VASO BOLD-correction

A very nice description of the problem by Denis Chaimow can be found in this document. Here, I am investigating the issue in my own data.

In [2]:

```
import numpy as np
import glob
import nibabel as nb
import pandas as pd
import matplotlib.pyplot as plt
from nilearn import plotting
import seaborn as sns
from scipy.interpolate import interp1d
```

Find TR of our acquisition

Later, we are going to make graphs based on our TR times. Therefore, getting the TR of our acquisition makes sense from the start. For this, I just reused old code that reads the log-files and computes the average TR. It also gives the duration of the individual triggers, which illustrates the problem of assumetry.

In [4]:

```
f = open('/media/sebastian/elements/data/s1_anfunco_analysis/log-files/sub14_run1-mPTS_2x3ch_2019_SlowBlock_8TRs_t3_1A.log')
df = pd.read_csv(f, skiprows=2, sep='\s+',usecols=["Type", "Code", 'Time', 'TTime'])
df.rename(columns={'Code': 'start_time'}, inplace=True)

first_pulse_time = int(df.at[1,'start_time'])
last_pulse_time = int(df.at[df.last_valid_index(),'start_time'])

tr = df.where(df['Type'] == '44')
tr = tr.dropna()
tr = tr.drop(columns = ['Type', 'Time', 'TTime'])

tr_list = [int(i) for i in tr['start_time']]
tr_arr = np.asarray(tr_list)
tr_arr = tr_arr - first_pulse_time

tr_durations = []

for i in range(len(tr_arr)):

    if i >= 1:
        duration = tr_arr[i] - tr_arr[i-1]
        tr_durations.append(duration)

trigger1 = np.mean(tr_durations[::2])
trigger2 = np.mean(tr_durations[1::2])

tr = (trigger1+trigger2)/10 # global TR in ms
```

Which gives us the following:

In [6]:

```
print(f'- the first trigger duration is: {trigger1} ms')
print(f'- the second trigger duration is: {trigger2} ms')
print(f'- the global TR is: {tr} ms')
```

```
- the first trigger duration is: 22559.575 ms
- the second trigger duration is: 16014.718592964824 ms - the
  global TR is: 3857.4293592964823 ms
```

I started with copying the temporally upsampled nulled and notnulled data to a new folder.

In [5]:

```
sub10Path = '/media/sebastian/Data/BOCO_shift/sub10/'
BoldTimeCourse = sub10Path + 'BOLD_intemp.nii'
VasoTimeCourse = sub10Path + 'Nulled_intemp.nii'
```

In order to reduce processing time and storage space, I cut irrelevant parts of the data and then do the standard BOLD-correction.

In [6]:

```
%%bash -s "$sub10Path" "$BoldTimeCourse" "$VasoTimeCourse"

echo $1
cd $1
pwd
fslroi "$1T1w.nii" $1T1w_trunc 35 20 85 20 10 3
fslroi $2 Notnulled_intemp_trunc 35 20 85 20 10 3
fslroi $3 Nulled_intemp_trunc 35 20 85 20 10 3
LN_BOCO -Nulled Nulled_intemp_trunc.nii.gz -BOLD Notnulled_intemp_trunc.nii.gz -
output trunc
```

```
/media/sebastian/Data/BOCO_shift/sub10/
/media/sebastian/Data/BOCO_shift/sub10
=======================
LAYNII v2.0.0
Compiled for Linux 64
=======================
LN_BOCO

    File name: Nulled_intemp_trunc.nii.gz
    Image details: 20 X | 20 Y | 3 Z | 400 T
    Voxel size = 0.753086 x 0.753086 x 1.29
    Datatype = 4

    File name: Notnulled_intemp_trunc.nii.gz
    Image details: 20 X | 20 Y | 3 Z | 402 T
    Voxel size = 0.753086 x 0.753086 x 1.29
    Datatype = 4

    Writing output as:
 trunc_VASO_LN.nii  Finished.
```

I then load the truncated timecourses as arrays for easy handling.

In [7]:

```
BoldTimeCourseTrunc = sub10Path + 'Notnulled_intemp_trunc.nii.gz'
VasoTimeCourseTrunc = sub10Path + 'Nulled_intemp_trunc.nii.gz'


Bold = nb.load(BoldTimeCourseTrunc)
Vaso = nb.load(VasoTimeCourseTrunc)

BoldData = nb.load(BoldTimeCourseTrunc).get_fdata()
VasoData = nb.load(VasoTimeCourseTrunc).get_fdata()


BoldData = BoldData[:,:,:,:400]
VasoData = VasoData[:,:,:,:400]
```

To see whether we the truncation worked and we loaded the correct files, lets look at the shapes of the arrays. Based on the values we gave the fslroi command, the shape should be (50, 50, 15, x)

In [8]:

```
print(BoldData.shape)
print(VasoData.shape)
```

```
(20, 20, 3, 400)
(20, 20, 3, 400)
```

Checks out, cool.

Because I had some issues fith handling files between fsl/laynii and the python environment, I just save the 'array'-like images to new images. I think the problems had to do with how the dimensions are saved. While the fslroi-truncated data still overlaps with the original in FSLeyes, they do not when I just truncate the matrices loaded in nibabel.

In [9]:

```
img = nb.Nifti1Image(VasoData, header=Vaso.header, affine=Vaso.affine)
img.to_filename(f'{sub10Path}nulled_intemp_trunc.nii')

img = nb.Nifti1Image(BoldData, header=Bold.header, affine=Bold.affine)

img.to_filename(f'{sub10Path}notnulled_intemp_trunc.nii') In [10]:
```

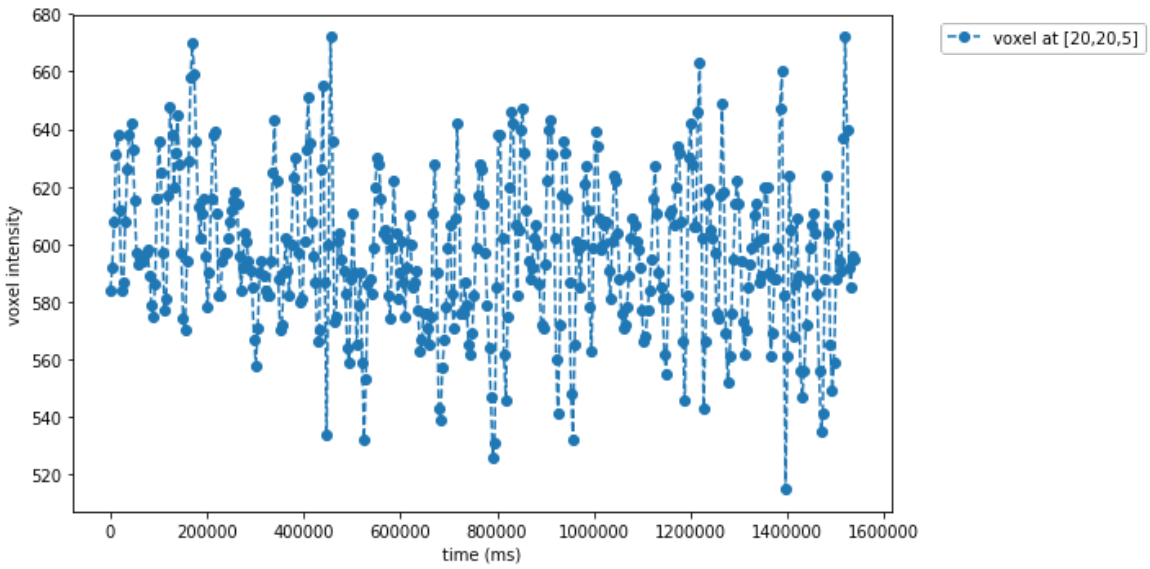
```
BoldTimeCourseTrunc = sub10Path + 'notnulled_intemp_trunc.nii'
VasoTimeCourseTrunc = sub10Path + 'nulled_intemp_trunc.nii'


Bold = nb.load(BoldTimeCourseTrunc)
Vaso = nb.load(VasoTimeCourseTrunc)

BoldData = Bold.get_fdata()
VasoData = Vaso.get_fdata()


BoldData = BoldData[:,:,:,:400]
VasoData = VasoData[:,:,:,:400]
```

We can then plot a single voxel's timecourse, just to see whether it makes sense. For this, we first create an array with the time of the TR in ms and then plot it.

In [11]:

```
x = np.arange(0, BoldData.shape[-1]*tr,tr)

fig = plt.figure(tight_layout = True, figsize=(10,5))

plt.plot(x, BoldData[19][19][0], 'o--', label = 'voxel at [20,20,5]')

plt.ylabel('voxel intensity')
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')
plt.xlabel('time (ms)')
plt.show()
```

```
/home/sebastian/anaconda3/lib/python3.7/site-packages/matplotlib/fig
ure.py:2299: UserWarning: This figure includes Axes that are not com
patible with tight_layout, so results might be incorrect.
warnings.warn("This figure includes Axes that are not compatible "
```



To try out the temporal shifting, we will first do it for the voxel we just plotted.

For this, we will use scipy's "interp1d" with cubic interpolation. Importantly, we have to set bounds_error to false as we want go beyond the boundaries of our original timepoints. The respective values should then be extrapolated.

In [12]:

```
dataValsInterpolated = interp1d(x, BoldData[19][19][0], kind='cubic', bounds_err
or=False, fill_value= 'extrapolate')
```

This class returns a funciton whose call method uses interpolation to find the value of new points.

To apply this, we have to create an array with the new time-points we want to sample from. For the sake of this test, we just shift forward by 2000ms. The new timepoints are then given to the interpolation function.

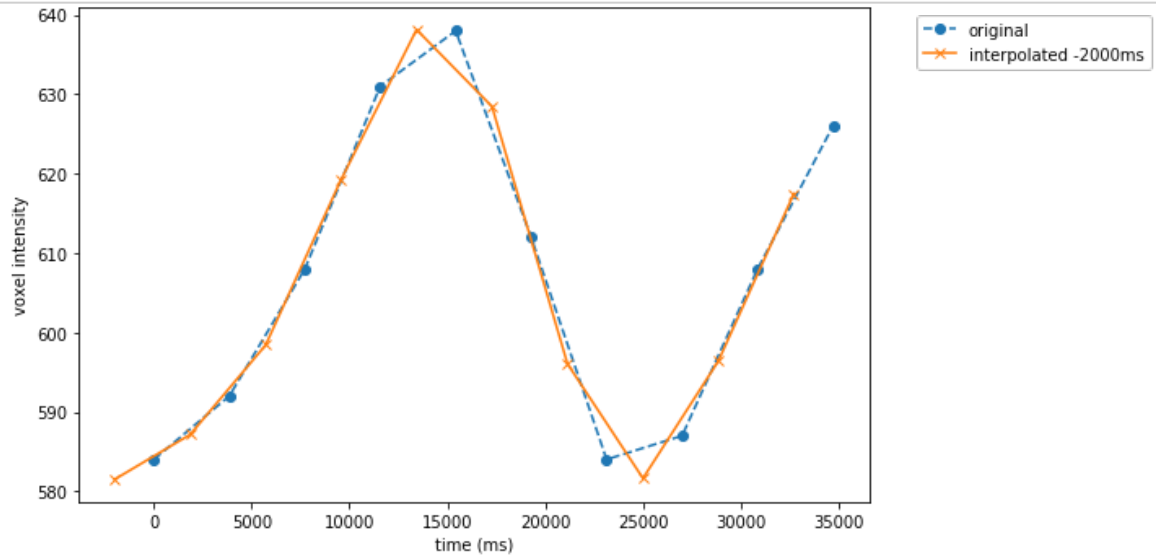
In [14]:

```
newTimePoints = x-2000 #subtract 2000 from each item in original array

shifted = dataValsInterpolated(newTimePoints) #apply interpolatoin function to n
ew timepoints
```

Because we won't see what happened if the entire timecourse is plotted, we just display the first 10.

In [18]:

```
fig = plt.figure(tight_layout = True, figsize=(10,5))

plt.plot(x[0:10], BoldData[19][19][0][0:10], 'o--', label = 'original')
plt.plot( newTimePoints[0:10], dataValsInterpolated(newTimePoints)[0:10], 'x-',
label = 'interpolated -2000ms')


plt.ylabel('voxel intensity')
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left') plt.xlabel('time
(ms)')
plt.savefig('/home/sebastian/Desktop/interpolatedTimecourse.png') plt.show()
```



Looks good to me. We have new timepoints in between the original ones, start earlier and end earlier.

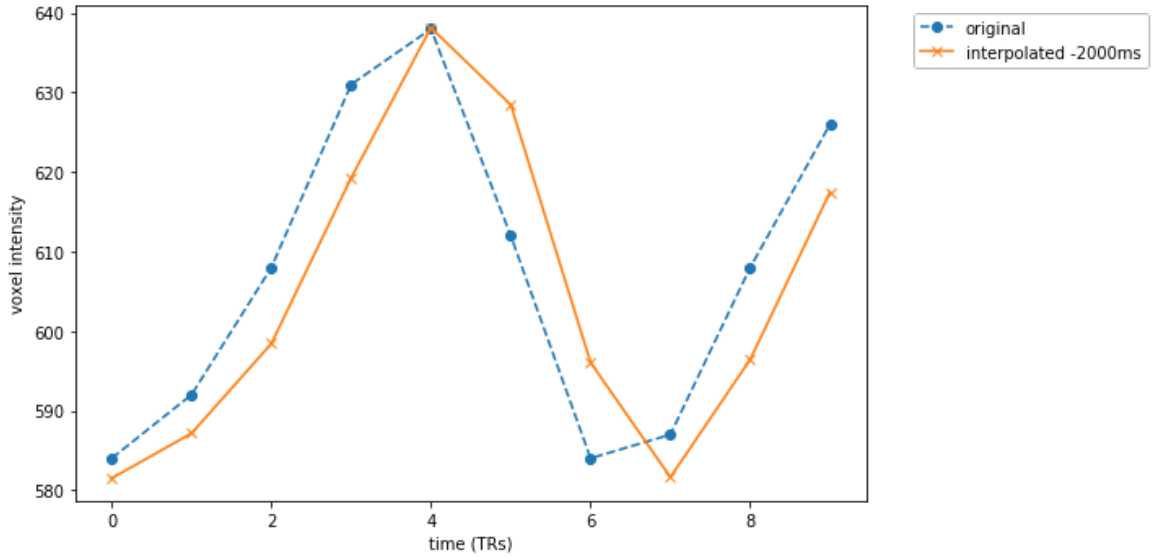
Checking this made a lot of sense for another issue as well. In event-ralated averages I made in earlier versions, it looked as if the 'negatively' shited (i.e. forward in time) data was actually shifted backward in time. This is nicely illustrated here.

In the plot above the curves overlap and the timepoints of the new data are "earlier". However, if we now save the data with respect to the first timepoint, the following happens.

In [20]:

```
first10OriginalTimePoints = BoldData[19][19][0][0:10]
first10ShiftedTimePoints = dataValsInterpolated(newTimePoints)[0:10]

xTest = np.arange(0,10) fig = plt.figure(tight_layout = True,

figsize=(10,5))
```

```
plt.plot(xTest, first10OriginalTimePoints, 'o--', label = 'original')
plt.plot(xTest,first10ShiftedTimePoints, 'x-', label = 'interpolated -2000ms')


plt.ylabel('voxel intensity')
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left') plt.xlabel('time
(TRs)')
plt.savefig('/home/sebastian/Desktop/interpolatedTimecourseRecentered.png')
plt.show()
```



Moving on

With this test looking alright, we can now define a range of shifts we want to do.

We do this with an array containing values for shifts of up to 2 TRs in both directions in steps of 200ms.

In [7]:

```
shiftVals = np.arange(-2*tr,2*tr,200)
```

Because I also want to store the resulting files, I need the shifts as srtings without the nagitive sign. I am not very happy with this string conversion and everyting, but it works for now.

In [8]:

```
shifts = np.arange(-2*tr,2*tr,200)
shifts = [str(int(i)) for i in shifts]

for i in range(len(shifts)):
    if not shifts[i][0]=='-':
        shifts[i] = 'pos' + shifts[i].zfill(4)
    if shifts[i][0]=='-':
        shifts[i] = shifts[i][1:]
        shifts[i] = 'neg' + shifts[i].zfill(4)
```

With the shifts ready, I create a new data-array with the same dimensions as the original data. I then loop over each voxel and use the procedure outlined above to interpolate its timecourse to the new shift.

In [ ]:

```
for shiftVal,shift in zip(shiftVals, shifts):
 print(f'shifting the data {shiftVal} ms')

    newData = np.zeros(BoldData.shape)

    for n in range(BoldData.shape[0]):
 for i in range(BoldData.shape[1]):
 for j in range(BoldData.shape[2]):
 timeCourse = BoldData[n][i][j]

                x = np.arange(0, BoldData.shape[-1]*tr,tr)
                dataValsInterpolated = interp1d(x, timeCourse, kind='cubic',boun
ds_error=False, fill_value= 'extrapolate')

                newTimePoints = x+shiftVal
                shifted =
 dataValsInterpolated(newTimePoints)
 newData[n][i][j] = shifted
    #Here, I save the shifted BOLD-data.     img =
 nb.Nifti1Image(newData, header=Vaso.header, affine=Vaso.affine)
 img.to_filename(f'{sub10Path}{shift}_BOLD.nii')

    # Doing the BOLD-correction manually. Laynii gave unwanted results when work
ing with data that was loaded as matrix and saved as .nii again
    manualBoco = np.divide(VasoData, newData)

    # Because FSL for some reason does not like values between 0 and 1 in their
 GLM, we multiply the entire thing by 100
 manualBoco = manualBoco*100

    # Running FEAT from the command line didn't work because the design matrix h
as more rows than there are
    # volumes. Therefore, I just copied a few at the end.     arrs =
 [manualBoco[:,:,:,i] for i in range(manualBoco.shape[-1])]     for n
 in range(4):         arrs.append(manualBoco[:,:,:,-1])     manualBoco
 = np.stack(arrs, axis=3)

    # The data is then saved.     img = nb.Nifti1Image(manualBoco,
 header=Vaso.header, affine=Vaso.affine)
 img.to_filename(f'{sub10Path}{shift}_VASO_LN.nii')
```

With the data shifted, I created a template for our FEAT analysis. Here, we can then just exchange the inand outputs and save a .fsf file for each shift.

In [86]:

```
# Set this to the directory where you'll dump all the fsf files
# May want to make it a separate directory, because you can delete them all o
#   once Feat runs
fsfdir="/media/sebastian/Data/BOCO_shift/sub10/fsfs"
```

```
for shift in shifts:    replacements =
 {'INFILE':f'/media/sebastian/Data/BOCO_shift/sub10/{shift}_VA
SO_LN.nii', 'OUTDIR': f'/media/sebastian/Data/BOCO_shift/sub10/{shift}_VASO_LN'}
            with open(f"{fsfdir}/template.fsf") as infile:
 with open(f"{fsfdir}/shifted_{shift}.fsf", 'w') as outfile:
 for line in infile:                for src, target in
 replacements.items():
                    line = line.replace(src, target)
 outfile.write(line)
```

To automatically run all GLMs, we can create a list in python and then loop over it in bash.

First the list:

In [87]: fsfs =

```
sorted(glob.glob(f'{fsfdir}/shifted_*.fsf'))
```

Then the looping.

In [ ]:

```
%%bash -s "{" ".join(fsfs)}"

for fsf in $1
do
wait; feat $fsf &
done
```

Next to the VASO data (BOLD-corrected with different shifts), I also ran the GLM for the BOLD-data. With this, I want to draw an ROI in which we later extract our VASO z-scores

To draw the ROI, I created a T1w image in the dimensions of our truncated data.

In [91]:

```
%%bash
cd /media/sebastian/Data/BOCO_shift/sub10/ echo
"calculating T1 in EPI space"
3dTcat -prefix combined.nii  moco_nulled.nii moco_notnulled.nii -overwrite
3dTstat -cvarinv -overwrite  -prefix T1w.nii combined.nii

rm combined.nii fslroi "$1T1w.nii" $1T1w_trunc 35 20 85

20 10 3 calculating T1 in EPI space
```

```
++ 3dTcat: AFNI version=AFNI_21.1.07 (May 11 2021) [64-bit]
++ elapsed time = 2.0 s
++ 3dTstat: AFNI version=AFNI_21.1.07 (May 11 2021) [64-bit]
++ Authored by: KR Hammett & RW Cox
++ Oblique dataset:/media/sebastian/Data/BOCO_shift/sub10/combined.n
ii is 27.697283 degrees from plumb. ++ Output dataset ./T1w.nii
```

All relevant data was then upsampled. Strictly speaking, this is not necessary. However, it enables us to look maybe look at depth depepndent effects.

We took care of the:

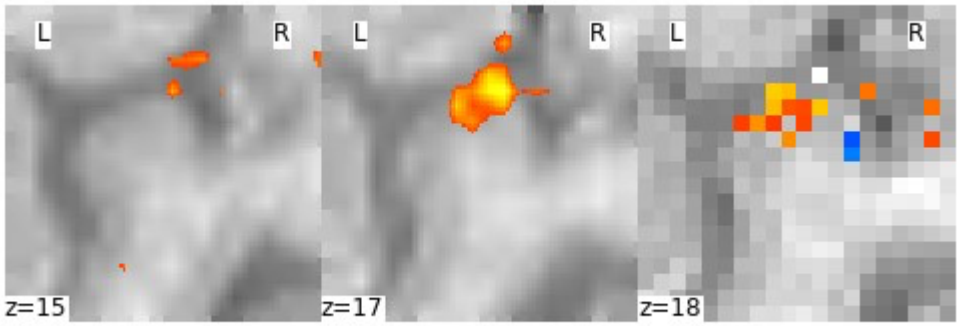
- T1w image in truncated space
- the zstat images from all VASO GLMs with shifted BOLD-correction
- the GLM with the BOLD data

In [92]:

```
%%bash -s "{" ".join(shifts)}" cd

/media/sebastian/Data/BOCO_shift/sub10

delta_x=$(3dinfo -di T1w_trunc.nii.gz) delta_y=$(3dinfo
-dj T1w_trunc.nii.gz) delta_z=$(3dinfo -dk
T1w_trunc.nii.gz)
sdelta_x=$(echo "((sqrt($delta_x * $delta_x) / 5))"|bc -l) sdelta_y=$(echo
"((sqrt($delta_y * $delta_y) / 5))"|bc -l) sdelta_z=$(echo
"((sqrt($delta_z * $delta_z) / 1))"|bc -l)

3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix scal
ed_T1w_trunc.nii -input T1w_trunc.nii.gz

for shift in $1
do
cd "/media/sebastian/Data/BOCO_shift/sub10/"$shift"_VASO_LN.feat/stats"

3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix VASO
_scaled_D2.nii -input zstat1.nii.gz
3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix VASO
_scaled_D3.nii -input zstat2.nii.gz
3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix VASO
_scaled_D4.nii -input zstat3.nii.gz done cd
/media/sebastian/Data/BOCO_shift/sub10/BOLD.feat/stats

3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix VASO
_scaled_D2.nii -input zstat1.nii.gz
3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix VASO
_scaled_D3.nii -input zstat2.nii.gz
3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix VASO
_scaled_D4.nii -input zstat3.nii.gz
```

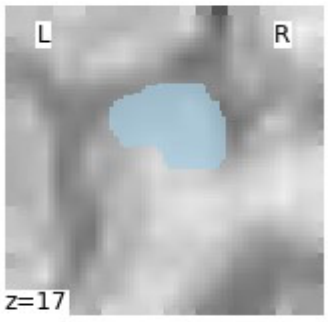
In [12]:

```
BOLDmap = '/media/sebastian/Data/BOCO_shift/sub10/BOLD.feat/stats/VASO_scaled_D
2.nii'
t1 ='/media/sebastian/Data/BOCO_shift/sub10/scaled_T1w_trunc.nii'



display = plotting.plot_stat_map(BOLDmap, t1, colorbar=False, threshold=4, displ
ay_mode = 'z', cut_coords = 3)
```



In [13]:

```
mask = '/media/sebastian/Data/BOCO_shift/sub10/VASO_scaled_D2_mask.nii.gz'

plotting.plot_roi(mask, bg_img=t1, cmap='Paired', display_mode = 'z', cut_coords
= 1)
```



From the mask, we then extract mean zscores from each GLM-result.

In [16]:

```
extracted = []

for shift in shifts:
    # Define files
    # Timecourse file
    FILE1 = f'/media/sebastian/Data/BOCO_shift/sub10/{shift}_VASO_LN.feat/stats/
VASO_scaled_D2.nii'
    # Load timecourse data
    nii1 = nb.load(FILE1)
    data = nii1.get_fdata()


    # For entire digit-ROIs
    FILE3 = mask
    nii3 = nb.load(FILE3)
    digit_mask = nii3.get_fdata()

    extracted.append(np.mean(data[digit_mask.astype(bool)]))
```

And plot it.

In [22]:

```
tickLocs = np.arange(0, len(dataframe['shift (ms)']), 10)
tickLocs
```

Out[22]:

```
array([ 0, 10, 20, 30, 40, 50, 60, 70])
```

In [30]:

```
formatted_list
```

Out[30]:

```
['-7715', '-5715', '-3715', '-1715', '285', '2285', '4285', '6285']
```

In [29]:

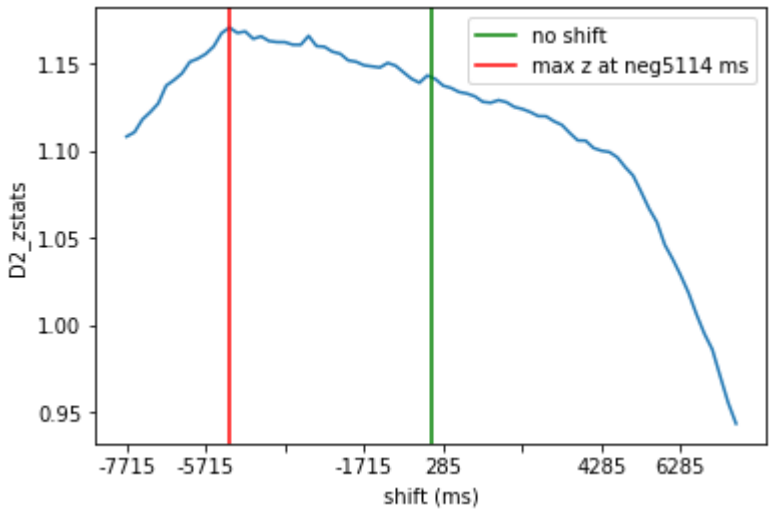
```
dataframe = pd.DataFrame({'shift (ms)': shifts, 'D2_zstats': extracted})

label_indices = np.linspace(0,len(dataframe['D2_zstats']),)
label_indices = [int(i) for i in label_indices] plot =
sns.lineplot(data=dataframe, x="shift (ms)", y="D2_zstats")

for ind, label in enumerate(plot.get_xticklabels()):
    if ind in label_indices:  # every 10th label is
 kept        label.set_visible(True)    else:
 label.set_visible(False)

         plt.axvline(77/2, 0,2.8, color = 'g', label=
'no shift')
plt.axvline(np.argmax(dataframe['D2_zstats'], axis=0), 0,2.8, color = 'r', label
= f"max z at {shifts[np.argmax(dataframe['D2_zstats'])]} ms")
```

```
# define tick locations and lenght
tickLocs = np.arange(0, len(dataframe['shift (ms)']), 10)
formatted_list = ["%.0f"%item for item in shiftVals[tickLocs]]
plt.xticks(tickLocs, formatted_list)


            plt.legend()
# plt.savefig('shiftVsZstat.jpg')
```

```
/home/sebastian/anaconda3/lib/python3.7/site-packages/numpy/core/fro
mnumeric.py:61: FutureWarning:  The current behaviour of
'Series.argmax' is deprecated, use 'idxmax' instead.
The behavior of 'argmax' will be corrected to return the positional
maximum in the future. For now, use 'series.values.argmax' or
'np.argmax(np.array(values))' to get the position of the maximum
row.  return bound(*args, **kwds)
```

Out[29]:

```
<matplotlib.legend.Legend at 0x7fee7a5d5dd0>
```



The effect is not dramatic. Also, the curve is not as smooth as for the other subject (see below) or Denis' data. Why is that the case?

In conclusion from this part, **negative** shifts of more than a TR yield better results (although maybe neglegable)

The entire thing is repeated with less annotation for another subject:

In [ ]:

sub14

In [22]:

```
subPath = '/media/sebastian/elements/data/s1_anfunco_analysis/sub14/func/vaso/stimulation/run1/'
BoldTimeCourse = subPath + 'BOLD_intemp.nii'
VasoTimeCourse = subPath + 'Nulled_intemp.nii'


Bold = nb.load(BoldTimeCourse)
Vaso = nb.load(VasoTimeCourse)

BoldData = nb.load(BoldTimeCourse).get_fdata()
VasoData = nb.load(VasoTimeCourse).get_fdata()


BoldData = BoldData[:,:,:,:400]
VasoData = VasoData[:,:,:,:400]
```

In [23]:

```
subPath = '/media/sebastian/elements/data/s1_anfunco_analysis/sub14/func/vaso/stimulation/run1/'
BoldTimeCourse = subPath + 'BOLD_intemp.nii' VasoTimeCourse
= subPath + 'Nulled_intemp.nii'
```

In [ ]:

```
%%bash -s "$subPath" "$BoldTimeCourse" "$VasoTimeCourse"

cd /media/sebastian/Data/BOCO_shift/sub14/run1 fslroi
$2 Notnulled_intemp_trunc 60 40 80 30 10 6 fslroi $3
Nulled_intemp_trunc 60 40 80 30 10 6
LN_BOCO -Nulled Nulled_intemp_trunc.nii.gz -BOLD Notnulled_intemp_trunc.nii.gz
output trunc
```

```
subPath = '/media/sebastian/Data/BOCO_shift/sub14/run1/' BoldTimeCourse
= subPath + 'Notnulled_intemp_trunc.nii.gz'
VasoTimeCourse = subPath + 'Nulled_intemp_trunc.nii.gz'


Bold = nb.load(BoldTimeCourse)
Vaso = nb.load(VasoTimeCourse)

BoldData = nb.load(BoldTimeCourse).get_fdata()
VasoData = nb.load(VasoTimeCourse).get_fdata() In
```

[ ]:

```
manualBoco = np.divide(VasoData, BoldData)
```

In [ ]:

```
img = nb.Nifti1Image(BoldData, header=Bold.header, affine=Bold.affine)
img.to_filename(f'/media/sebastian/Data/BOCO_shift/sub14/run1/BOLD_intemp_trunc.
nii')
```

In [ ]:

```
subPath = '/media/sebastian/Data/BOCO_shift/sub14/run1/'
BoldTimeCourse = subPath + 'Notnulled_intemp_trunc.nii.gz'
VasoTimeCourse = subPath + 'Nulled_intemp_trunc.nii.gz'


Bold = nb.load(BoldTimeCourse)
Vaso = nb.load(VasoTimeCourse)

BoldData = nb.load(BoldTimeCourse).get_fdata()
VasoData = nb.load(VasoTimeCourse).get_fdata()


BoldData = BoldData[:,:,:,:len(tr_arr)]
VasoData = VasoData[:,:,:,:len(tr_arr)]
```

In [ ]:

```
img = nb.Nifti1Image(VasoData, header=Vaso.header, affine=Vaso.affine)
img.to_filename(f'/media/sebastian/Data/BOCO_shift/sub14/run1/nulled_intemp_trun
c.nii')

img = nb.Nifti1Image(BoldData, header=Bold.header, affine=Bold.affine)
img.to_filename(f'/media/sebastian/Data/BOCO_shift/sub14/run1/notnulled_intemp_t
runc.nii')
```

In [25]:

In [ ]:

```
shifts = np.arange(-2*tr,2*tr,200)
shifts = [str(int(i)) for i in shifts]

for i in range(len(shifts)):
    if not shifts[i][0]=='-':
        shifts[i] = 'pos' + shifts[i].zfill(4)
    if shifts[i][0]=='-':
        shifts[i] = shifts[i][1:]
        shifts[i] = 'neg' + shifts[i].zfill(4)
```

In [26]:

```
shiftVals = np.arange(-2*tr,2*tr,200)
```

we want to shift the BOLD in order to leave the VASO as untouched as possible

In [ ]:

```
for shiftVal,shift in zip(shiftVals[:1], shifts[:1]):
 print(f'shifting the data {shiftVal} ms')

    newData = np.zeros(BoldData.shape)

    for n in range(BoldData.shape[0]):
 for i in range(BoldData.shape[1]):
 for j in range(BoldData.shape[2]):
 timeCourse = BoldData[n][i][j]

                x = np.arange(0, BoldData.shape[-1]*tr,tr)
                dataValsInterpolated = interp1d(x, timeCourse,
 kind='cubic',boun
ds_error=False, fill_value= 'extrapolate')

                newTimePoints = x+shiftVal
                shifted = dataValsInterpolated(newTimePoints)
 newData[n][i][j] = shifted

    # Doing the BOLD-correction manually. Laynii gave unwanted results when
work ing with data that was loaded as matrix and saved as .nii again
    manualBoco = np.divide(VasoData, newData)
    # Because FSL for some reason does not like values between 0 and 1 in their
 GLM, we multiply the entire thing by 100
 manualBoco = manualBoco*100

    img  =  nb.Nifti1Image(manualBoco,  header=Bold.header,  affine=Bold.affine)
img.to_filename(f'/media/sebastian/Data/BOCO_shift/sub14/run1/{shift}_VASO_L
N_test.nii')
# Set this to the directory where you'll dump all the fsf files
# May want to make it a separate directory, because you can delete them all o
#   once Feat runs
fsfdir="/media/sebastian/Data/BOCO_shift/sub14/fsfs"
```

```
for shift in shifts:    replacements =
{'INFILE':f'/media/sebastian/Data/BOCO_shift/sub14/run1/{shif t}_VASO_LN.nii',
'OUTDIR': f'/media/sebastian/Data/BOCO_shift/sub14/run1/{shift}
_VASO_LN'}            with open(f"{fsfdir}/template.fsf") as
 infile:        with open(f"{fsfdir}/shifted_{shift}.fsf", 'w') as
 outfile:            for line in infile:                for src,
 target in replacements.items():
                    line = line.replace(src, target)
 outfile.write(line)
```

In [ ]: fsfs =

```
sorted(glob.glob(f'{fsfdir}/shifted_*.fsf')) In [
```

]:

```
%%bash -s "{" ".join(fsfs)}"

for fsf in $1
do
wait; feat $fsf &
done
```

In [ ]: %%bash cd

```
/media/sebastian/Data/BOCO_shift/sub14/run1/neg3114_VASO_LN.feat/stats

delta_x=$(3dinfo -di /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
delta_y=$(3dinfo -dj /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
delta_z=$(3dinfo -dk /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
sdelta_x=$(echo "((sqrt($delta_x * $delta_x) / 5))"|bc -l)
sdelta_y=$(echo "((sqrt($delta_y * $delta_y) / 5))"|bc -l)
sdelta_z=$(echo "((sqrt($delta_z * $delta_z) / 1))"|bc -l)

3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix VASO
_scaled_D2.nii -input zstat1.nii.gz
3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix VASO
_scaled_D3.nii -input zstat2.nii.gz
3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix VASO
_scaled_D4.nii -input zstat3.nii.gz
```

In [ ]:

```
```

In [ ]:

```
T1w = 1/(np.divide(np.std(test, axis=3), np.mean(test, axis=3)))
```

In [ ]:

```
img = nb.Nifti1Image(T1w, header=Bold.header, affine=Bold.affine)
img.to_filename(f'/media/sebastian/Data/BOCO_shift/sub14/run1/T1w.nii')
```

In [ ]:

```
%%bash cd

/media/sebastian/Data/BOCO_shift/sub14/run1

delta_x=$(3dinfo -di T1w.nii) delta_y=$(3dinfo
-dj T1w.nii) delta_z=$(3dinfo -dk T1w.nii)
sdelta_x=$(echo "((sqrt($delta_x * $delta_x) / 5))"|bc -l)
sdelta_y=$(echo "((sqrt($delta_y * $delta_y) / 5))"|bc -l)
sdelta_z=$(echo "((sqrt($delta_z * $delta_z) / 1))"|bc -l)

3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix scal
ed_T1w.nii -input T1w.nii
```

In [ ]:

```
%%bash -s "{" ".join(shifts)}"

for shift in $1 do echo $shift cd
"/media/sebastian/Data/BOCO_shift/sub14/run1/"$shift"_VASO_LN.feat/stats"

delta_x=$(3dinfo -di /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
delta_y=$(3dinfo -dj /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
delta_z=$(3dinfo -dk /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
sdelta_x=$(echo "((sqrt($delta_x * $delta_x) / 5))"|bc -l)
sdelta_y=$(echo "((sqrt($delta_y * $delta_y) / 5))"|bc -l)
sdelta_z=$(echo "((sqrt($delta_z * $delta_z) / 1))"|bc -l)

3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix VASO
_scaled_D2.nii -input zstat1.nii.gz
3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix VASO
_scaled_D3.nii -input zstat2.nii.gz
3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix VASO
_scaled_D4.nii -input zstat3.nii.gz done
```

```
%%bash cd

/media/sebastian/Data/BOCO_shift/sub14/run1/BOLD.feat/stats
```

```
delta_x=$(3dinfo -di /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
```

```
delta_y=$(3dinfo -dj /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
delta_z=$(3dinfo -dk /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
sdelta_x=$(echo "((sqrt($delta_x * $delta_x) / 5))"|bc -l)
sdelta_y=$(echo "((sqrt($delta_y * $delta_y) / 5))"|bc -l)
sdelta_z=$(echo "((sqrt($delta_z * $delta_z) / 1))"|bc -l)

3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix VASO
_scaled_D2.nii -input zstat1.nii.gz
3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix VASO
_scaled_D3.nii -input zstat2.nii.gz
3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix VASO
_scaled_D4.nii -input zstat3.nii.gz
```

In [ ]:

```
%%bash

fslmaths /media/sebastian/Data/BOCO_shift/sub14/run1/BOLD_scaled_D2_mask.nii.gz
 -bin
```

/media/sebastian/Data/BOCO_shift/sub14/run1/BOLD_scaled_D2_mask_bin.nii.gz In

[27]:

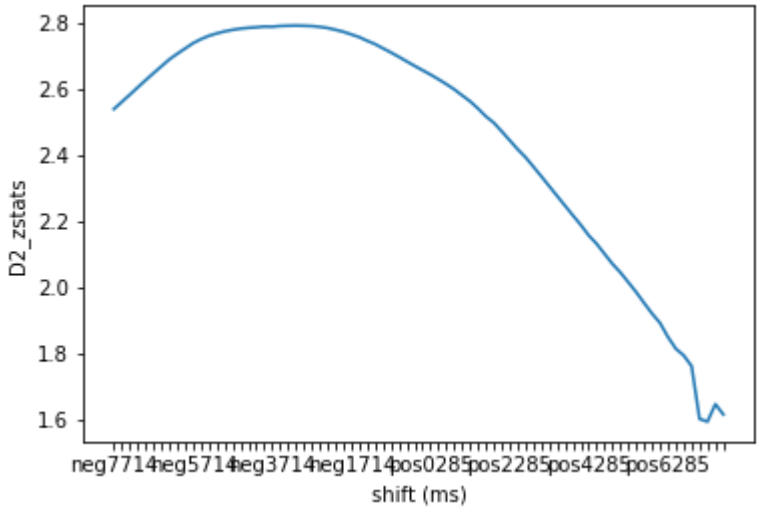
```
extracted = []

for shift in shifts:
    # Define files
    # Timecourse file
    FILE1 = f'/media/sebastian/Data/BOCO_shift/sub14/run1/{shift}_VASO_LN.feat/s
tats/VASO_scaled_D2.nii'
    # Load timecourse data
    nii1 = nb.load(FILE1)
    data = nii1.get_fdata()


    # For entire digit-ROIs
    FILE3 = f'/media/sebastian/Data/BOCO_shift/sub14/run1/BOLD_scaled_D2_mask_bi
n.nii.gz'
    nii3 = nb.load(FILE3)
    digit_mask = nii3.get_fdata()

    extracted.append(np.mean(data[digit_mask.astype(bool)]))
```

In [43]:

In [ ]:

```
dataframe = pd.DataFrame({'shift (ms)': shifts, 'D2_zstats': extracted})

label_indices = np.linspace(0,len(dataframe['D2_zstats']),10)
label_indices = [int(i) for i in label_indices]

import seaborn as sns
# sns.set_style("whitegrid")
plot = sns.lineplot(data=dataframe, x="shift (ms)", y="D2_zstats")

# for ind, label in enumerate(plot.get_xticklabels()):
#     if ind in label_indices:  # every 10th label is kept
#         label.set_visible(True)
#     else:
#         label.set_visible(False)

for label in plot.get_xticklabels():
    label.set_visible(False)

for label in plot.get_xticklabels()[::10]:
    label.set_visible(True)


for label in plot.get_xticks():
    label.set_visible(False)

for label in plot.get_xticks()[::10]:
    label.set_visible(True)


plt.xticks(rotation=45)
plt.ylabel('zstats')
plt.axvline(77/2, 0,2.8, color = 'g', label= 'no shift')
plt.axvline(np.argmax(dataframe['D2_zstats'], axis=0), 0,2.8, color = 'r', label
= f"max z at {shifts[np.argmax(dataframe['D2_zstats'])]} ms")
plt.legend()
# plt.savefig('shiftVsZstat.jpg')
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent cal
l last)
<ipython-input-43-7c8079f3bd62> in <module>
     22
     23 for label in plot.get_xticks():
---> 24     label.set_visible(False)
     25
     26 for label in plot.get_xticks()[::10]:

AttributeError: 'int' object has no attribute 'set_visible'
```



event related averages

Here, we want to plot event related averages for 5 data-types:

- BOLD
- BOLD shifted to 'optimum'
- VASO with standard BOLD-correction
- VASO with 'optimal' BOLD-correction
- CBV timecourse without BOLD-correction

```
In [347]: %%bash cd

/media/sebastian/Data/BOCO_shift/sub14/run1

delta_x=$(3dinfo -di /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
delta_y=$(3dinfo -dj /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
delta_z=$(3dinfo -dk /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
sdelta_x=$(echo "((sqrt($delta_x * $delta_x) / 5))"|bc -l)
sdelta_y=$(echo "((sqrt($delta_y * $delta_y) / 5))"|bc -l)
sdelta_z=$(echo "((sqrt($delta_z * $delta_z) / 1))"|bc -l)

3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix /med
ia/sebastian/Data/BOCO_shift/sub14/run1/BOLD_intemp_trunc_upsample.nii -input /m
edia/sebastian/Data/BOCO_shift/sub14/run1/BOLD_intemp_trunc.nii
```

```
** AFNI converts NIFTI_datatype=4 (INT16) in file /media/sebastian/D
ata/BOCO_shift/sub14/run1/BOLD_intemp_trunc.nii to FLOAT32
     Warnings of this type will be muted for this session.
     Set AFNI_NIFTI_TYPE_WARN to YES to see them all, NO to see non
e.
*+ WARNING:   If you are performing spatial transformations on an ob
lique dset,
  such as /media/sebastian/Data/BOCO_shift/sub14/run1/BOLD_intemp_tr
unc.nii,  or viewing/combining it with volumes of differing
 obliquity,  you should consider running:      3dWarp -
 deoblique   on this and  other oblique datasets in the same
 session. See 3dWarp -help for details.
++ Oblique dataset:/media/sebastian/Data/BOCO_shift/sub14/run1/BOLD_
intemp_trunc.nii is 38.554882 degrees from plumb.
```

```
In [391]: %%bash cd

/media/sebastian/Data/BOCO_shift/sub14/run1

delta_x=$(3dinfo -di /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
delta_y=$(3dinfo -dj /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
delta_z=$(3dinfo -dk /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
sdelta_x=$(echo "((sqrt($delta_x * $delta_x) / 5))"|bc -l)
sdelta_y=$(echo "((sqrt($delta_y * $delta_y) / 5))"|bc -l)
sdelta_z=$(echo "((sqrt($delta_z * $delta_z) / 1))"|bc -l)

3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix /med
ia/sebastian/Data/BOCO_shift/sub14/run1/notnulled_intemp_trunc_upsample.nii -inp
ut /media/sebastian/Data/BOCO_shift/sub14/run1/Notnulled_intemp_trunc.nii.gz
```

```
*+ WARNING:   If you are performing spatial transformations on an ob
lique dset,
  such as /media/sebastian/Data/BOCO_shift/sub14/run1/Notnulled_inte
mp_trunc.nii.gz,  or viewing/combining it with volumes of
 differing obliquity,  you should consider running:
 3dWarp -deoblique   on this and  other oblique datasets in
 the same session. See 3dWarp -help for details.
++ Oblique dataset:/media/sebastian/Data/BOCO_shift/sub14/run1/Notnu
lled_intemp_trunc.nii.gz is 38.554882 degrees from plumb.
```

```
In [434]: %%bash cd

/media/sebastian/Data/BOCO_shift/sub14/run1

delta_x=$(3dinfo -di /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
delta_y=$(3dinfo -dj /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
delta_z=$(3dinfo -dk /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
sdelta_x=$(echo "((sqrt($delta_x * $delta_x) / 5))"|bc -l)
sdelta_y=$(echo "((sqrt($delta_y * $delta_y) / 5))"|bc -l)
sdelta_z=$(echo "((sqrt($delta_z * $delta_z) / 1))"|bc -l)

3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix /med
ia/sebastian/Data/BOCO_shift/sub14/run1/neg3114_VASO_LN_upsample.nii -input /med
ia/sebastian/Data/BOCO_shift/sub14/run1/neg3114_VASO_LN.nii
```

```
** AFNI converts NIFTI_datatype=4 (INT16) in file /media/sebastian/D
ata/BOCO_shift/sub14/run1/neg3114_VASO_LN.nii to FLOAT32
     Warnings of this type will be muted for this session.
     Set AFNI_NIFTI_TYPE_WARN to YES to see them all, NO to see non
e.
*+ WARNING:   If you are performing spatial transformations on an ob
lique dset,
  such as /media/sebastian/Data/BOCO_shift/sub14/run1/neg3114_VASO_L
N.nii,  or viewing/combining it with volumes of differing
 obliquity,  you should consider running:      3dWarp -
 deoblique   on this and  other oblique datasets in the same
 session. See 3dWarp -help for details.
++ Oblique dataset:/media/sebastian/Data/BOCO_shift/sub14/run1/neg31
14_VASO_LN.nii is 38.554882 degrees from plumb.
```

```
In [ ]: %%bash cd

/media/sebastian/Data/BOCO_shift/sub14/run1

delta_x=$(3dinfo -di /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
delta_y=$(3dinfo -dj /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
delta_z=$(3dinfo -dk /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
sdelta_x=$(echo "((sqrt($delta_x * $delta_x) / 5))"|bc -l)
sdelta_y=$(echo "((sqrt($delta_y * $delta_y) / 5))"|bc -l)
sdelta_z=$(echo "((sqrt($delta_z * $delta_z) / 1))"|bc -l)

3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix /med
ia/sebastian/Data/BOCO_shift/sub14/run1/nulled_intemp_trunc_upsample.nii -input
 /media/sebastian/Data/BOCO_shift/sub14/run1/Nulled_intemp_trunc.nii.gz
```

```
In [435]: %%bash cd

/media/sebastian/Data/BOCO_shift/sub14/run1

delta_x=$(3dinfo -di /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
delta_y=$(3dinfo -dj /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
delta_z=$(3dinfo -dk /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
sdelta_x=$(echo "((sqrt($delta_x * $delta_x) / 5))"|bc -l)
sdelta_y=$(echo "((sqrt($delta_y * $delta_y) / 5))"|bc -l)
sdelta_z=$(echo "((sqrt($delta_z * $delta_z) / 1))"|bc -l)

3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix /med
ia/sebastian/Data/BOCO_shift/sub14/run1/trunc_VASO_LN_upsample.nii -input /medi
a/sebastian/Data/BOCO_shift/sub14/run1/trunc_VASO_LN.nii
```

```
*+ WARNING:   If you are performing spatial transformations on an ob
lique dset,
  such as /media/sebastian/Data/BOCO_shift/sub14/run1/trunc_VASO_LN.
nii,  or viewing/combining it with volumes of differing obliquity,
you should consider running:      3dWarp -deoblique   on this and
other oblique datasets in the same session. See 3dWarp -help for
details.
```

```
++ Oblique dataset:/media/sebastian/Data/BOCO_shift/sub14/run1/trunc
_VASO_LN.nii is 38.554882 degrees from plumb.
```

```
In [444]: %%bash cd

/media/sebastian/Data/BOCO_shift/sub14/run1

delta_x=$(3dinfo -di /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
delta_y=$(3dinfo -dj /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
delta_z=$(3dinfo -dk /media/sebastian/elements/data/s1_anfunco_analysis/sub14/fu
nc/vaso/stimulation/run1/T1w.nii)
sdelta_x=$(echo "((sqrt($delta_x * $delta_x) / 5))"|bc -l)
sdelta_y=$(echo "((sqrt($delta_y * $delta_y) / 5))"|bc -l)
sdelta_z=$(echo "((sqrt($delta_z * $delta_z) / 1))"|bc -l)

3dresample -dxyz $sdelta_x $sdelta_y $sdelta_z -rmode Cu -overwrite -prefix /med
ia/sebastian/Data/BOCO_shift/sub14/run1/BOLD_neg3114_upsample.nii -input /media/
sebastian/Data/BOCO_shift/sub14/run1/BOLD_neg3114.nii
```

```
** AFNI converts NIFTI_datatype=4 (INT16) in file /media/sebastian/D
ata/BOCO_shift/sub14/run1/BOLD_neg3114.nii to FLOAT32
     Warnings of this type will be muted for this session.
     Set AFNI_NIFTI_TYPE_WARN to YES to see them all, NO to see non
e.
*+ WARNING:   If you are performing spatial transformations on an ob
lique dset,
  such as /media/sebastian/Data/BOCO_shift/sub14/run1/BOLD_neg3114.n
ii,  or viewing/combining it with volumes of differing
 obliquity,  you should consider running:      3dWarp -
 deoblique   on this and  other oblique datasets in the same
 session. See 3dWarp -help for details.
++ Oblique dataset:/media/sebastian/Data/BOCO_shift/sub14/run1/BOLD_
neg3114.nii is 38.554882 degrees from plumb.
```

```
In [16]:
```

```
import numpy as np
import pandas as pd
import nibabel as nb


def event_related_average(runs,
 event_protocols,                          mask,

 event_of_interest,
 start_before=1,
 end_after=5                          ):

    '''Compute event related averages for a given event of interest within a
 specified mask.



    TODO:
 1. write proper description - somewhat done
 2. enable specification of window before and after onset - done
 3. enable selsection of single/ all event types'''
```

```
    if isinstance(runs, str):        runs
 = [runs]    if
 isinstance(event_protocols, str):
 event_protocols = [event_protocols]

    if len(runs) != len(event_protocols):        return 'Number
 of runs does not match number of protocols'
 trialwise_activity = []

    for run, event_file in zip(runs, event_protocols):
        # Load run.
        Nii = nb.load(run)
        # As before, get the data as an array.
 data = Nii.get_fdata()
        # load the nifty-header to get some meta-data.
 header = Nii.header

        # As the number of volumes which is the 4th position of
        # get_shape. This seems to be unused so I will comment it out to check.
        # nr_volumes = int(header.get_data_shape()[3])

        # Or the TR, which is the 4th position of get_zooms().
#         tr = int(header.get_zooms()[3])
 tr = 3.857429359296482/2
        # Next, we need the events in order to know the time-points
        # during which each finger was stimulated. Remeber that the
        # file was defined in the for-loop.
#         events = pd.read_csv(event_file, delimiter='\t')
 events = event_file

        # We can then iterate over each trial and look for events pertaining to
        # the specific type we are interested in.
 for i, row in events.iterrows():
            # See if event is stimulating the digit in question.
            if row['trial_type'] == f'{event_of_interest}':
                # Read the onset time. The time is in seconds,
                # therefore, we need to divide it by out TR.
 onset = round(row['onset']/tr)                # Do the same
 for the offset
                offset = round(onset + row['duration']/tr)
                # Because we want the % signal-change, we need the mean
 # of the voxel we are looking at. This is done with
                # some fancy matrix operations.
 mask_mean = np.mean(data[:, :, :][mask.astype(bool)])

                # The next command is a bit annoying to break down but
                # I will give it a try.
                # First thing to note is that in the end, we want an
                # array that contains a row for each trial in which our
                # finger in question was stimulated with a length of
 # the event + a given window. The window we chose was:
 # 1 volume preceeding and 5 volumes exceeding the
                # stimulation time.
                #
                # Because I did not want to define the number of
                # stimulations upfront, I created a list
                # ('TrialWiseEstimates'), which we can append to. This
```

```
                # will later be transformed into an array.
                #
                # So, for each trial we append an array to the list.
                # This array contains values from the variable 'Data',
                # which contains our functional data as an array. The
 # first 3 dimensions of the array are spacial
                # coordinates. We take all of them (indicated by ':')
                # because they will be masked anyway.
                # For the 4th dimension (time), we only
                # want to extract the values for the given trial.
 # Therefore, we take the indices from the onset
                # until the offset +- a defined window.
                #
                # This array is then divided by the voxel mean. Because
 # we want % signal-change, we then subtract 1 and
                # multiply by 100.

                trialwise_activity.append((((data[:, :, :, onset
 - start_before:offset
                                            + end_after]
                                            [mask.astype(bool)]) / mask_mean)
                                            - 1) * 100)




#                 trialwise_activity.append((data[:, :, :, onset-
start_before:of fset+end_after][mask.astype(bool)]))
#                 # Voila

#     # We then create the aray we spoke of by just concatinating.
    trialwise_activity = np.concatenate(trialwise_activity, axis=0)
#     # Finally, we can average the array element-wise in
#     # order to get our EVA.    return
 np.mean(trialwise_activity, axis=0)
```

In [22]:

```
runs = ['/media/sebastian/Data/BOCO_shift/sub14/run1/BOLD_intemp_trunc_upsample.
nii']
mask = digit_mask
event_of_interest = 'D2'
```

In [23]:

```
eventfile = '/media/sebastian/elements/data/s1_anfunco_analysis/sub14/func/vaso/
stimulation/events/events_run1_D2_s.txt'

events = pd.read_csv(eventfile, names=['onset','duration','trial_type'], delim_w
hitespace=True)
events['trial_type'] = ['D2','D2','D2','D2'] #
events['onset'] = events['onset']/(tr/1000) #
events['duration'] = events['duration']/(tr/1000)
events
```

Out[23]:

	onset	duration	trial_type

0	96	31	D2
1	280	31	D2
2	464	31	D2
3	587	31	D2

In [24]:

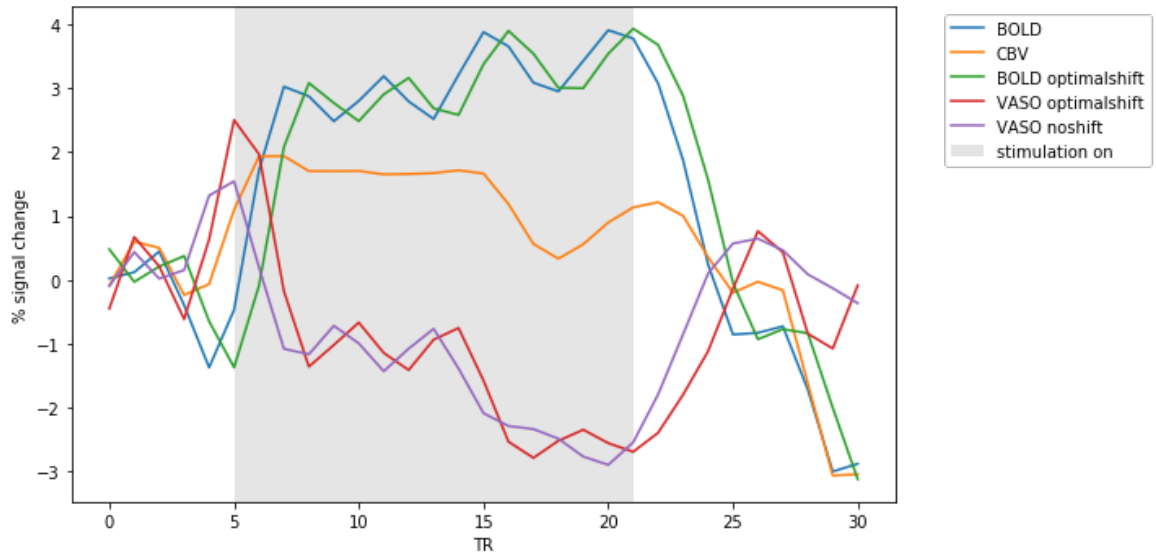
```
runs = ['/media/sebastian/Data/BOCO_shift/sub14/run1/BOLD_intemp_trunc_upsample.
nii'] mask = digit_mask event_of_interest = 'D2'

BOLDunshifted = event_related_average(runs,
 [events],                          mask,

 event_of_interest,
 start_before=5,
 end_after=10                              )

runs = ['/media/sebastian/Data/BOCO_shift/sub14/run1/nulled_intemp_trunc_upsampl
e.nii']

nulledunshifted = event_related_average(runs,
                           [events],
 mask,

 event_of_interest,
 start_before=5,
 end_after=10                              )

runs = ['/media/sebastian/Data/BOCO_shift/sub14/run1/neg3114_VASO_LN_upsample.ni
i']

VASO_LN_optimal = event_related_average(runs,
                           [events],
 mask,

 event_of_interest,
 start_before=5,
 end_after=10                              )


runs = ['/media/sebastian/Data/BOCO_shift/sub14/run1/trunc_VASO_LN_upsample.nii'
]

VASO_LN_noshift = event_related_average(runs,
                           [events],
 mask,
                           event_of_interest,
start_before=5,                                end_after=10                                )
runs = ['/media/sebastian/Data/BOCO_shift/sub14/run1/BOLD_neg3114_upsample.nii']

BOLD_optimal = event_related_average(runs,
 [events],                          mask,
                           event_of_interest,
 start_before=5,
 end_after=10
                           )
```

In [25]:

```
fig = plt.figure(tight_layout = True, figsize=(10,5))
plt.plot(BOLDunshifted, label = 'BOLD')
plt.plot(nulledunshifted, label='CBV') plt.plot(BOLD_optimal,
label = 'BOLD optimalshift') plt.plot(VASO_LN_optimal,
```

```
label='VASO optimalshift') plt.plot(VASO_LN_noshift,
label='VASO noshift')

plt.axvspan(5, 21, color='grey', alpha=0.2, lw=0, label = 'stimulation on')
plt.ylabel('% signal change')
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left') plt.xlabel('TR')
plt.savefig('eventrelatedaverages_shifts.jpg')
plt.show()
```

```
/home/sebastian/anaconda3/lib/python3.7/site-packages/matplotlib/fig
ure.py:2299: UserWarning: This figure includes Axes that are not com
patible with tight_layout, so results might be incorrect.
warnings.warn("This figure includes Axes that are not compatible "
```



In [443]:

```
for shiftVal,shift in zip(shiftVals, shifts):
 if shift == 'neg3114':        print('yes')


        newData = np.zeros(BoldData.shape)

        for n in range(BoldData.shape[0]):
 for i in range(BoldData.shape[1]):
 for j in range(BoldData.shape[2]):
 timeCourse = BoldData[n][i][j]

                    x = np.arange(0, BoldData.shape[-1]*tr,tr)
                    dataValsInterpolated = interp1d(x, timeCourse,
 kind='cubic',
bounds_error=False, fill_value= 'extrapolate')

                    newTimePoints = x+shiftVal
                    shifted = dataValsInterpolated(newTimePoints)
 newData[n][i][j] = shifted



        img = nb.Nifti1Image(newData, header=Bold.header, affine=Bold.affine)
 img.to_filename(f'/media/sebastian/Data/BOCO_shift/sub14/run1/BOLD_{shif
```

```
t}.nii') yes
```