nimh.nih.gov/afni/community/board/read.php?1,74554,74555

(b) ***Perform the FFT and inverse FFT to estimate the HRF in the time domain for each of the voxels in the ROI***

I followed every steps... but didn't work... Below one is estimated HRF...



(c) ***Check the estimated HRFs across the voxels and average the estimated HRFs***

(d) ***Evaluate your results by comparing the average estimated HRF and average pBOLD signal***

I've

---

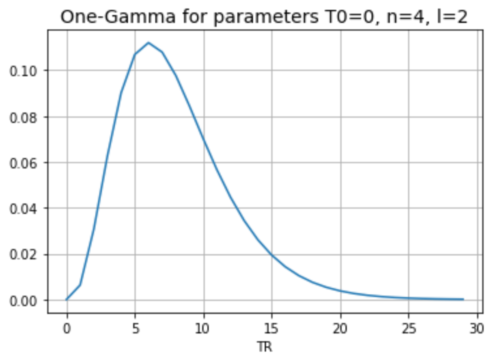
[2] Using a flexible mathematical model of HRF (one gamma function)

(a) ***Set the reasonable range of values as a grid for each of the parameters (i.e., T0, n, and l)***

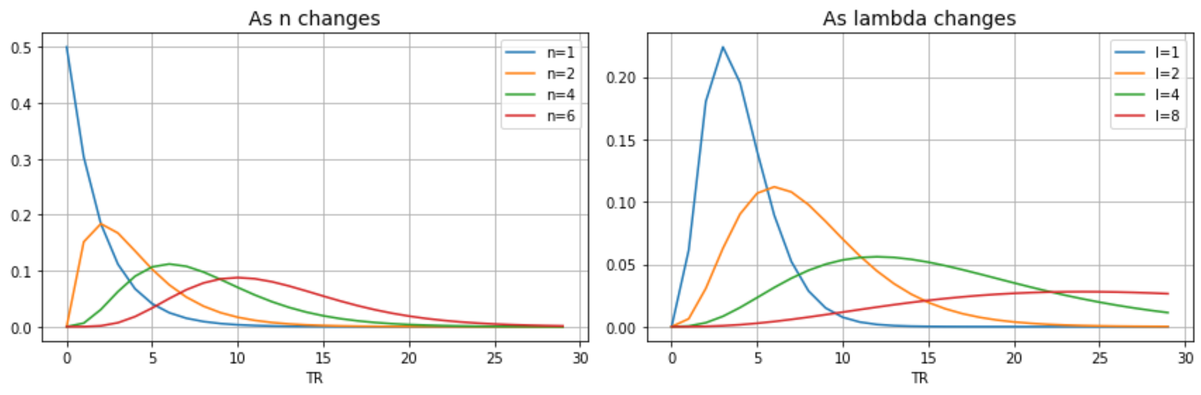
With T_0, I didn't set a range for this — I just set this to 0.

Since one-gamma function looks like below formula, and it's shape with a given parameters, T_0 only cares about the onset. Therefore I didn't modify any of

$$\Gamma_1(t; T_0, n, \lambda) = \frac{(t - T_0)^{n-1}}{\lambda^n (n-1)!}$$



We know that n takes a role of the shape, while λ takes a role of scale. With different parameters



I aimed to find the one that has similar peak. So I set n around 2 ~ 5, λ around 2 ~ 7.

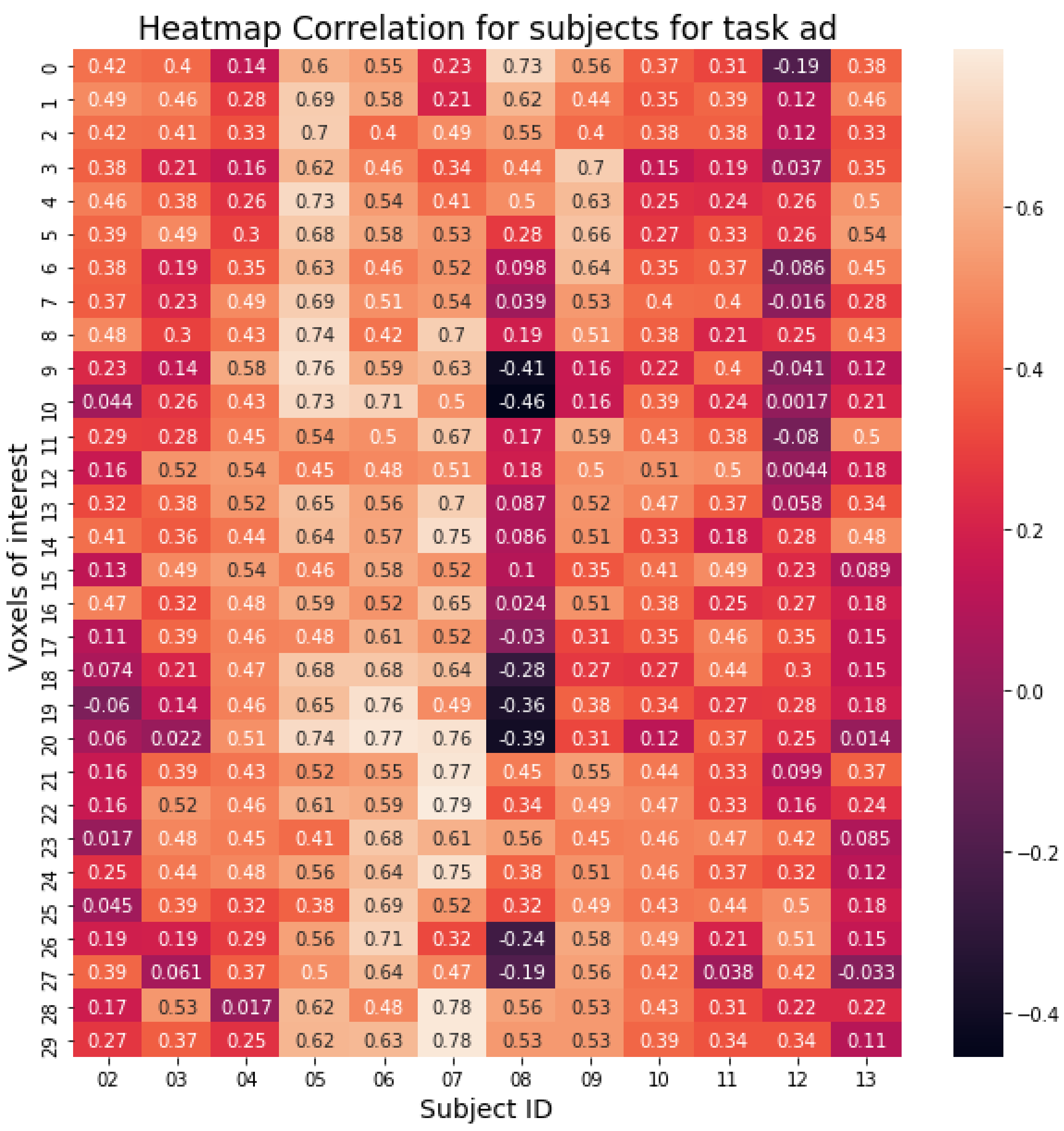
(b) ***Calculate Pearson's correlation coefficients (CC) between the pBOLD signal and estimated pBOLD from the modeled HRF across all combinatorial sets of model parameters for each voxel.***
(c). ***Calculate the average CC across all voxels within an ROI for each of the parameter set and choose the optimal parameter set that showed the maximum average CC. The HRF obtained from the optimal parameter set is the estimated HRF of the pBOLD signal.***

With mean value among ROI voxel and subjects, I found the results below

One-gamma Correlation set

Task	Best Correlation	n1	l1
Auditory	0.3501	3	2
Left hand	0.549	3	2
Right hand	0.488	3	2
Visual	0.5857	3.2	2

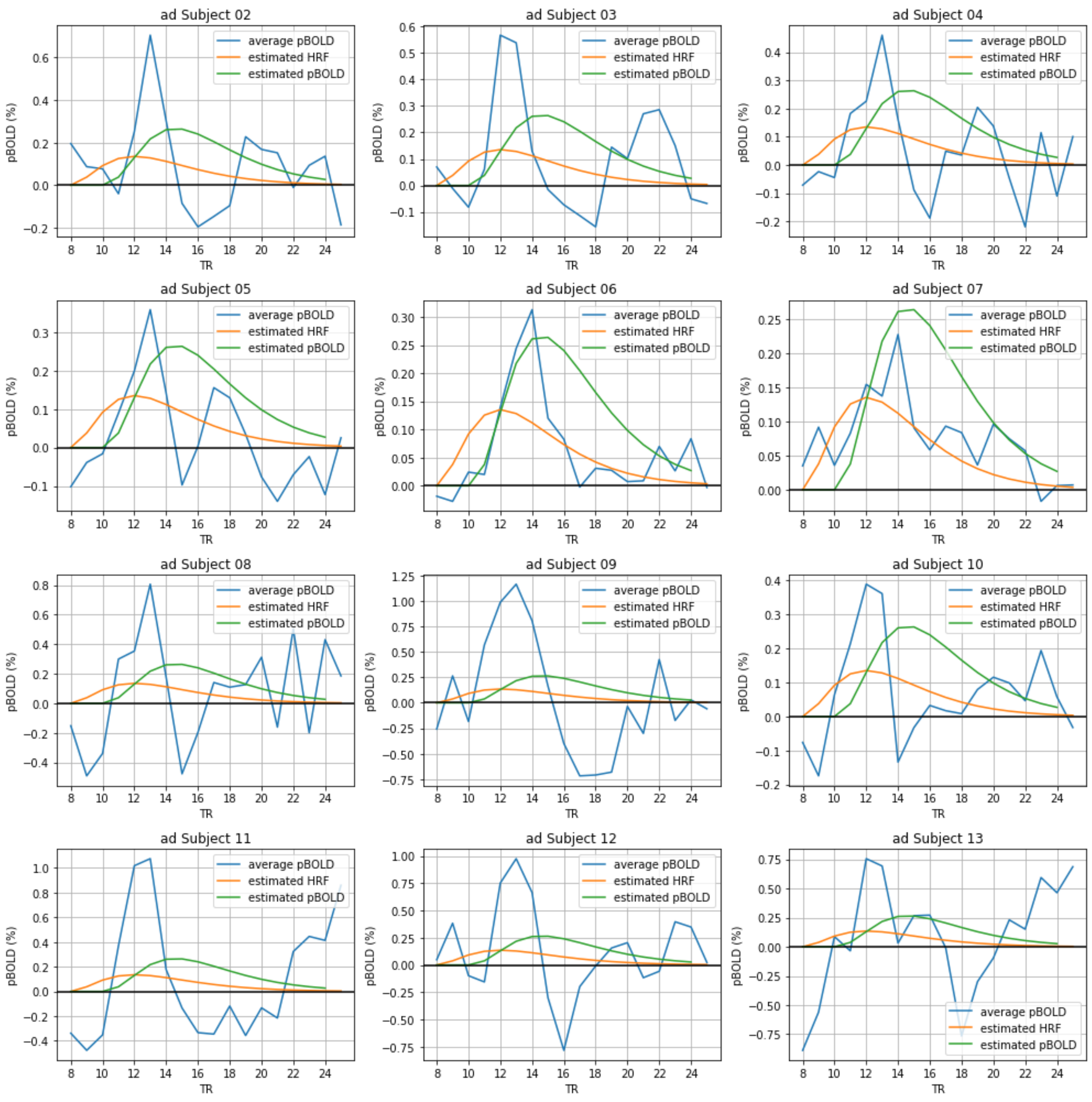
Below figure is a heatmap of estimated HRF and pBOLD among voxels of interest and subjects. Since the figure is too big, I didn't attach them here, but you can see them on codes.



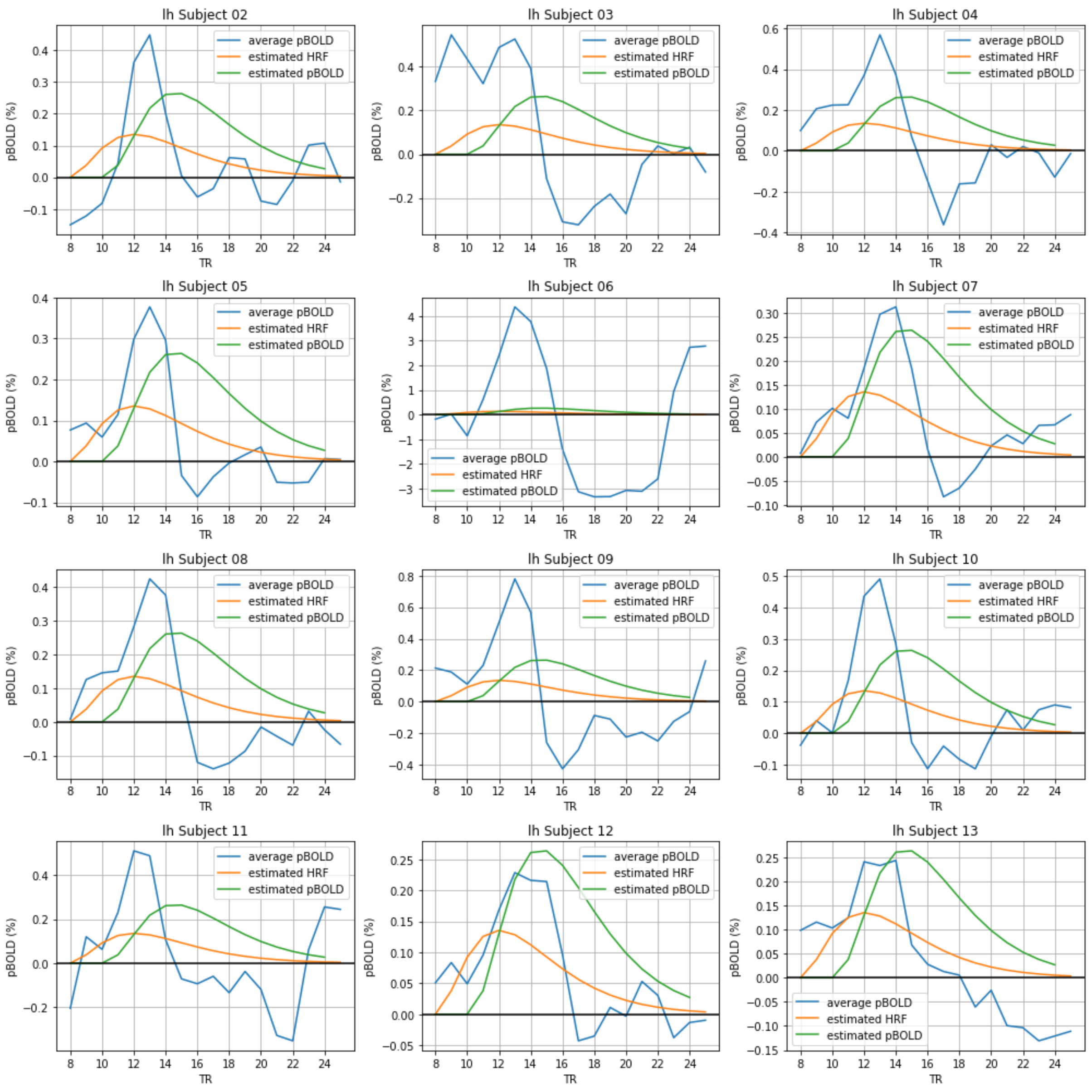
	02	03	04	05	06	07	08	09	10	11	12	13
0	0.42	0.4	0.14	0.6	0.55	0.23	0.73	0.56	0.37	0.31	-0.19	0.38
1	0.49	0.46	0.28	0.69	0.58	0.21	0.62	0.44	0.35	0.39	0.12	0.46
2	0.42	0.41	0.33	0.7	0.4	0.49	0.55	0.4	0.38	0.38	0.12	0.33
3	0.38	0.21	0.16	0.62	0.46	0.34	0.44	0.7	0.15	0.19	0.037	0.35
4	0.46	0.38	0.26	0.73	0.54	0.41	0.5	0.63	0.25	0.24	0.26	0.5
5	0.39	0.49	0.3	0.68	0.58	0.53	0.28	0.66	0.27	0.33	0.26	0.54
6	0.38	0.19	0.35	0.63	0.46	0.52	0.098	0.64	0.35	0.37	-0.086	0.45
7	0.37	0.23	0.49	0.69	0.51	0.54	0.039	0.53	0.4	0.4	-0.016	0.28
8	0.48	0.3	0.43	0.74	0.42	0.7	0.19	0.51	0.38	0.21	0.25	0.43
9	0.23	0.14	0.58	0.76	0.59	0.63	-0.41	0.16	0.22	0.4	-0.041	0.12
10	0.044	0.26	0.43	0.73	0.71	0.5	-0.46	0.16	0.39	0.24	0.0017	0.21
11	0.29	0.28	0.45	0.54	0.5	0.67	0.17	0.59	0.43	0.38	-0.08	0.5
12	0.16	0.52	0.54	0.45	0.48	0.51	0.18	0.5	0.51	0.5	0.0044	0.18
13	0.32	0.38	0.52	0.65	0.56	0.7	0.087	0.52	0.47	0.37	0.058	0.34
14	0.41	0.36	0.44	0.64	0.57	0.75	0.086	0.51	0.33	0.18	0.28	0.48
15	0.13	0.49	0.54	0.46	0.58	0.52	0.1	0.35	0.41	0.49	0.23	0.089
16	0.47	0.32	0.48	0.59	0.52	0.65	0.024	0.51	0.38	0.25	0.27	0.18
17	0.11	0.39	0.46	0.48	0.61	0.52	-0.03	0.31	0.35	0.46	0.35	0.15
18	0.074	0.21	0.47	0.68	0.68	0.64	-0.28	0.27	0.27	0.44	0.3	0.15
19	-0.06	0.14	0.46	0.65	0.76	0.49	-0.36	0.38	0.34	0.27	0.28	0.18
20	0.06	0.022	0.51	0.74	0.77	0.76	-0.39	0.31	0.12	0.37	0.25	0.014
21	0.16	0.39	0.43	0.52	0.55	0.77	0.45	0.55	0.44	0.33	0.099	0.37
22	0.16	0.52	0.46	0.61	0.59	0.79	0.34	0.49	0.47	0.33	0.16	0.24
23	0.017	0.48	0.45	0.41	0.68	0.61	0.56	0.45	0.46	0.47	0.42	0.085
24	0.25	0.44	0.48	0.56	0.64	0.75	0.38	0.51	0.46	0.37	0.32	0.12
25	0.045	0.39	0.32	0.38	0.69	0.52	0.32	0.49	0.43	0.44	0.5	0.18
26	0.19	0.19	0.29	0.56	0.71	0.32	-0.24	0.58	0.49	0.21	0.51	0.15
27	0.39	0.061	0.37	0.5	0.64	0.47	-0.19	0.56	0.42	0.038	0.42	-0.033
28	0.17	0.53	0.017	0.62	0.48	0.78	0.56	0.53	0.43	0.31	0.22	0.22
29	0.27	0.37	0.25	0.62	0.63	0.78	0.53	0.53	0.39	0.34	0.34	0.11

(d) ***Evaluate your results by comparing the estimated HRFs, estimated pBOLD signal, and average measured pBOLD signal***

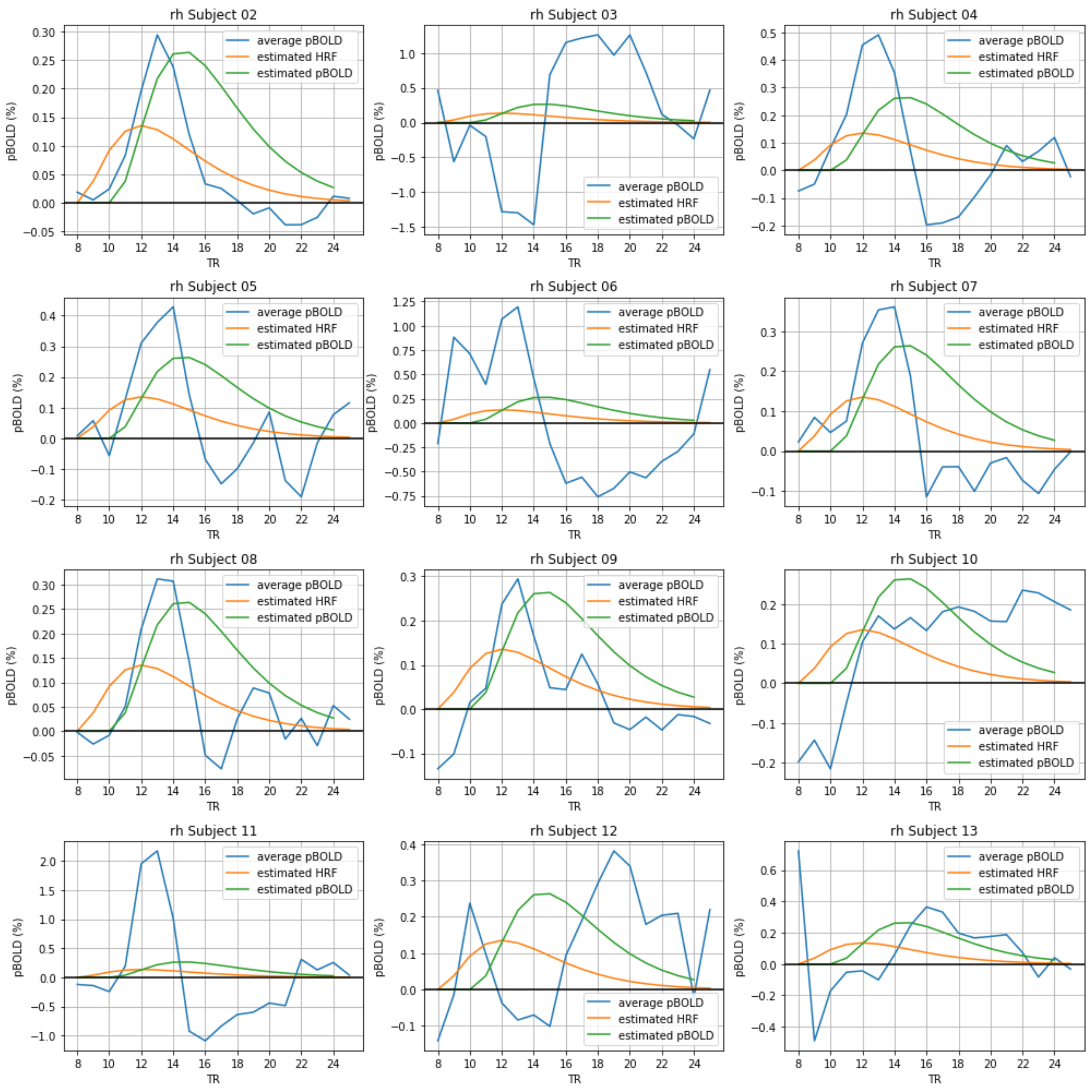
I attach a file of estimated HRF, estimated pBOLD by convolving HRF with neural boxcar, and real value of average measured pBOLD signal.



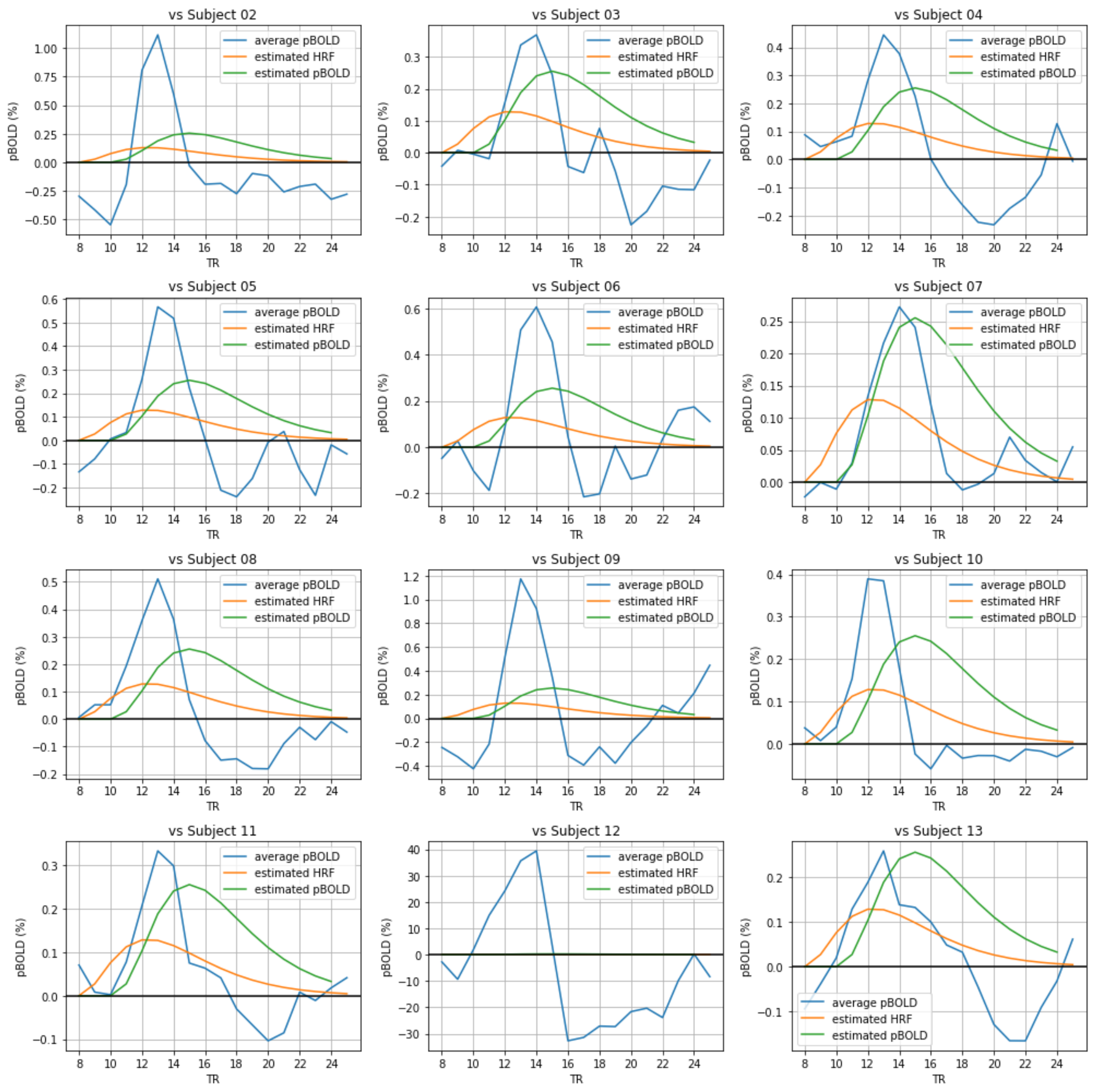
Auditory Task



Left Hand Task



Right Hand Task



Visual Task

[3] Using a flexible model of HRF (two gamma functions)

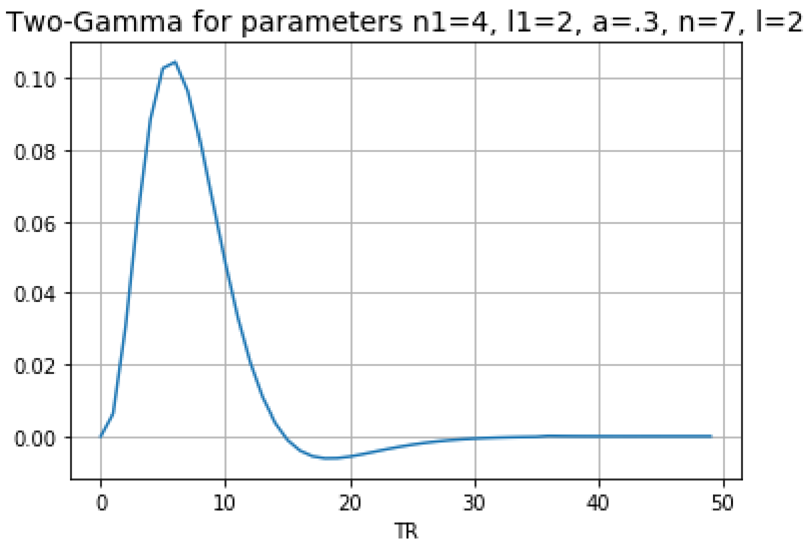
Follow the similar steps in [2] with difference in the adjustable parameters for the two gamma functions (i.e., T_0, n_1, λ_1, a, n_2, λ_2***)***

a. ***Set parameter sets.***

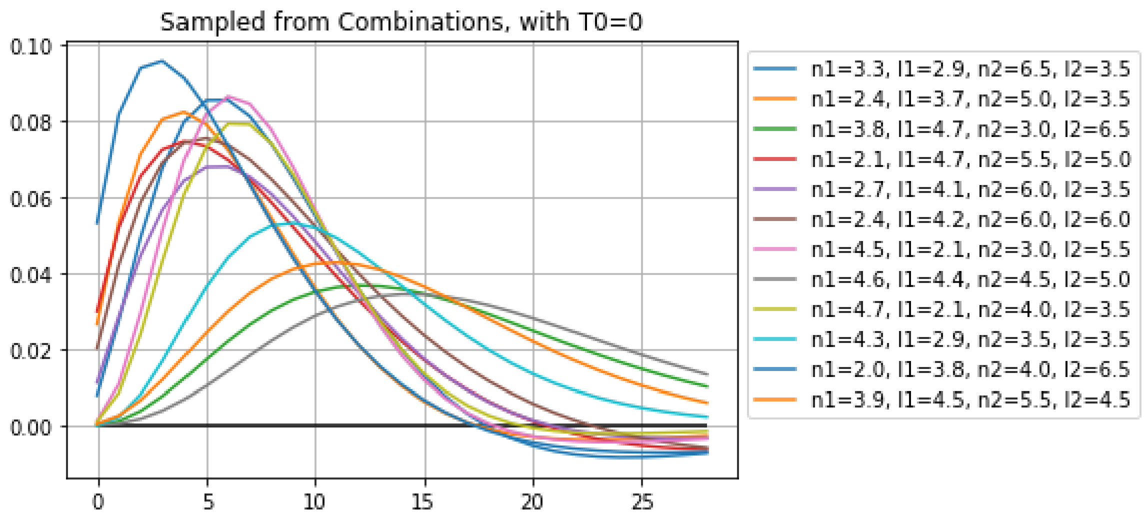
For T_0 and a, I fixed them with 0 and 0.3 respectively since our onset is fixed and damping is better dealt with n_2 and λ_2.

$$\Gamma_2(t; T_0, n_1, \lambda_1, a, n_2, \lambda_2) = \frac{(t-T_0)^{n_1-1}}{\lambda^{n_1}(n_1-1)!} - a\frac{(t-T_0)^{n_2-1}}{\lambda^{n_2}(n_2-1)!}$$

Since second term(undershoot) should have its peak on larger TR, I find n_2, λ_2 bigger. Therefore I set the candidates around — n_1 from 2 ~ 5, λ_1 from 2 ~ 5, n_2 from 3 ~ 7, λ_2 from 3 ~ 7.



As we

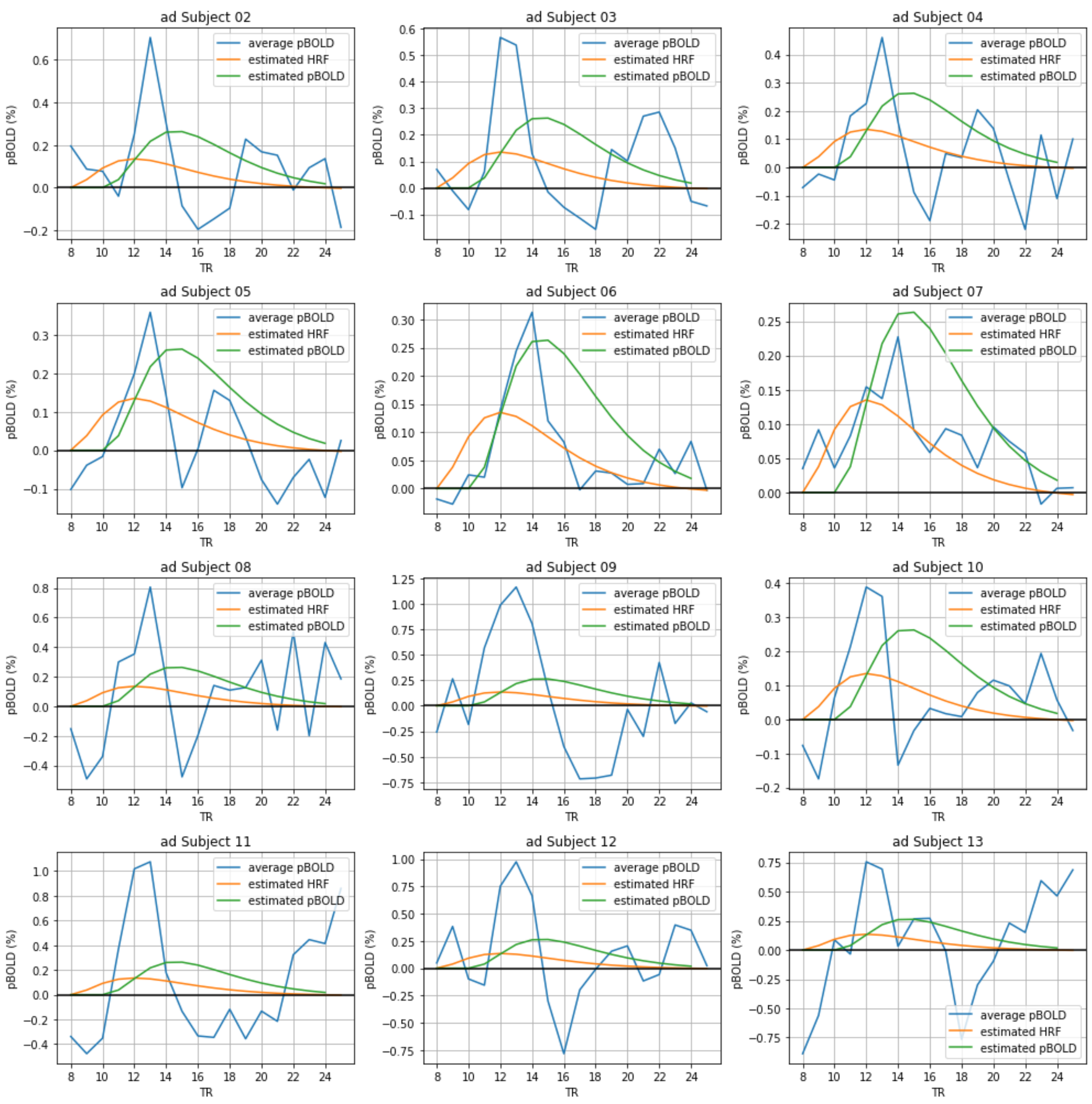


b, c ***Calculate Pearson Correlation Coefficients and choose the best one.***

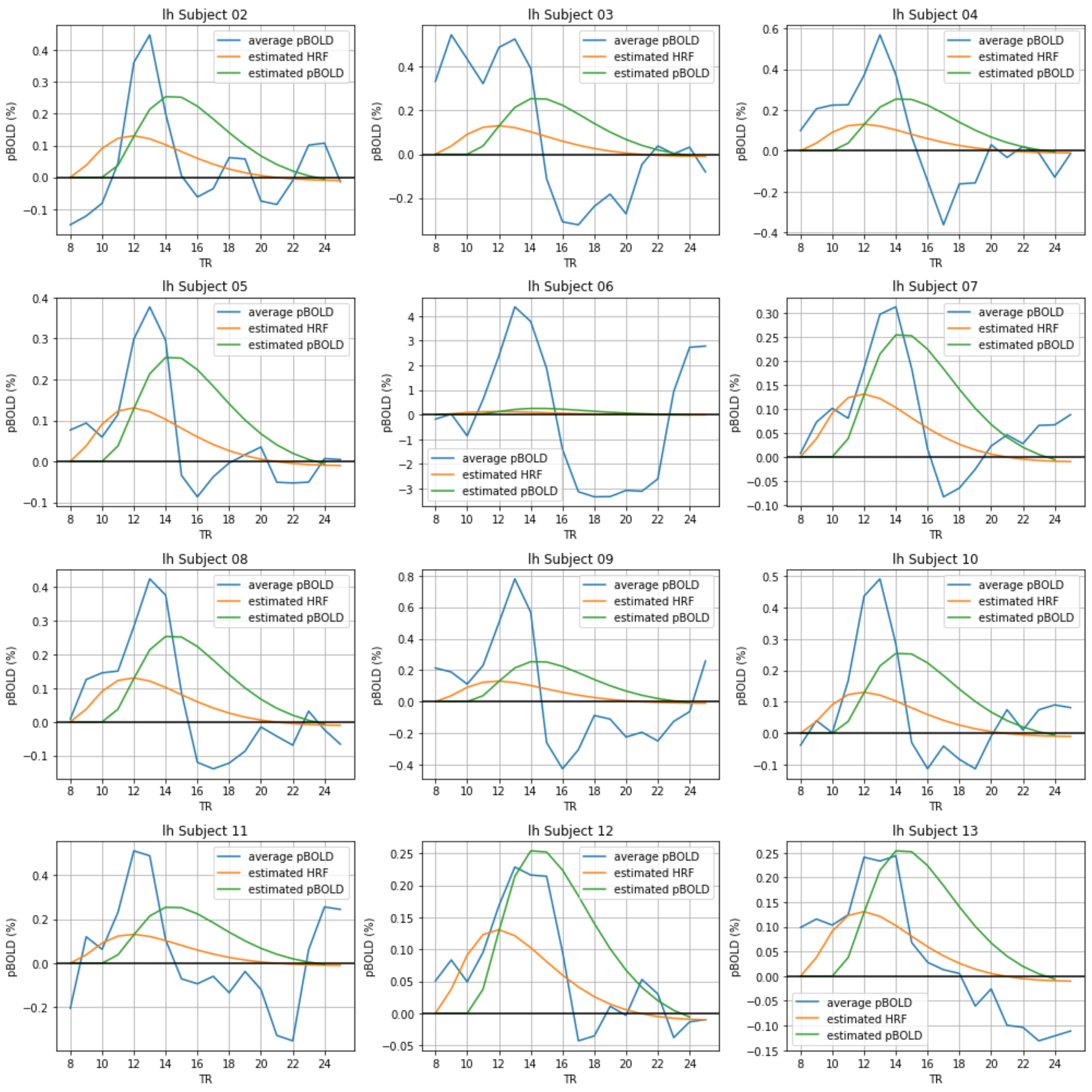
Two-gamma Correlation set, with ▪ and ▪

Aa Task	# Correlatoin	# n1	# lambda1	# n2	# lambda2
Auditory	0.3429	3	2	5	6.5
Left hand	0.5687	3	2	4	4

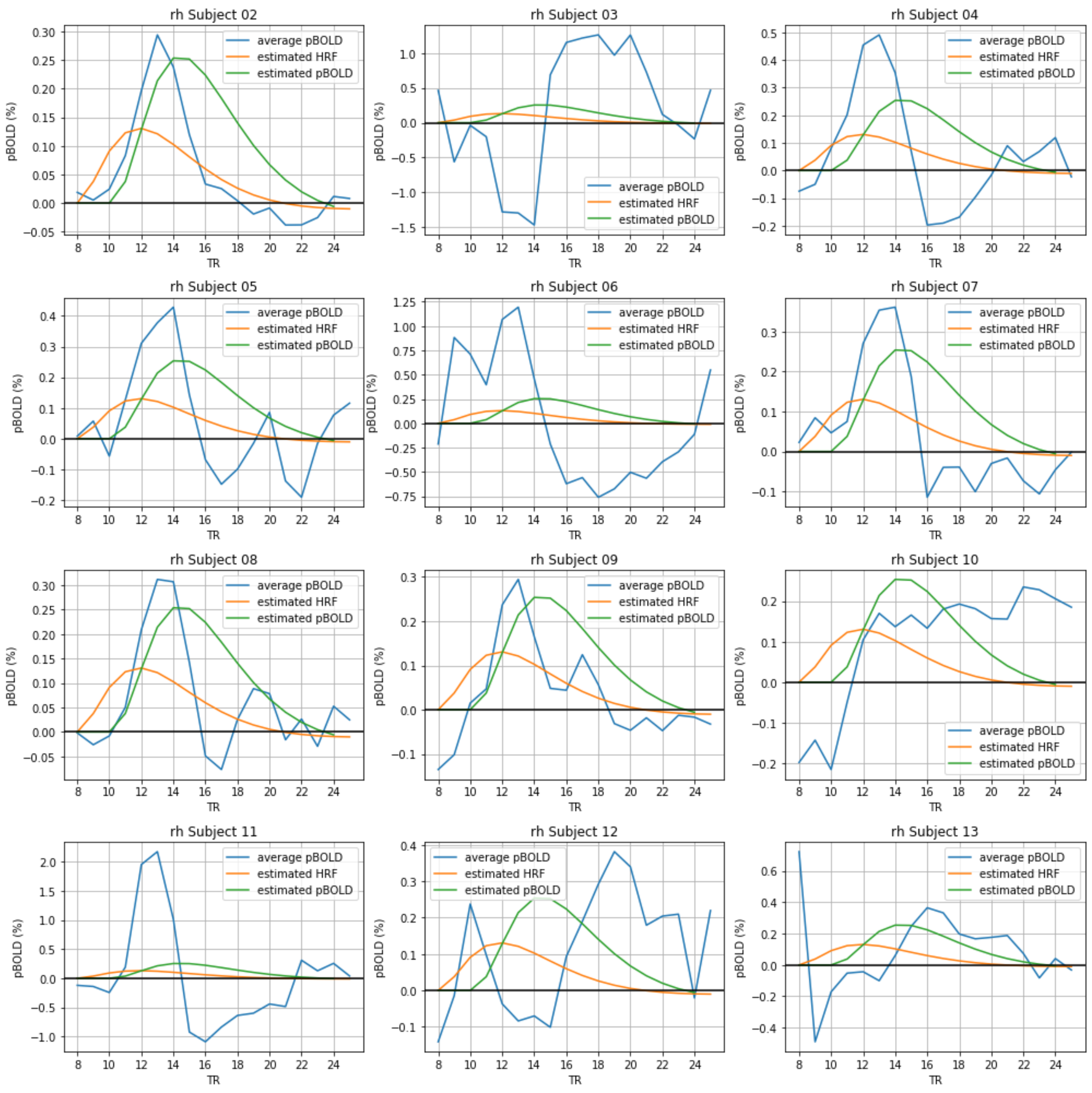
Task	Correlatoin	n1	lambda1	n2	lambda2
Right hand	0.489	3	2	4	4
Visual	0.5907	3.5	2	4	4



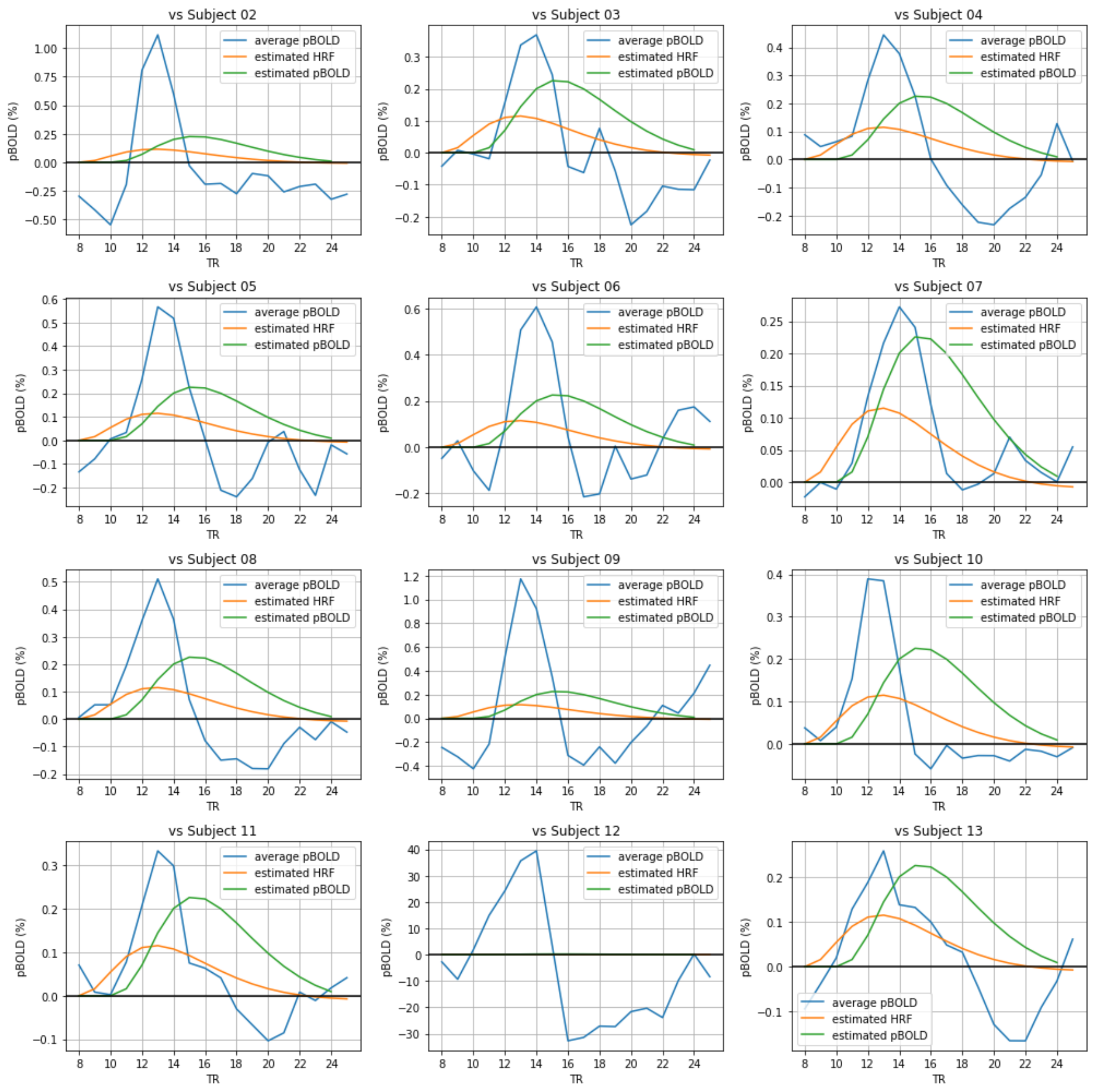
Auditory Task with two-gamma function estimation



Left Hand Task with two-gamma function estimation



Right Hand Task with two-gamma function estimation


Visual Task with two-gamma function estimation

[4] Based on the model with three fixed gamma functions (Friston et al., 1998) using least-squares (LS) algorithm

(a) ***Study the LS algorithm and show your work***

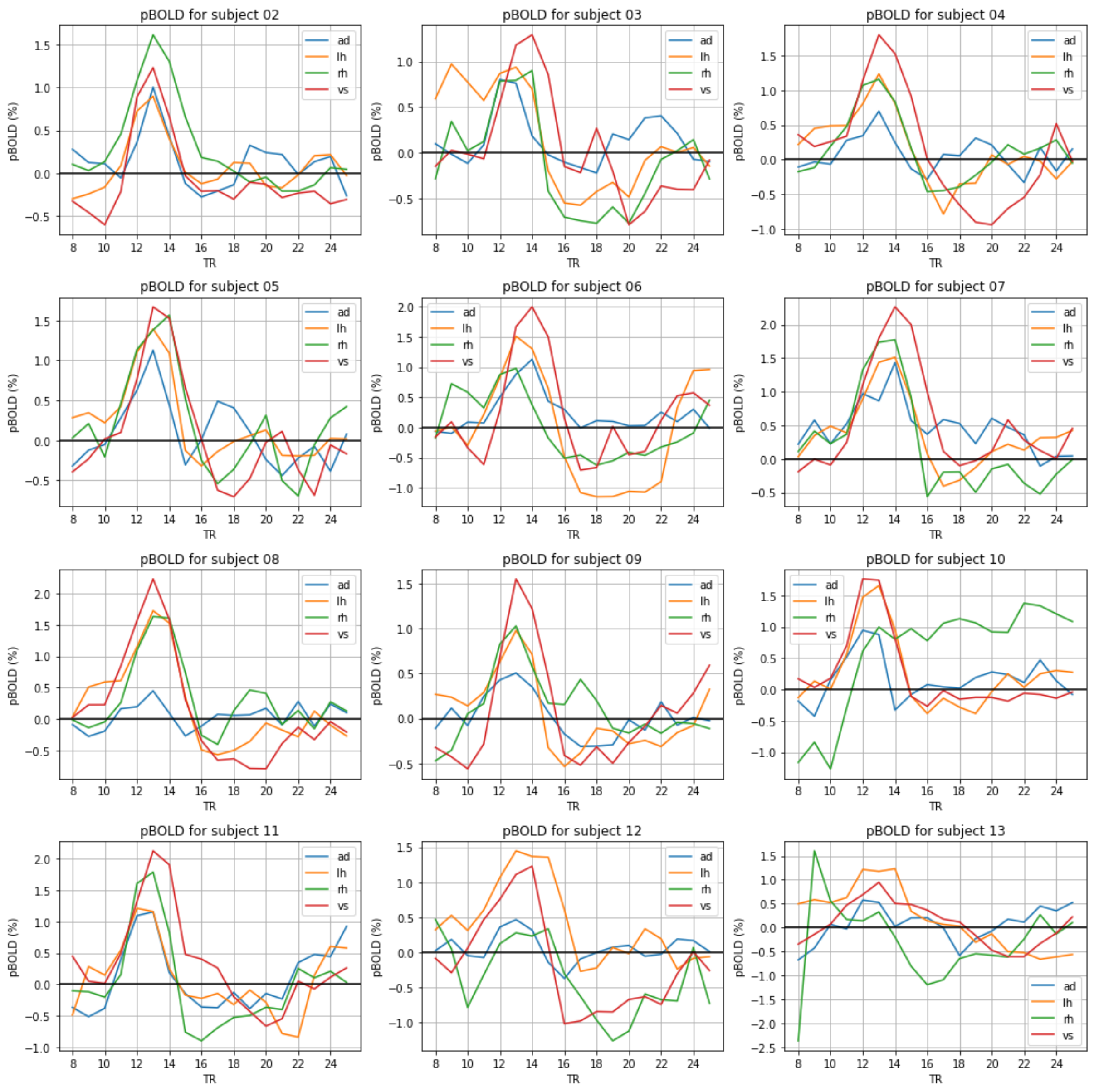
So-called LS algorithm is to use Squared error as a loss or reference of direction to where parameters should head to. We can either take this squared error as — 1. Checking all of combinations' SSE and choose the lowest one or 2. Take Gradient descent method using SSE as a loss function.

I take Gradient descent method to find the best parameters. Detailed works are on the codes.

```
SSE = lambda true, pred: sum(np.square(true - pred)) / len(true)

def partial_difference_quotient(func, v, i, h=1e-9):

    w = [v_j + (h if j==i else 0) for j, v_j in enumerate(v)]
    return (func(w) - func(v)) / h

def gradient_estimate(func, v, h=1e-9):

    return [partial_difference_quotient(func, v, i, h) for i, _ in enumerate(v)]


class LeastSquares:

    def __init__(self, epochs, task, gamma_type, init_params, lr, verbose=False):

        self.epochs = epochs
        self.task   = task
        self.gamma = one_gamma if gamma_type == 1 else two_gamma
        self.init_params = init_params
        self.lr = lr
        self.verbose = verbose
        self.true = sum(pBOLD(self.task, zfill(sub_id)) for sub_id in sub_ids)[8:
26]
        self.true /= sum(self.true)

    def fit(self):

        params = self.init_params
        self.losses = []
        for e in range(self.epochs):

            pred = self.gamma(params)
            loss = SSE(self.true, pred)

            params = list(map(lambda w, g: w - g * self.lr, params, gradient_esti
mate(self.mse, params)))
            params = [0] + params[1:]

            if e % 100 == 0 and self.verbose:
                self.losses.append(loss)
                print(f'{str(e):<5} th Epochs')
#                 print(f'Prediction:: {pred}')
                print(f'LOSS:: {loss}, Gradient:: {params}')
                print('\n')

        self.params = params

    def mse(self, params):

        true = self.true
        pred = self.gamma(params)
        return np.sum(np.square(true - pred)) / len(true)


    def plot_loss(self):
        plt.plot(self.losses)
        plt.title(f'MSE Loss among {self.epochs} epochs', fontsize=16)
        plt.xlabel('Epochs')
        plt.ylabel('MSE Loss')
        plt.grid()


    def plot_gamma(self, params=None):

        if params is None:
            params = self.params

        if len(params) == 3:
            T0, n, l = params
            ttl = f'Task {self.task} Params of  T0={T0}, n={n:.3f}, l={l:.3f}'

        elif len(params) == 6:
            T0, n1, l1, a, n2, l2 = params
            ttl = f'Task {self.task} Params of n1={n1:.3f}, l1={l1:.3f}, a={a:.3
f}, n2={n2:.3f}, l2={l2:.3f}'


        estimated_hrf = self.gamma(params)
        plt.plot(self.true, label='average pBOLD')
        plt.plot(self.gamma(params), label='estimated HRF')
        plt.plot(np.convolve(boxcar[8:26], estimated_hrf)[:17], label='estimated
 pBOLD')

        plt.title(ttl)
        plt.xlabel('TR')
        plt.grid()
        plt.legend()
```

(b) ***Calculate average pBOLD signal across all the voxels ('avg_pBOLD')***

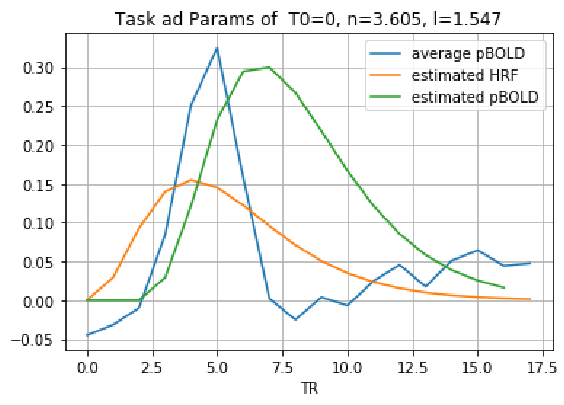
There is a bigger picture for this on the script(.ipynb) or the github.

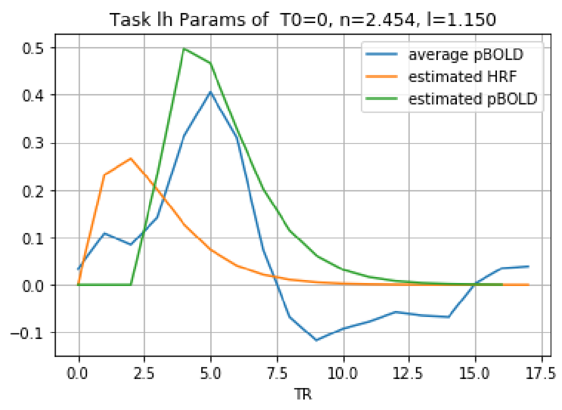


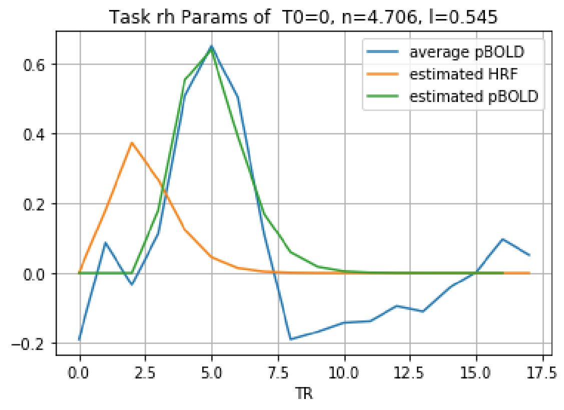
(c) ***Apply the LS algorithm to this 'avg_pBOLD' signal and estimate the θ parameters of this model***

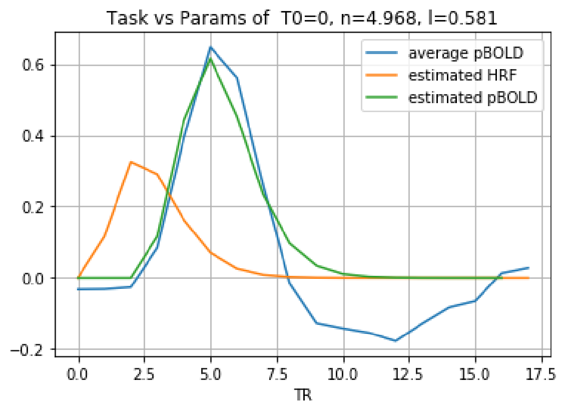
Results below.

One-Gamma



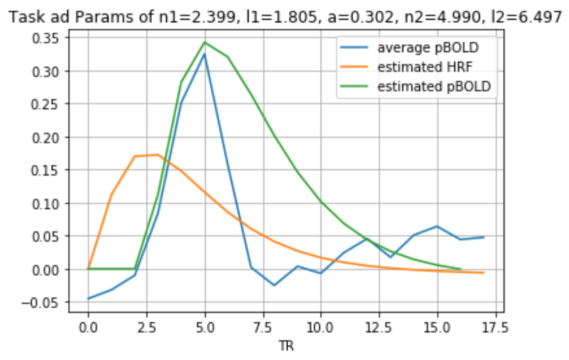


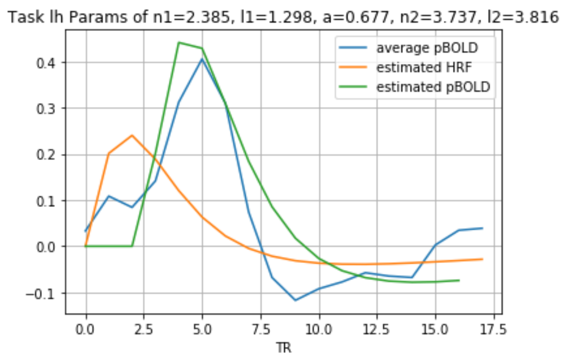


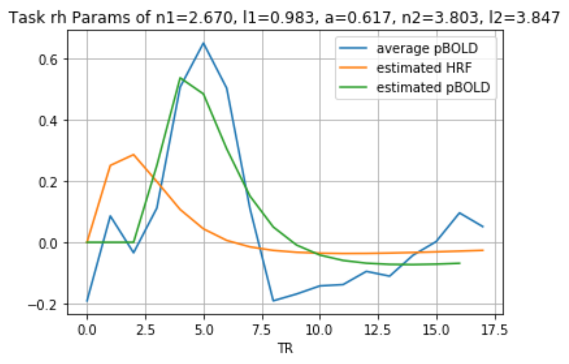


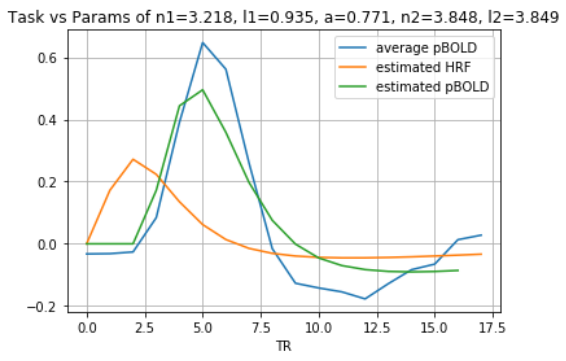
One-gamma SSE, with ᴅ

Task	Lowest SSE	n1	l1
Auditory	0.00583	3.605	1.547
Left hand	0.01277	2.454	1.15
Right hand	0.4547	4.706	0.545
Visual	0.041911	4.968	0.581









Two-gamma SSE, with

Task	Lowset SSE	n1	lambda1	a	n2	lambda2
Auditory	0.00592	2.399	1.805	0.302	4.99	6.497
Left hand	0.012265	2.385	1.298	0.677	3.737	3.816
Right hand	0.045004	2.67	0.983	0.617	3.803	3.847
Visual	0.04039	3.218	0.935	0.771	3.848	3.849

[5] Discuss overall results obtained from [1] to [4].

Don't understand why Deconvolution didn't work still.

I've tried my best though...

Confused between HRF and convolved HRF to make estimated pBOLD

Convolved with HRF and $N(t)$, now I get pBOLD that looks fine-ish. But peak value of the true pBOLD and estimated pBOLD has some delay. Maybe it's due to setting TR=10, 11 as stimulus, since we have 3 seconds of stimulus only. Some subjects worked perfect while some didn't.

SSE is more precise than Correlation method

But the loss has too complex parameter space of loss function. This looks

Damping not clearly seen on Two-gamma Function

It's sad that those dampings doesn't visually look great...

No AFNI

It would be better to use AFNI command while doing the assignment but I couldn't. First, I couldn't find command that I should use here(maybe there was but it was just me who couldn't find the right command). Second, I was using

Windows so I installed it with WSL on my desktop but since COVID19 is swarming again, I couldn't use my desktop anymore...

[Note] Common to all of the problems

- Note that TR = 2 s, so measured BOLD signals were acquired in every 2 s
- Perform the analyses for each of the four tasks
- Summarize the estimated parameters of the HRFs: e.g., for each of the four tasks across the subjects and/or for each subject across the four tasks preferably using some kinds of graphs for visualization
- Visualize the estimated HRFs for each of the four tasks and for each of the subjects (e.g., refer to p.17 of the lecture note, "*CH. 8 Signal, Noise, and Preprocessing of fMRI Data*")
- Discuss your results/figures at least with a few lines of text
- Use AFNI and MATLAB/Python to perform the analyses