FCS

June 15, 2021

1 SPAD-FFS yellow-green beads Bright-Eyes TTM

[1]:
```
import numpy as np
import pandas as pd
import math
import h5py
import matplotlib.pyplot as plt
from scipy.signal import savgol_filter
import copy
from ipyfilechooser import FileChooser
import os

from spad_tools import *
from spad_fcs import *
```

[3]:
```
fc = FileChooser()
fc.default_path='/home/labuser/dataSlowDisk'
fc.use_dir_icons = True
display(fc)
```

```
FileChooser(path='/home/labuser/dataSlowDisk', filename='',␣
↪title='HTML(value='', layout=Layout(display='none'…
```

[4]:
```
filename = fc.selected
data_head, data_filename = os.path.split(filename)
```

[5]:
```
sysclk_MHz=240.  # FPGA system clock (MHz)
laser_MHz=40.    # Laser repetition rate (MHz)
laser_factor=1   # Adimensional number to account for laser actual repetiton␣
↪frequency
nchannel = 21    # Number of active channels
kC4=43           # Time width of TCSPC histogram bin (picoseconds
```

[6]:
```
laser_MHz=laser_MHz*laser_factor
laser_Hz=laser_MHz*10**6
max_counter=2**16-1
sysclk_ps=1000000./sysclk_MHz #ps
```

```
print("SysClk ps:", sysclk_ps)
laser_ps=1000000./laser_MHz #ps
print("LaserClk ps:", laser_ps)
ratio=sysclk_MHz/laser_MHz
sysclk_ps=1e6/sysclk_MHz
laser_ps=1e6/laser_MHz
nbins=int(round(laser_ps/kC4))
```

```
SysClk ps: 4166.666666666667
LaserClk ps: 25000.0
```

[7]:
```
myReturn=ttp.convertDataRAW( filenameToRead=filename,
                    sysclk_MHz = sysclk_MHz,
                    laser_MHz=laser_MHz,
                    dwell_time_us=100.,
                    list_of_channels=np.arange(0,nchannel),
                    autoCalibration=True,
                    kC4=45.,
                    textInPlot=None,
                    compressionLevel=1,
                    makePlots=True,
                    ignorePixelLineFrame = False)
```

```
/mnt/Disk1T/dataSlowDisk/data-2021-04-13 FCS nanoBeads/FCS_scanfcs_Dataset_40MHz
**************************
* Size table:  44324384
**************************
Convert to DataFrame
Converted

Calculate rates
Calculate cumulative step
Add cumulativeStep
Acquisition lasted:  327.6038857 s
Scan_enable 43888760.0 ratio 0.9901719107929396 rate 133968.98484955914
line_enable 1310535.0 ratio 0.029566908363577032 rate 4000.364639142618
pixel_enable 32000373.0 ratio 0.7219586627532151 rate 97680.07766948169
Laser 11543473.0 ratio 0.26043166217493285 rate 35236.06863006143
Calculate totalphotons
kC4<=== 45.0
sysclk_ps<=== 4166.666666666667
kC4<=== 44.80286738351255
Start process
Current frame: 1        :   3%|          | 1329729/44324384 [00:00<00:05,
7641187.76it/s]
Start analysisForImg
Arrays copied into analysisForImg
```

```
Current frame: 23       : : 44767543it [00:08, 5038197.80it/s]

('Total Frame:', 23)

  0%|          | 0/21 [00:00<?, ?it/s]

New HDF5 written
total_photon        uint8
cumulative_step     int64
arr_px             uint16
arr_px_corr        uint16
arr_py             uint16
arr_frame          uint16
dtype: object
Start conversion of 0 channel
t_0 valid_tdc_0
..


  0%|          | 0/12143360 [00:00<?, ?it/s]
 36%|         | 4371588/12143360 [00:00<00:00, 42971705.80it/s]
 70%|       | 8500310/12143360 [00:00<00:00, 42450584.90it/s]

Data ready, conversion to array
Adding keys to HDF5…  "ch_0"

  5%|          | 1/21 [00:03<01:19,  3.96s/it]

t_0      int16
t_L      int16
dS_0    uint16
dtype: object
Start conversion of 1 channel
t_1 valid_tdc_1
..



  0%|          | 0/11691547 [00:00<?, ?it/s]

 10%|          | 2/21 [00:07<01:10,  3.73s/it], 66552914.42it/s]

Data ready, conversion to array
Adding keys to HDF5…  "ch_1"
t_1      int16
t_L      int16
dS_1    uint16
dtype: object
Start conversion of 2 channel
t_2 valid_tdc_2
..
```

```
  0%|          | 0/11687101 [00:00<?, ?it/s]


12264733it [00:07, 1681320.79it/s]
11808415it [00:03, 3393295.45it/s]
11803971it [00:00, 41974798.54it/s]

Data ready, conversion to array
Adding keys to HDF5…  "ch_2"

 14%|         | 3/21 [00:10<01:05,  3.62s/it]

t_2      int16
t_L      int16
dS_2    uint16
dtype: object
Start conversion of 3 channel
t_3 valid_tdc_3
..


  0%|          | 0/11693968 [00:00<?, ?it/s]
 57%|       | 6665523/11693968 [00:00<00:00, 66399151.12it/s]

Data ready, conversion to array
Adding keys to HDF5…  "ch_3"

 19%|         | 4/21 [00:13<01:00,  3.53s/it]

t_3      int16
t_L      int16
dS_3    uint16
dtype: object
Start conversion of 4 channel
t_4 valid_tdc_4
..


  0%|          | 0/11980522 [00:00<?, ?it/s]

 41%|        | 4912005/11980522 [00:00<00:00, 48476904.95it/s]

 81%|      | 9704205/11980522 [00:00<00:00, 48002514.80it/s]

Data ready, conversion to array
Adding keys to HDF5…  "ch_4"

 24%|         | 5/21 [00:17<00:57,  3.59s/it]
```

```
t_4      int16
t_L      int16
dS_4    uint16
dtype: object
Start conversion of 5 channel
t_5 valid_tdc_5
..


  0%|          | 0/11731730 [00:00<?, ?it/s]


 29%|         | 6/21 [00:20<00:52,  3.50s/it], 63744006.05it/s]
Data ready, conversion to array
Adding keys to HDF5…  "ch_5"
t_5      int16
t_L      int16
dS_5    uint16
dtype: object
Start conversion of 6 channel
t_6 valid_tdc_6
..


  0%|          | 0/11963205 [00:00<?, ?it/s]


 40%|         | 4785280/11963205 [00:00<00:00, 47458293.34it/s]


11810839it [00:10, 1095946.52it/s]
12100305it [00:07, 1632114.85it/s]
11849017it [00:03, 3203696.46it/s]
12082832it [00:00, 29303746.06it/s]
Data ready, conversion to array
Adding keys to HDF5…  "ch_6"


 33%|         | 7/21 [00:24<00:49,  3.56s/it]
t_6      int16
t_L      int16
```

```
dS_6    uint16
dtype: object
Start conversion of 7 channel
t_7 valid_tdc_7
..


  0%|          | 0/12118376 [00:00<?, ?it/s]
 36%|         | 4362588/12118376 [00:00<00:00, 43302671.80it/s]
 70%|       | 8482810/12118376 [00:00<00:00, 42644307.28it/s]

Data ready, conversion to array
Adding keys to HDF5…  "ch_7"

 38%|        | 8/21 [00:28<00:47,  3.67s/it]

t_7      int16
t_L      int16
dS_7    uint16
dtype: object
Start conversion of 8 channel
t_8 valid_tdc_8
..



  0%|          | 0/11816043 [00:00<?, ?it/s]

 48%|        | 5671680/11816043 [00:00<00:00, 56165682.38it/s]

 94%|     | 11107040/11816043 [00:00<00:00, 55436856.66it/s]

Data ready, conversion to array
Adding keys to HDF5…  "ch_8"

 43%|        | 9/21 [00:31<00:42,  3.57s/it]

t_8      int16
t_L      int16
dS_8    uint16
dtype: object
Start conversion of 9 channel
t_9 valid_tdc_9
..



  0%|          | 0/11981988 [00:00<?, ?it/s]


 36%|        | 4313484/11981988 [00:00<00:00, 42857563.28it/s]
```

```
 73%|      | 8746787/11981988 [00:00<00:00, 43200639.80it/s]

Data ready, conversion to array
Adding keys to HDF5…  "ch_9"

 48%|        | 10/21 [00:35<00:39,  3.63s/it]

t_9      int16
t_L      int16
dS_9    uint16
dtype: object
Start conversion of 10 channel
t_10 valid_tdc_10
..




  0%|          | 0/11753843 [00:00<?, ?it/s]



12239483it [00:11, 1085821.22it/s]
11934160it [00:07, 1591889.82it/s]
12101719it [00:04, 2982719.47it/s]
11871338it [00:00, 46871264.48it/s]

Data ready, conversion to array
Adding keys to HDF5…  "ch_10"

 52%|       | 11/21 [00:39<00:35,  3.57s/it]

t_10      int16
t_L       int16
dS_10    uint16
dtype: object
Start conversion of 11 channel
t_11 valid_tdc_11
..


  0%|          | 0/11736691 [00:00<?, ?it/s]
 52%|       | 6103032/11736691 [00:00<00:00, 60541634.02it/s]

Data ready, conversion to array
Adding keys to HDF5…  "ch_11"

 57%|       | 12/21 [00:42<00:32,  3.57s/it]
```

```
t_11      int16
t_L       int16
dS_11    uint16
dtype: object
Start conversion of 12 channel
t_12 valid_tdc_12
..


  0%|          | 0/11824200 [00:00<?, ?it/s]

 45%|        | 5320890/11824200 [00:00<00:00, 52327162.64it/s]

 88%|      | 10405296/11824200 [00:00<00:00, 51791567.90it/s]

Data ready, conversion to array
Adding keys to HDF5…  "ch_12"

 62%|       | 13/21 [00:46<00:28,  3.53s/it]

t_12      int16
t_L       int16
dS_12    uint16
dtype: object
Start conversion of 13 channel
t_13 valid_tdc_13
..


  0%|          | 0/11751946 [00:00<?, ?it/s]


 51%|        | 5993469/11751946 [00:00<00:00, 58946473.54it/s]


100%|     | 11751900/11751946 [00:00<00:00, 58238339.97it/s]

Data ready, conversion to array
Adding keys to HDF5…  "ch_13"

 67%|       | 14/21 [00:49<00:24,  3.54s/it]

t_13      int16
t_L       int16
dS_13    uint16
dtype: object
Start conversion of 14 channel
t_14 valid_tdc_14
..
```

```
  0%|          | 0/11753035 [00:00<?, ?it/s]


 51%|        | 5994030/11753035 [00:00<00:00, 59288149.36it/s]


11853966it [00:10, 1096473.62it/s]                                ]
100%|     | 11824200/11824200 [00:07<00:00, 1610150.04it/s]
11869419it [00:03, 3191820.94it/s]
11870530it [00:00, 43994570.52it/s]
 71%|      | 15/21 [00:53<00:21,  3.50s/it]
Data ready, conversion to array
Adding keys to HDF5…  "ch_14"
t_14      int16
t_L       int16
dS_14    uint16
dtype: object
Start conversion of 15 channel
t_15 valid_tdc_15
..


  0%|          | 0/11825992 [00:00<?, ?it/s]
 47%|        | 5558173/11825992 [00:00<00:00, 54693244.35it/s]
 83%|      | 9815497/11825992 [00:00<00:00, 49803589.57it/s]
Data ready, conversion to array
Adding keys to HDF5…  "ch_15"
 76%|      | 16/21 [00:56<00:17,  3.54s/it]
t_15      int16
t_L       int16
dS_15    uint16
dtype: object
Start conversion of 16 channel
t_16 valid_tdc_16
..


  0%|          | 0/11714297 [00:00<?, ?it/s]

 55%|        | 6442810/11714297 [00:00<00:00, 63264177.80it/s]
```

```
 81%|      | 17/21 [01:00<00:14,  3.51s/it], 59198164.57it/s]
Data ready, conversion to array
Adding keys to HDF5…  "ch_16"
t_16      int16
t_L       int16
dS_16    uint16
dtype: object
Start conversion of 17 channel
t_17 valid_tdc_17
..


  0%|          | 0/11874046 [00:00<?, ?it/s]


 45%|        | 5343300/11874046 [00:00<00:00, 53168873.54it/s]


 88%|      | 10449120/11874046 [00:00<00:00, 52504184.73it/s]
Data ready, conversion to array
Adding keys to HDF5…  "ch_17"
 86%|      | 18/21 [01:03<00:10,  3.49s/it]
t_17      int16
t_L       int16
dS_17    uint16
dtype: object
Start conversion of 18 channel
t_18 valid_tdc_18
..

11944159it [00:10, 49803589.57it/s]


  0%|          | 0/11764466 [00:00<?, ?it/s]


 50%|        | 5882200/11764466 [00:00<00:00, 58182290.88it/s]


11944159it [00:10, 1115860.39it/s] [00:00<00:00, 58260068.61it/s]
```

```
11831342it [00:07, 1673623.94it/s]
11992740it [00:03, 3187000.12it/s]
11882044it [00:00, 40023759.30it/s]

Data ready, conversion to array
Adding keys to HDF5…  "ch_18"

 90%|      | 19/21 [01:07<00:07,  3.52s/it]

t_18      int16
t_L       int16
dS_18    uint16
dtype: object
Start conversion of 19 channel
t_19 valid_tdc_19
..


  0%|          | 0/11729054 [00:00<?, ?it/s]
 95%|      | 20/21 [01:10<00:03,  3.47s/it] 61315288.89it/s]

Data ready, conversion to array
Adding keys to HDF5…  "ch_19"
t_19      int16
t_L       int16
dS_19    uint16
dtype: object
Start conversion of 20 channel
t_20 valid_tdc_20
..



  0%|          | 0/11579272 [00:00<?, ?it/s]

100%|     | 21/21 [01:13<00:00,  3.50s/it] 77187443.20it/s]

Data ready, conversion to array
Adding keys to HDF5…  "ch_20"
t_20      int16
t_L       int16
dS_20    uint16
dtype: object
Data saved:  /mnt/Disk1T/dataSlowDisk/data-2021-04-13 FCS
nanoBeads/output/FCS_scanfcs_Dataset_40MHz-raw.h5
```

1.1 Import data YG beads

.h5 file name

[8]:
```
fname = myReturn['filenameH5']
```

Load data, channels = number of channels to load, typically 21

[9]:
```
data=loadATimesData(fname, channels=nchannel)
```

```
11846290it [00:18, 630750.87it/s]
11694992it [00:25, 458891.15it/s]
100%|      | 21/21 [02:00<00:00,  5.74s/it]
```

```
Loading channel 0
Loading channel 1
Loading channel 2
Loading channel 3
Loading channel 4
Loading channel 5
Loading channel 6
Loading channel 7
Loading channel 8
Loading channel 9
Loading channel 10
Loading channel 11
Loading channel 12
Loading channel 13
Loading channel 14
Loading channel 15
Loading channel 16
Loading channel 17
Loading channel 18
Loading channel 19
Loading channel 20
```

As an example, show the data array for channel 0. The first column shows for each photon the macrotimes, the second the microtimes. Expressed in ps.

[10]:
```
data.det0
```

[10]:
```
array([[1.58154140e+12, 6.70359354e+03],
       [1.58158378e+12, 6.92520161e+03],
       [1.58168863e+12, 2.71489808e+03],
       …,
       [3.21740444e+14, 4.65510347e+03],
       [3.21740567e+14, 1.63848092e+04],
       [3.21741377e+14, 1.52087098e+04]])
```

Macrotimes and microtimes are expressed in ps.

[11]:
```
data.macrotime = 1e-12
data.microtime = 1e-12
```

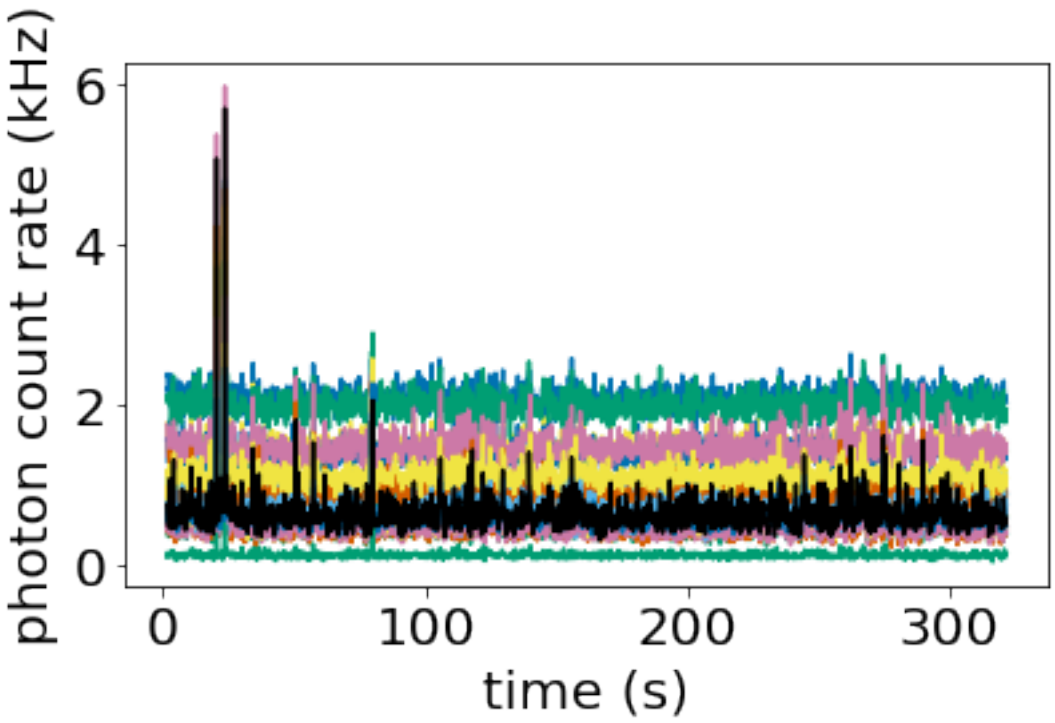
Plot parameters

[12]:
```
fontSize = 20
plt.rcParams.update({'font.size': fontSize})
plt.style.use("seaborn-colorblind")
plt.rcParams['svg.fonttype'] = 'none'
plt.rcParams['mathtext.rm'] = 'Arial'
lineW = 1
```

1.2 Plot intensity traces

[13]:
```
Ndet = nchannel # number of channels to plot
totTime = data.macrotime * np.max(data.det10[:,0])
print("Measurement duration: " + '{:.0f}'.format(totTime) + " s")

# plot intensity trace
Lseg = 0.2
maxseg = int(np.floor(totTime / Lseg))

plt.figure()
for det in range(Ndet):
    if det != 10:
        time = getattr(data, "det" + str(det))[:,0]
        timeAbs = time * data.macrotime
        [Itrace, timeBins] = np.histogram(timeAbs, maxseg)
        plt.plot(timeBins[0:-1], Itrace[0:] / (timeBins[2] - timeBins[1]) / 1e3)

# channel 10
time = getattr(data, "det10")[:,0]
timeAbs = time * data.macrotime
[Itrace, timeBins] = np.histogram(timeAbs, maxseg)
plt.plot(timeBins[0:-1], Itrace[0:] / (timeBins[2] - timeBins[1]) / 1e3,␣
 ↪color='black')
#plt.plot(1e-3*lifetimeBins, histD[:,1], linewidth=lineW*2)
plt.xlabel("time (s)")
output = plt.ylabel("photon count rate (kHz)")
#plt.axis([0, 226, 0, 7])
plt.rcParams['svg.fonttype'] = 'none'
plt.tight_layout()
# plt.savefig('FLFS_time_traces.svg')
```

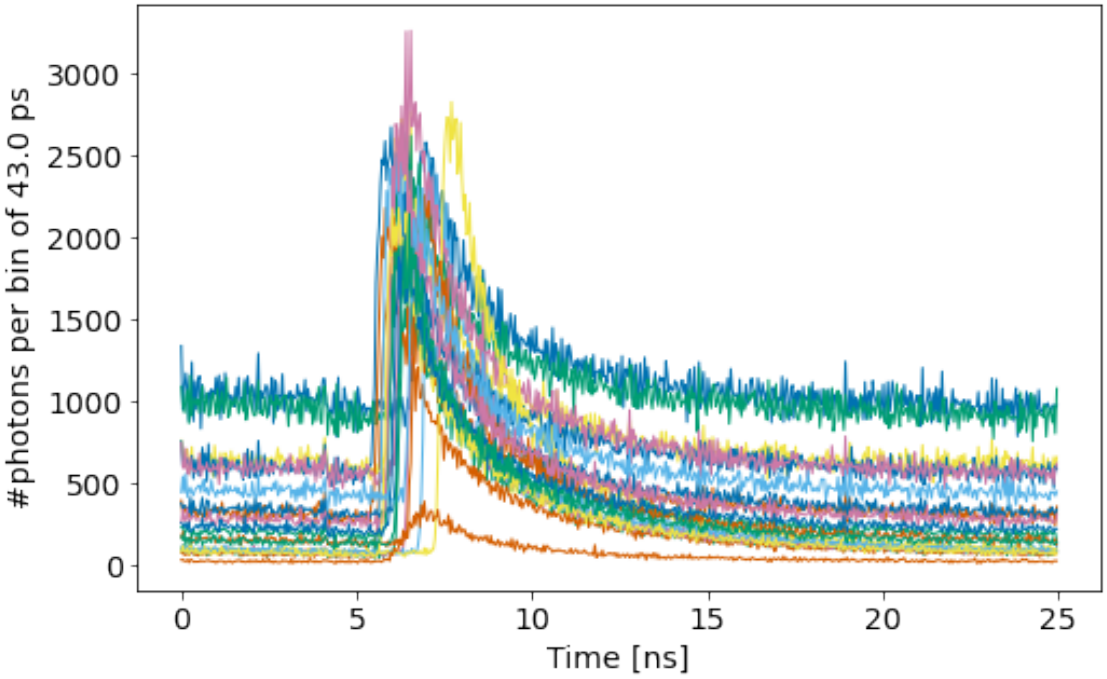
```
Measurement duration: 322 s
```



1.3 Plot lifetime histograms

[14]:
```
MM = nbins
laserF = laser_Hz # laser frequency (Hz)
plt.rcParams.update({'font.size': 14})
plt.figure(figsize=(8,5))
for det in range(Ndet):
    macroTime = getattr(data, "det" + str(det))[:,0] # ps
    microTime = getattr(data, "det" + str(det))[:,1]
    microTime = np.mod(microTime, 1e12 / laserF)
    microTime_flipped = -copy.deepcopy(microTime) + np.max(microTime)
    [Ihist, lifetimeBins] = np.histogram(microTime_flipped, MM)
    lifetimeBins = lifetimeBins[0:-1] * data.microtime * 1e12
    setattr(data, "hist" + str(det), np.transpose(np.stack((lifetimeBins,␣
 ↪Ihist))))
    setattr(data, "det" + str(det), np.transpose([macroTime,␣
 ↪microTime_flipped]))
    lifetimeBinsN = (lifetimeBins - lifetimeBins[0]) / lifetimeBins[1]
    plt.plot(1e-3*lifetimeBins, Ihist, linewidth=lineW)

data.microbintime = 1e-12 * lifetimeBins[1] # s
plt.xlabel("Time [ns]")
```

```
output = plt.ylabel("#photons per bin of " + '{:.1f}'.format(lifetimeBins[1]) +␣
↪" ps")
```



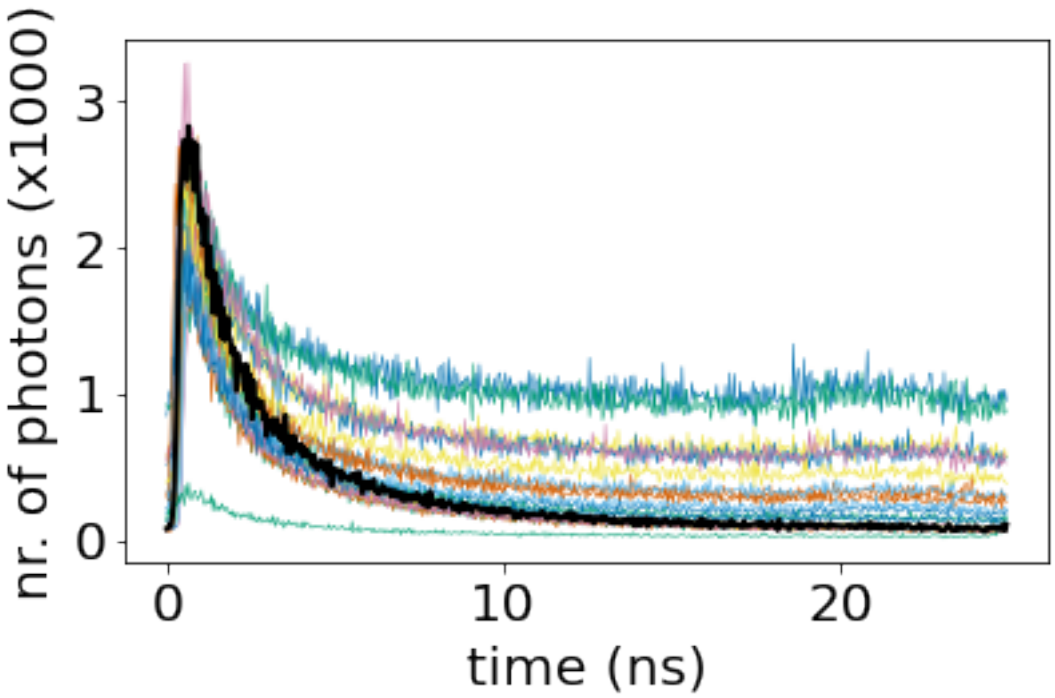
1.4 Align lifetime histograms

[15]:
```
data = alignLifetimeHist(data)
```

[16]:
```
plt.rcParams.update({'font.size': fontSize})
plt.style.use("seaborn-colorblind")
lineW = 1
```

[17]:
```
plt.figure()
#plt.margins(0,0)
for det in range(Ndet):
    histD = getattr(data, "Ahist" + str(det))
    if det != 10:
        plt.plot(1e-3*lifetimeBins, histD[:,1]/1000, linewidth=0.5)

histD = getattr(data, "Ahist" + str(10))
plt.plot(1e-3*lifetimeBins, histD[:,1]/1000, linewidth=lineW*2, color='black')


plt.xlabel("time (ns)")
#plt.axis([0, 25, 0, 3600/1000])
```

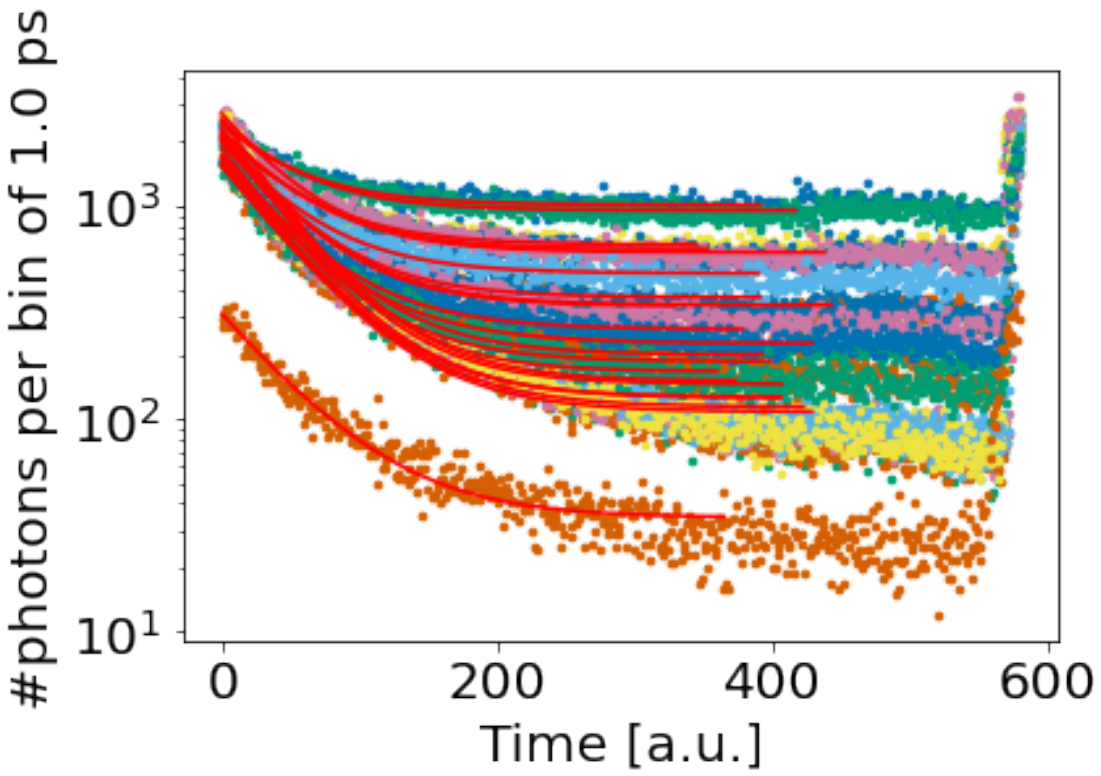
```
output = plt.ylabel("nr. of photons (x1000)") # of 43.4 ps
plt.rcParams['svg.fonttype'] = 'none'
plt.tight_layout()
plt.savefig('FLFS_histograms.svg')
```



1.5 Crop first and last part of microtime histograms to calculate lifetimes and filter functions

[18]:
```
# find index peak and zoom
Tmax = 210 # 210
fitRange = np.zeros((Ndet, 2), dtype='int')
for det in range(Ndet):
    IhistSingle = getattr(data, "hist" + str(det)) # number of photons per bin␣
 ↪of 43.4 ps as a function of time in ps
    Ihist = IhistSingle[:, 1]
    idxStart = np.where(Ihist == np.max(Ihist))[0][0] + 1
    idxStop = np.where(Ihist[Tmax:] == np.min(Ihist[Tmax:]))[0][0] + Tmax + 1
    idxStop = np.minimum(idxStop, len(Ihist) - 1)
    fitRange[det, :] = [idxStart, idxStop]
    T = idxStop - idxStart
    # print("T = " + str(T))
data.fitRange = fitRange
```

[19]:
```
fitresults = np.zeros((Ndet, 3))

plt.figure()
for det in range(Ndet):
    # get histogram detector element i
    IhistSingle = getattr(data, "hist" + str(det)) # number of photons per bin␣
↪of 43.4 ps as a function of time in ps
    Ihist = IhistSingle[:, 1]
    lifetimeBins = IhistSingle[:, 0]
    binTime = lifetimeBins[1]
    lifetimeBins /= binTime # bin numbers

    [idxStart, idxStop] = data.fitRange[det, :]
    fitRangeL = idxStop - idxStart

    Ihist = np.roll(Ihist, -idxStart)
    #lifetimeBins = np.roll(lifetimeBins, -idxStart)

    lifetimeBinsFit = (lifetimeBins[0:fitRangeL])
    IhistFit = Ihist[0:fitRangeL]

    plt.scatter(lifetimeBins, Ihist, s=7)
    plt.xlabel("Time [a.u.]")
    plt.ylabel("#photons per bin of " + '{:.1f}'.format(lifetimeBins[1]) + "␣
↪ps")
    #plt.xlim([0, lifetimeBinsFit[-1]])

    # fit exponential

    fitresult = fitPowerLaw(IhistFit, lifetimeBinsFit, 'exp', [1, 1, 1],␣
↪[60000, 4, 100], [0, 0, -1e4], [1e6, 20, 1e6])
    A      = fitresult.x[0]
    alpha  = fitresult.x[1]
    B      = fitresult.x[2]
    plt.plot(lifetimeBinsFit, A * np.exp(-alpha * lifetimeBinsFit) + B,␣
↪color='r')
    plt.yscale('log')

    fitresults[det, :] = [A, alpha, B]
    # output = plt.title("A = " + '{:.0f}'.format(A) + "  -  tau = " + '{:.3f}'.
↪format(1/alpha) + "  -  B = " + '{:.5f}'.format(B/A))
```



[20]:
```
print("lifetime = (" + '{:.2f}'.format(np.mean(1e9*data.microbintime/
↪fitresults[:, 1])) + " +/- " + '{:.2f}'.format(np.std(1e9*data.microbintime/
↪fitresults[:, 1])) + ") ns")
```

```
lifetime = (2.11 +/- 0.10) ns
```

1.6 Calculate filter functions

1.6.1 Theoretical filter assuming monoexponential decay + offset

[21]:
```
data.fitRange
```

[21]:
```
array([[163, 555],
       [151, 575],
       [140, 570],
       [155, 561],
       [153, 499],
       [165, 539],
       [140, 579],
       [147, 565],
       [136, 580],
       [153, 507],
```

```
        [180, 541],
        [143, 551],
        [149, 541],
        [150, 580],
        [162, 556],
        [150, 481],
        [145, 553],
        [158, 549],
        [149, 528],
        [152, 551],
        [162, 528]])
```

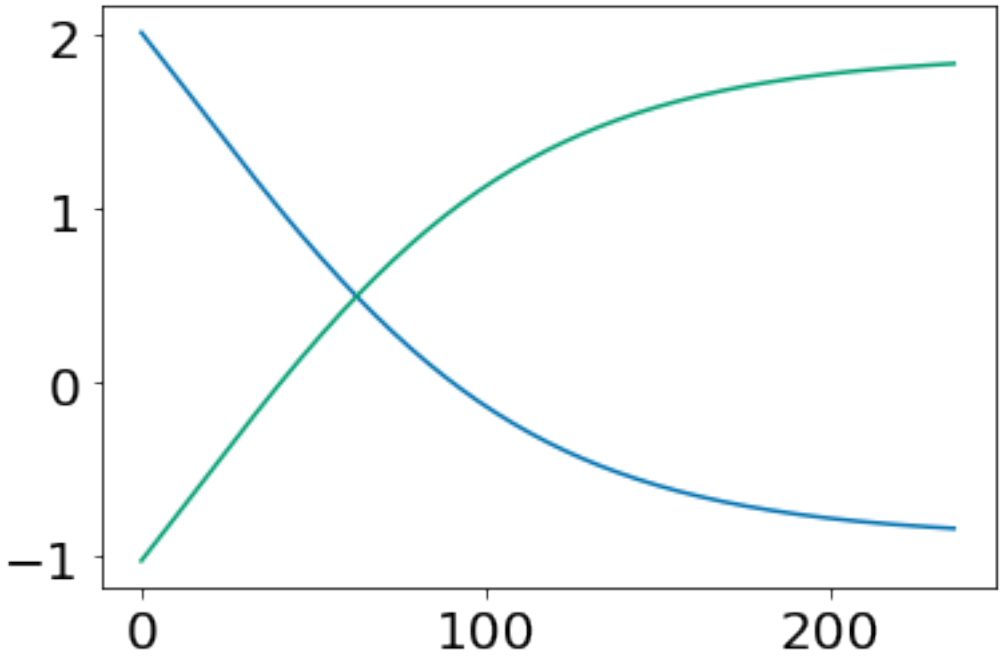
[22]:
```
data.fitRange[:,1] = 400
```

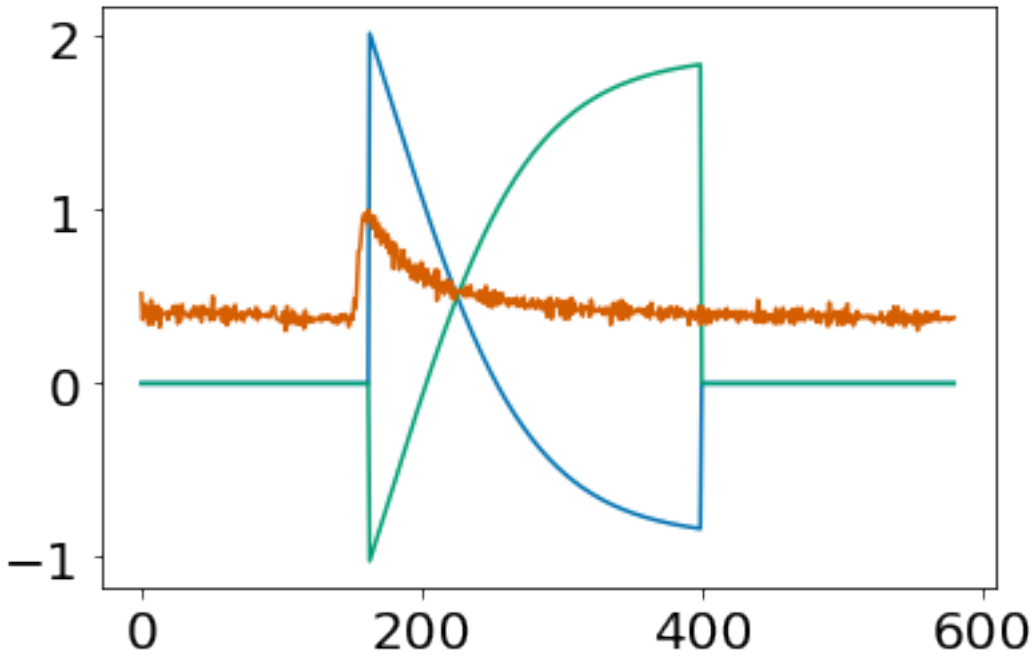
[23]:
```
A = fitresults[:, 0]
alpha = fitresults[:, 1]
B = fitresults[:, 2]
```

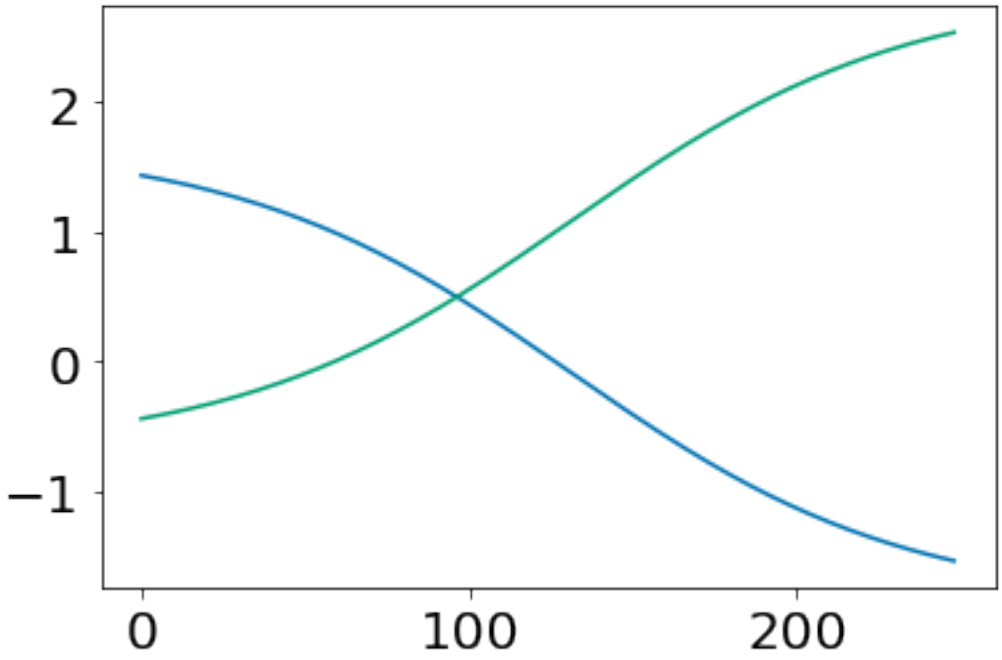
[24]:
```
data.fitRange[data.fitRange > len(Ihist)] = len(Ihist)
```

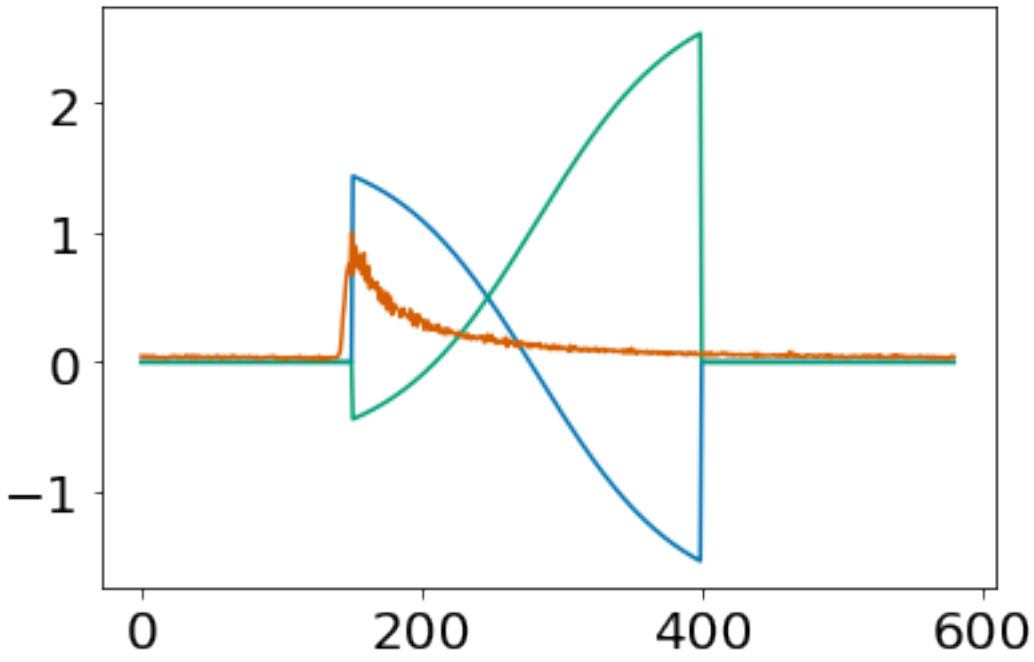
[25]:
```
# filtersTheo contains all filters [det element, filtervalues, filter number]
filtersTheo = np.zeros((Ndet, len(Ihist), 2))
for det in range(Ndet):
    # calculate filter
    tau = 1 / alpha[det]
    T = int(np.diff(data.fitRange[det, :]))
    offset = B[det] / A[det] # relative offset, wrt amplitude
    Ftheo = filterAP(0, tau, T, offset, False)

    # pad zeros before and after filter domain
    [startIdx, stopIdx] = data.fitRange[det, :]
    filtersTheo[det, startIdx:stopIdx, :] = np.transpose(Ftheo)

    # figure
    histSingle = getattr(data, 'hist' + str(det))
    histSingle[:,1] /= np.max(histSingle[:,1])
    plt.figure()
    plt.plot(filtersTheo[det, :, 0])
    plt.plot(filtersTheo[det, :, 1])
    plt.plot(histSingle[:,1])
```

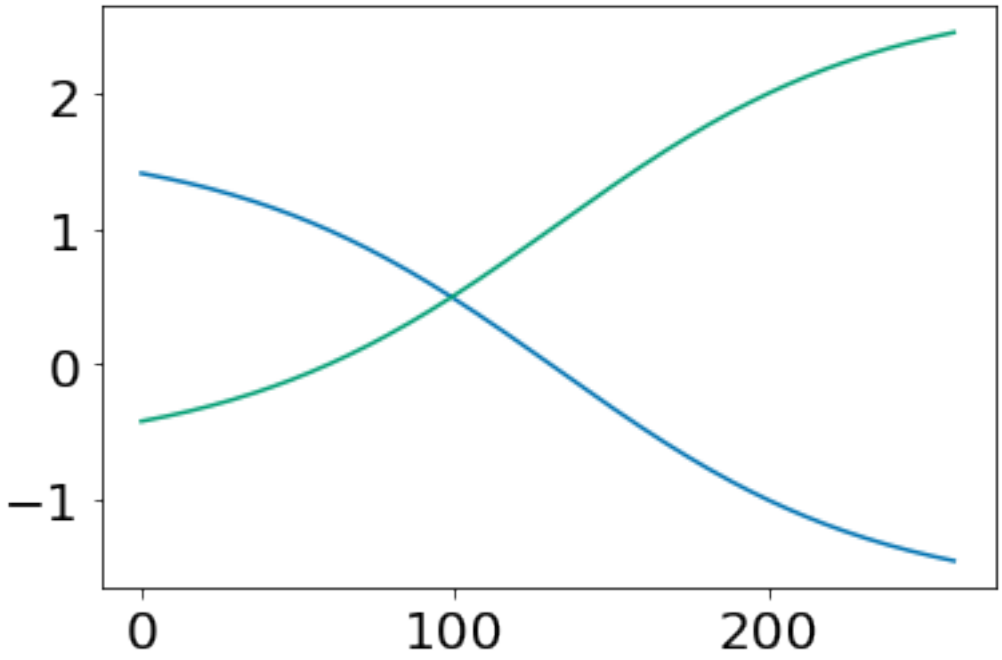
```
/home/labuser/myDev/timetaggingplatform/dataProcessing/libs/spad_ffs/spad_fcs/fi
lterAP.py:57: RuntimeWarning: More than 20 figures have been opened. Figures
created through the pyplot interface (`matplotlib.pyplot.figure`) are retained
until explicitly closed and may consume too much memory. (To control this
warning, see the rcParam `figure.max_open_warning`).
  plt.figure()
```

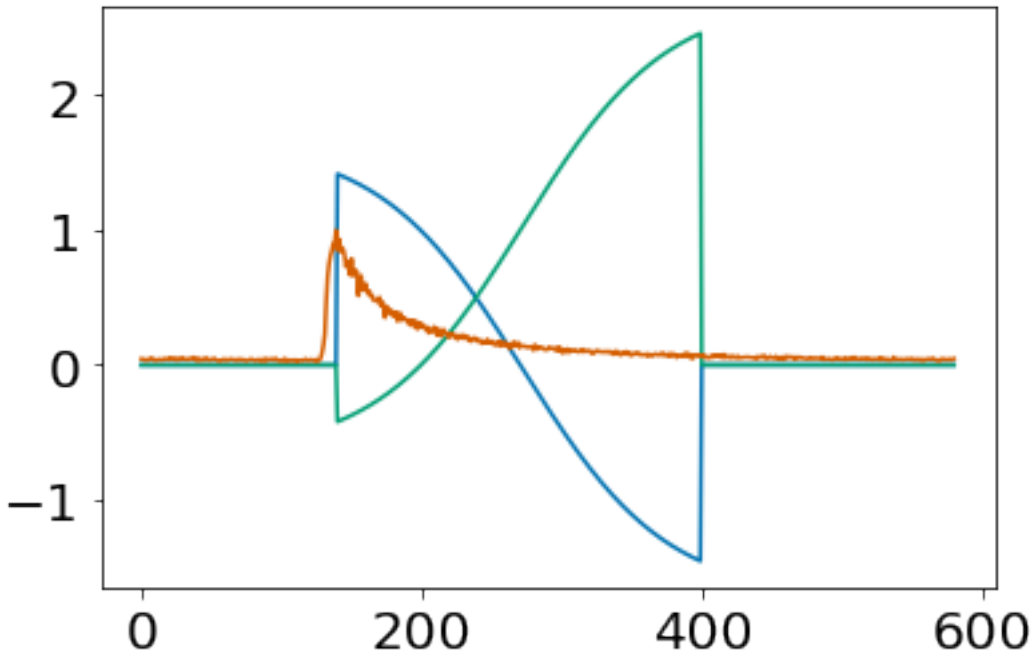

2
1
0
−1
0
100
200

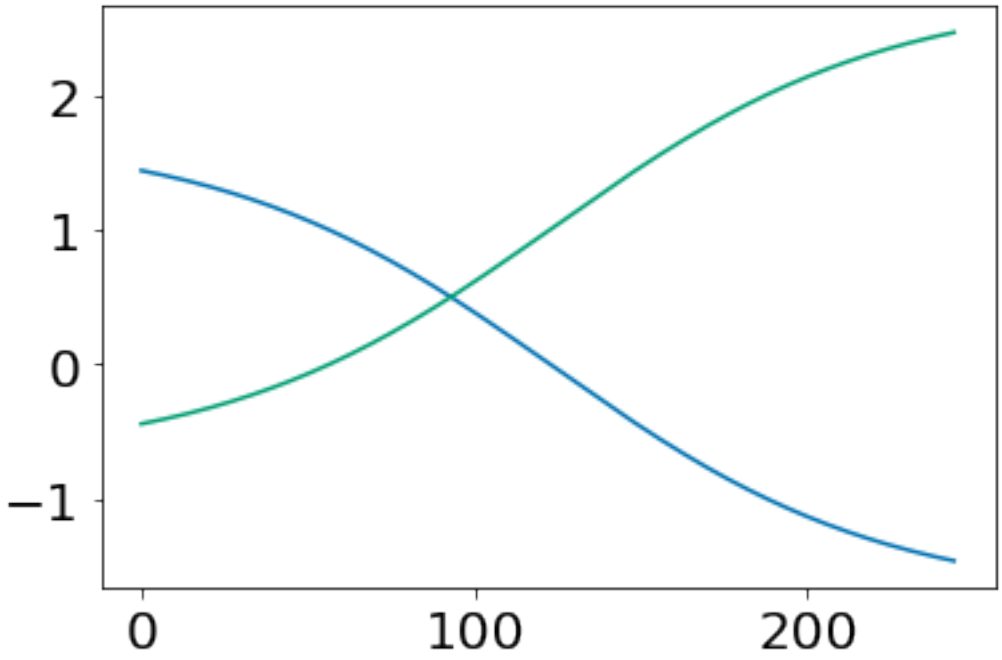

2
1
0
−1
0
200
400
600

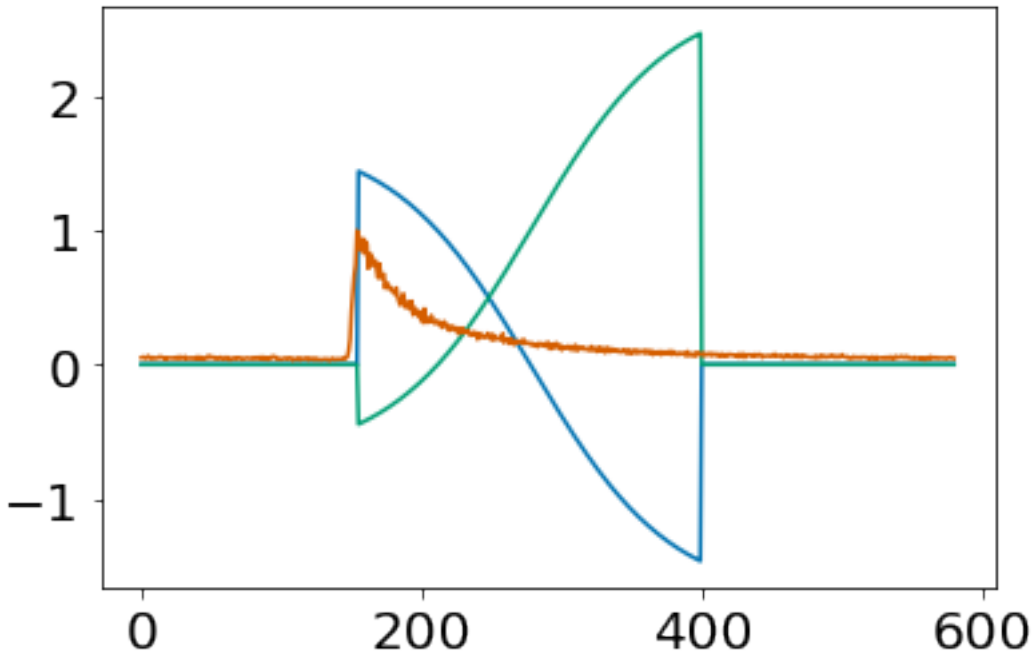

2
1
0
−1
0
100
200

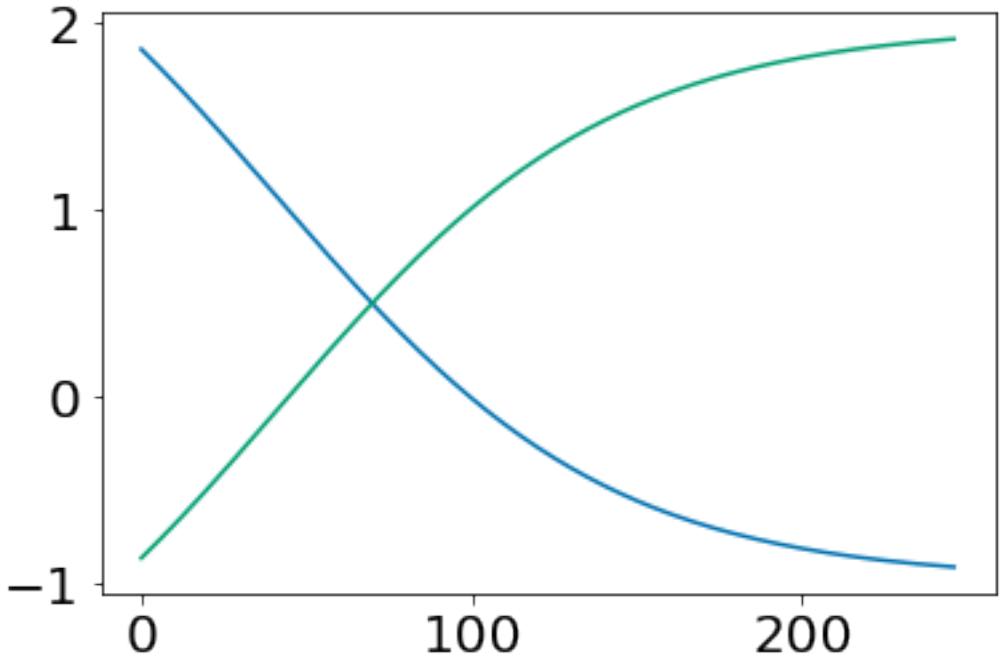

2
1
0
−1
0
200
400
600

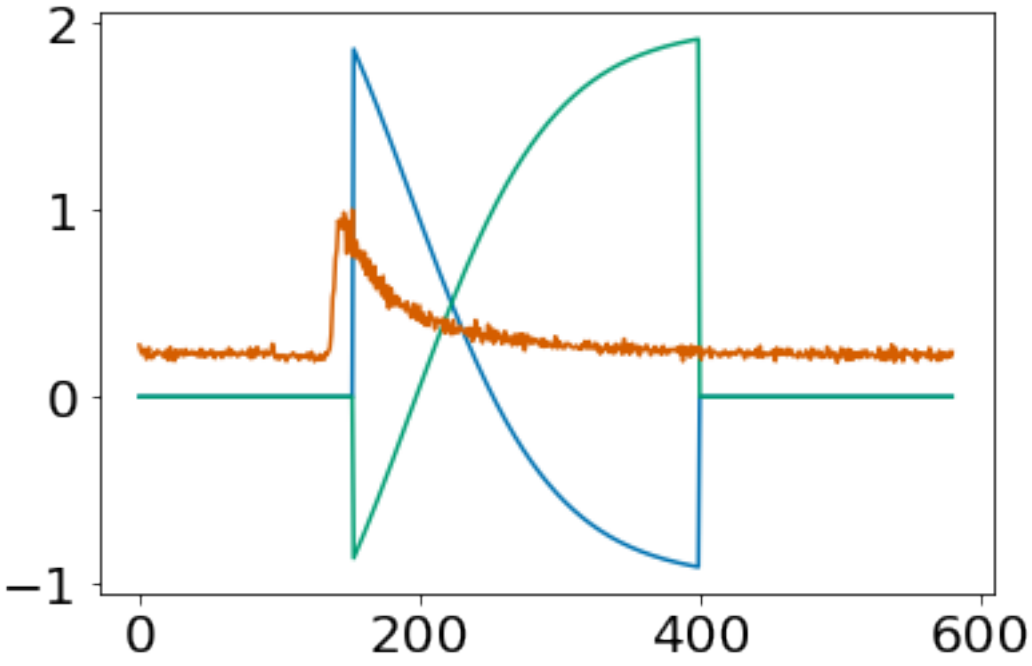

2
1
0
−1
0
100
200

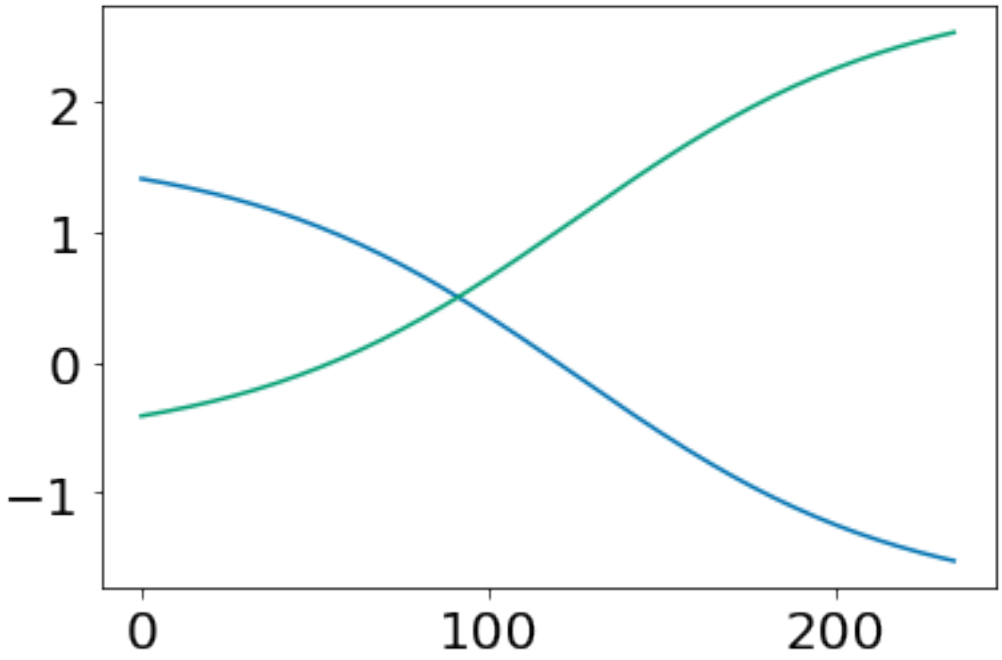

2
1
0
−1
0
200
400
600

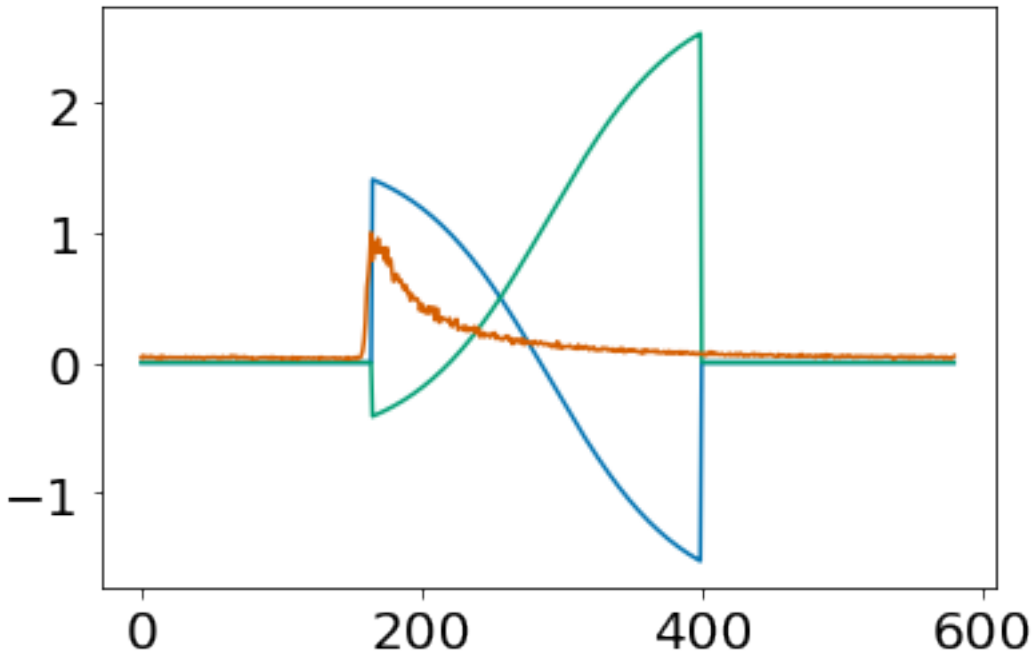

2
1
0
−1
0
100
200

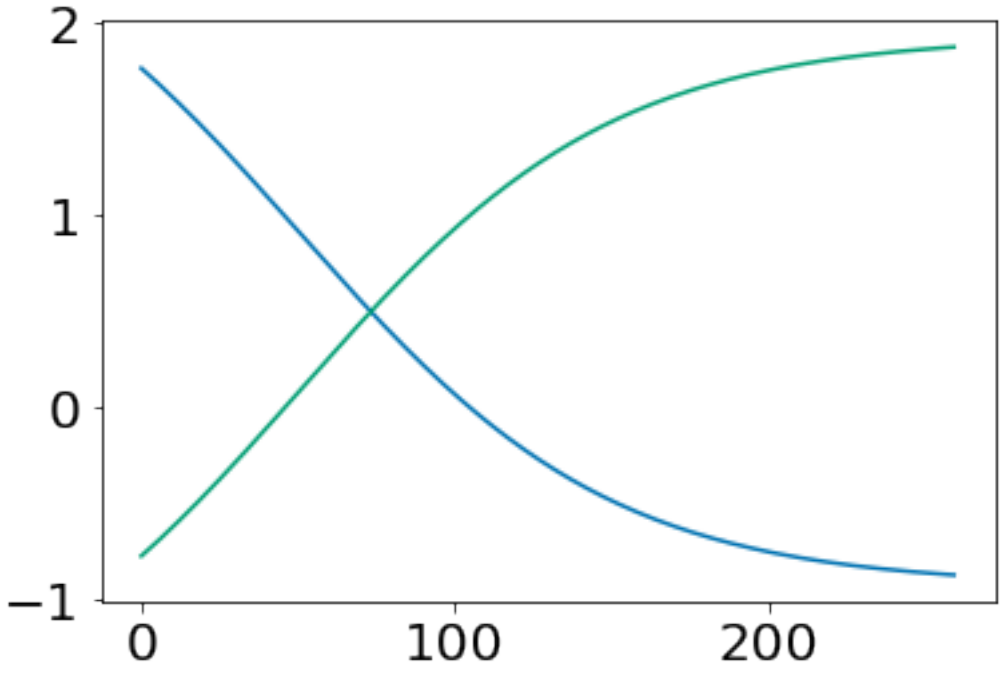

2
1
0
−1
0
200
400
600

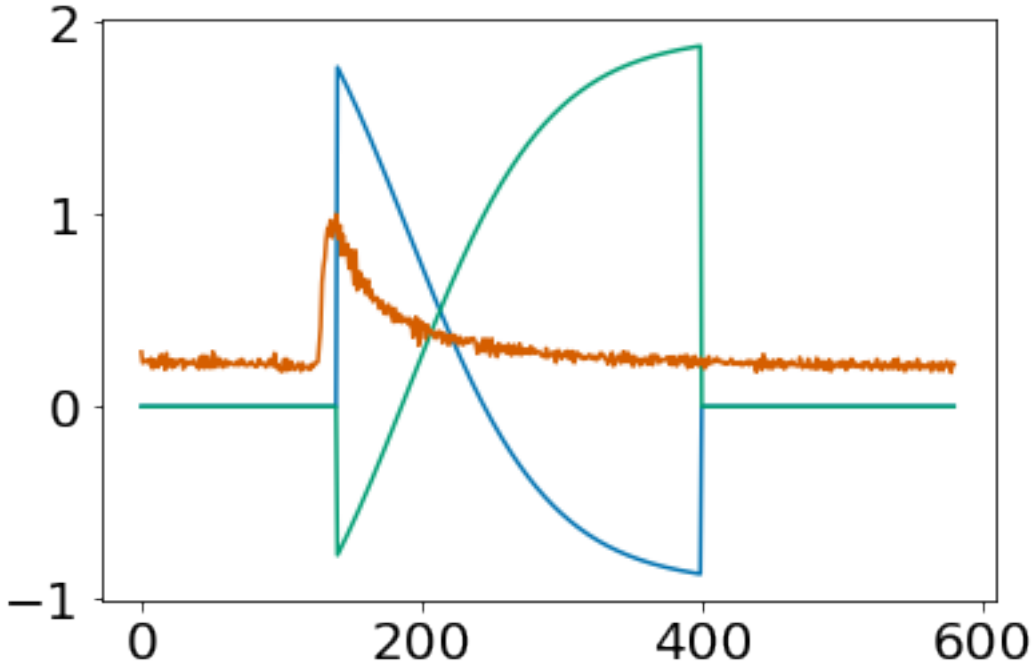

2
1
0
−1
0
100
200

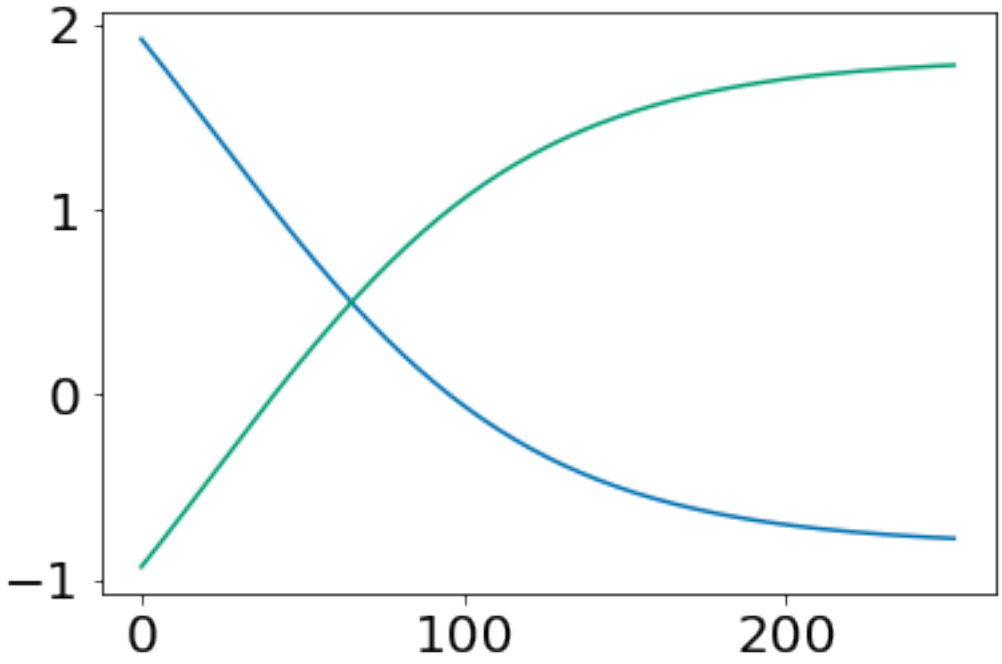

2
1
0
−1
0
200
400
600

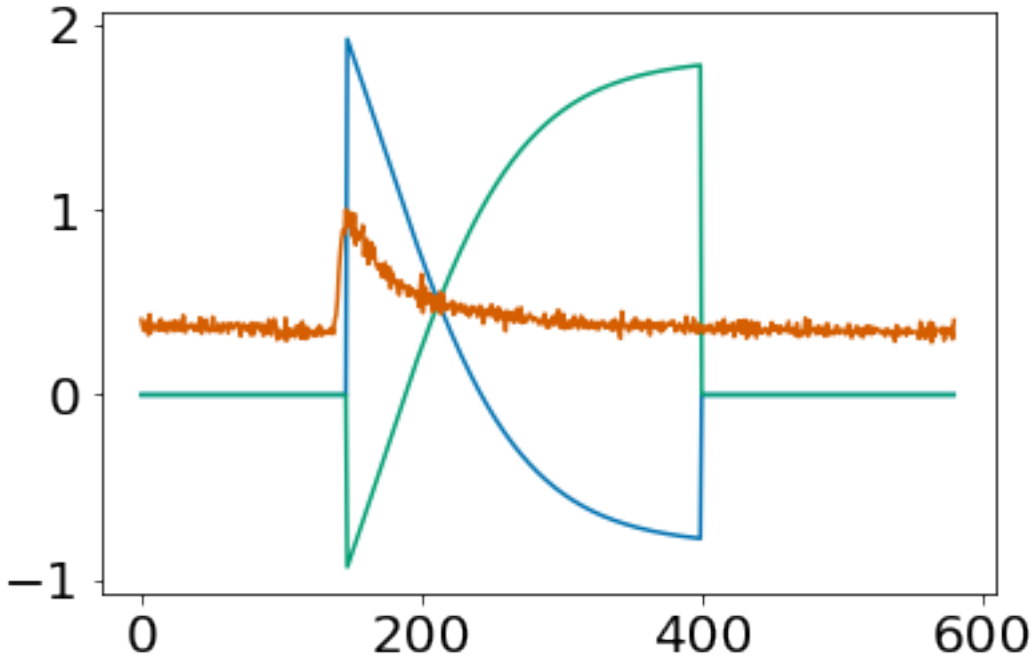

2
1
0
−1
0
100
200

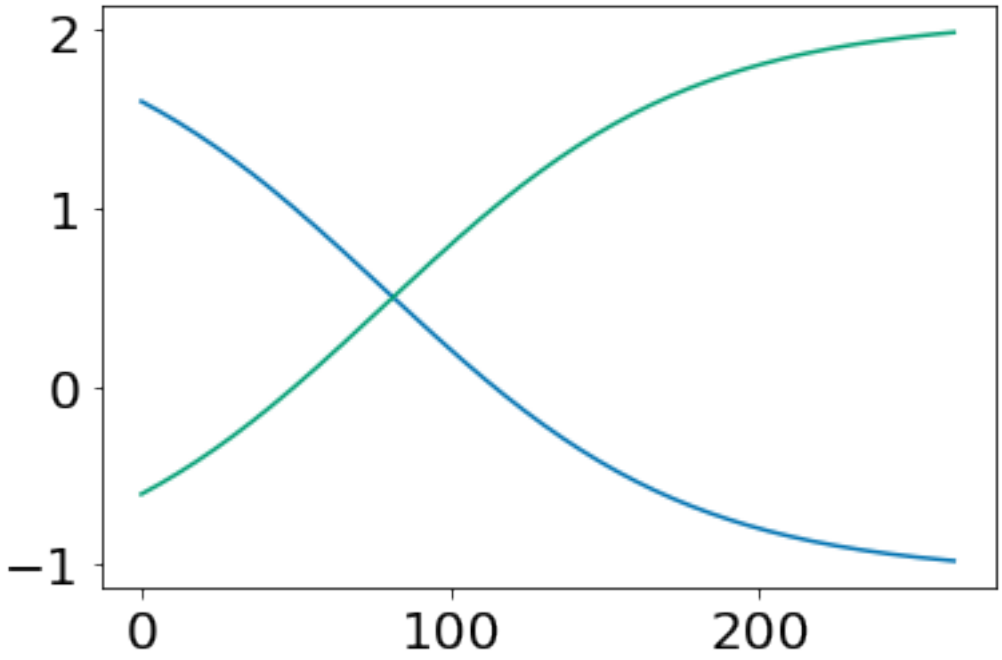

2
1
0
−1
0
200
400
600

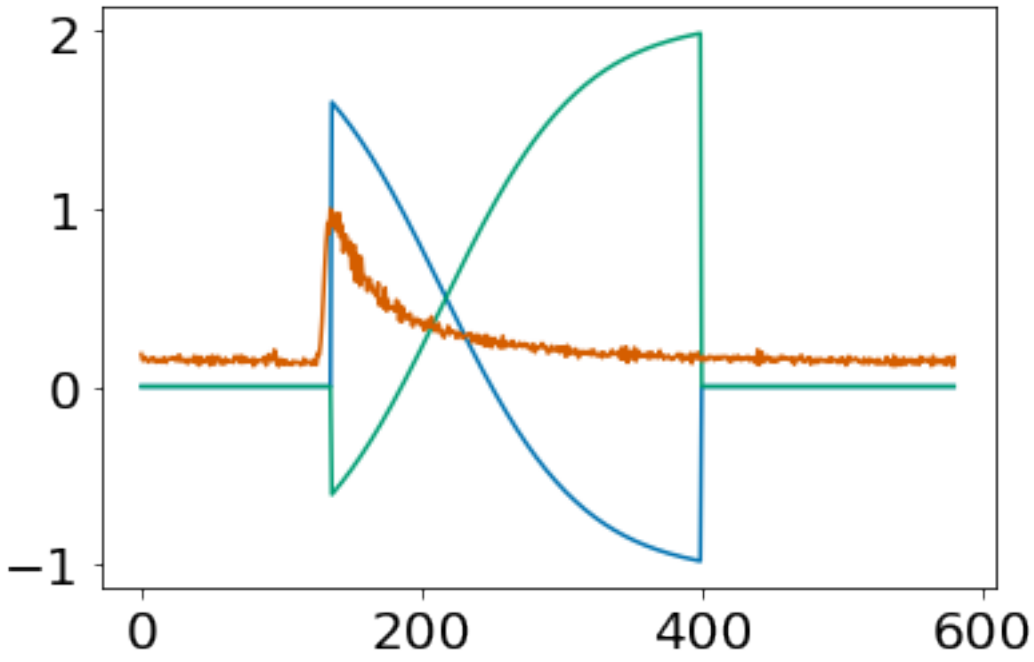

2
1
0
−1
0
100
200

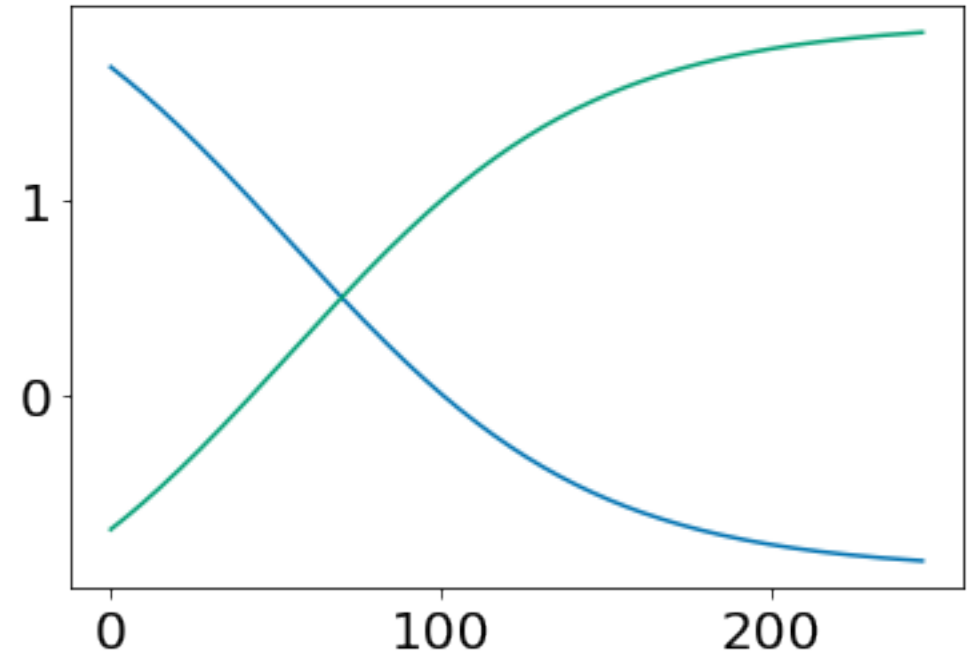

2
1
0
−1
0
200
400
600

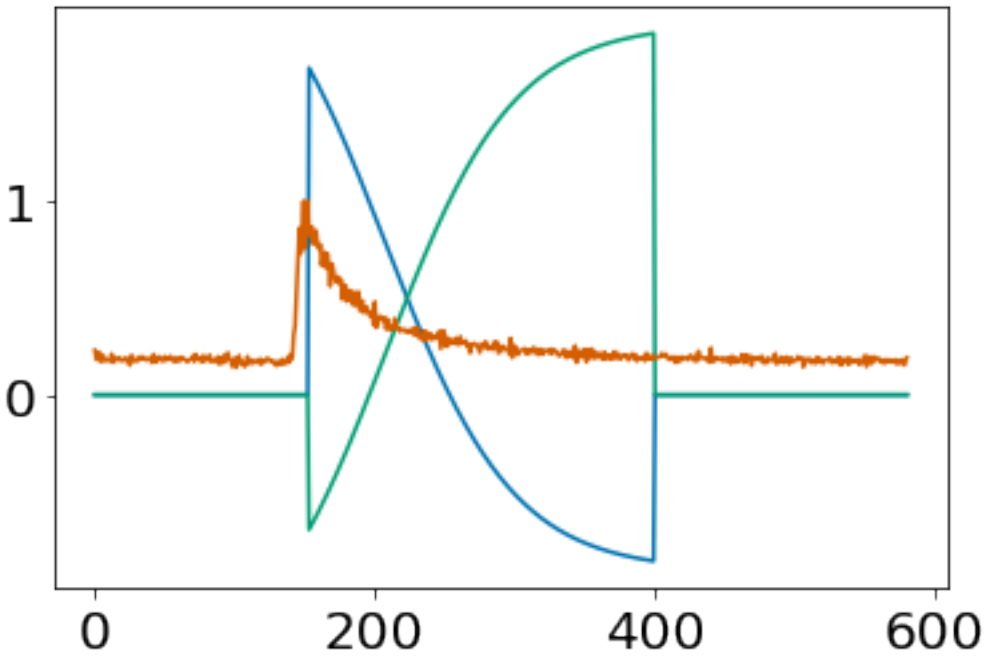

2
1
0
−1
0
100
200

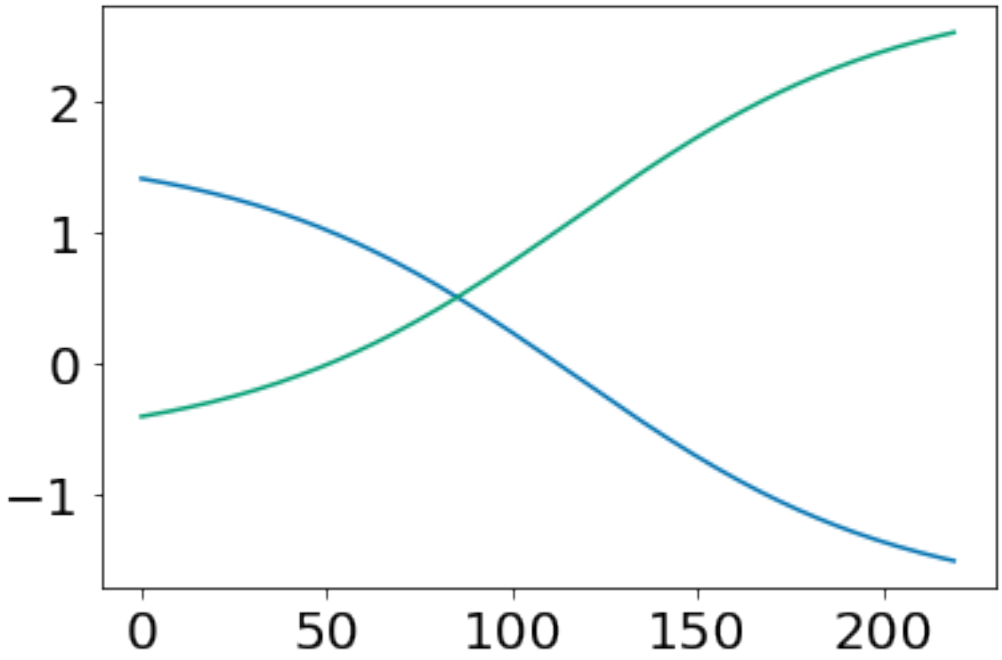

2
1
0
−1
0
200
400
600

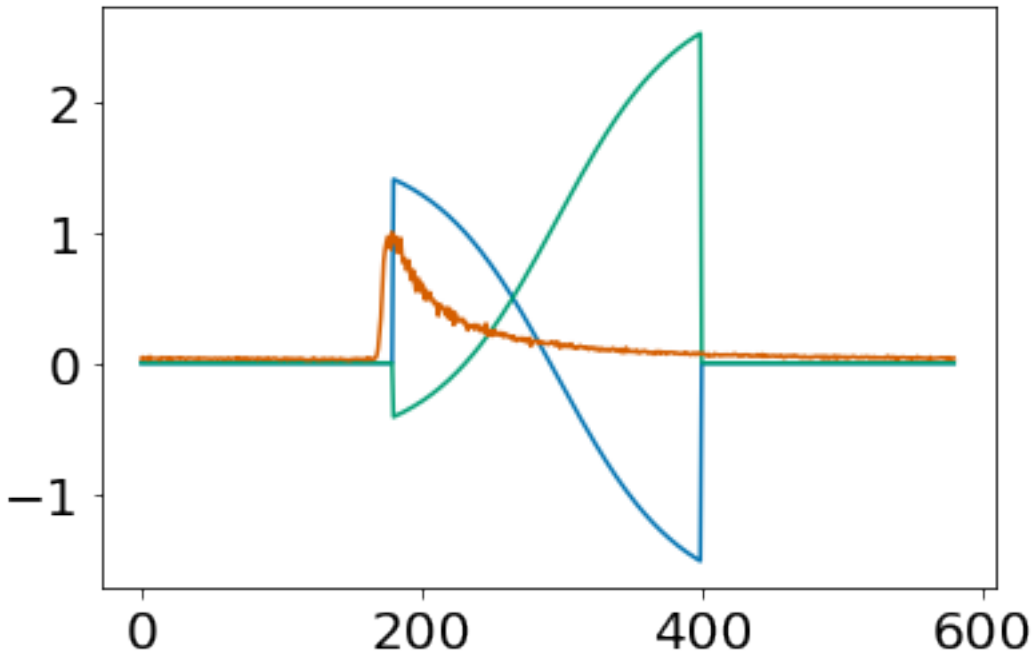

2
1
0
−1
0
100
200

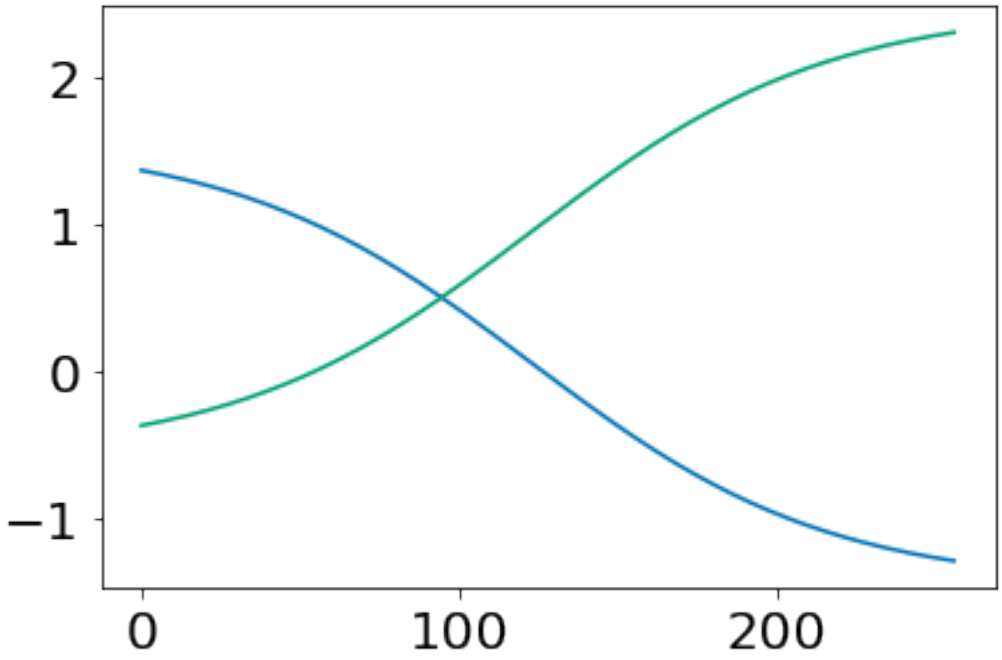

2
1
0
−1
0
200
400
600

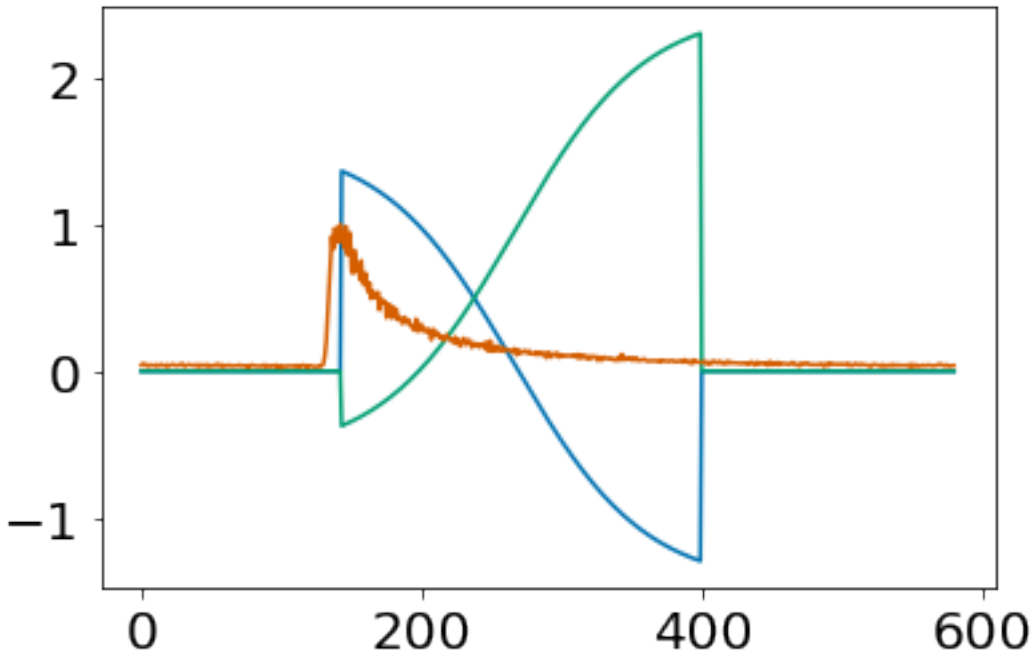

1
0
0
100
200

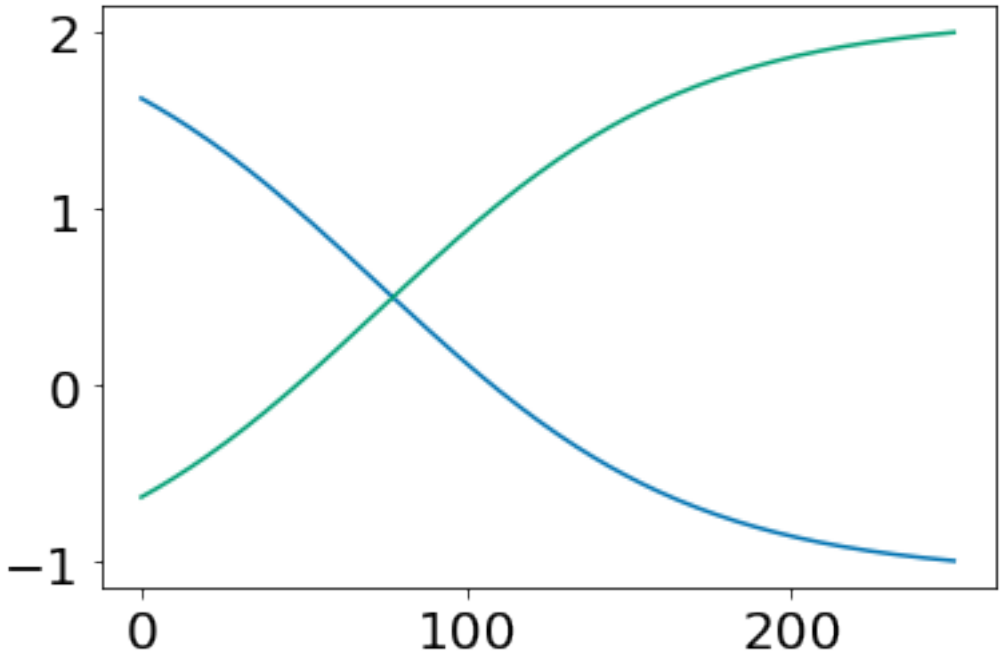

1
0
0
200
400
600

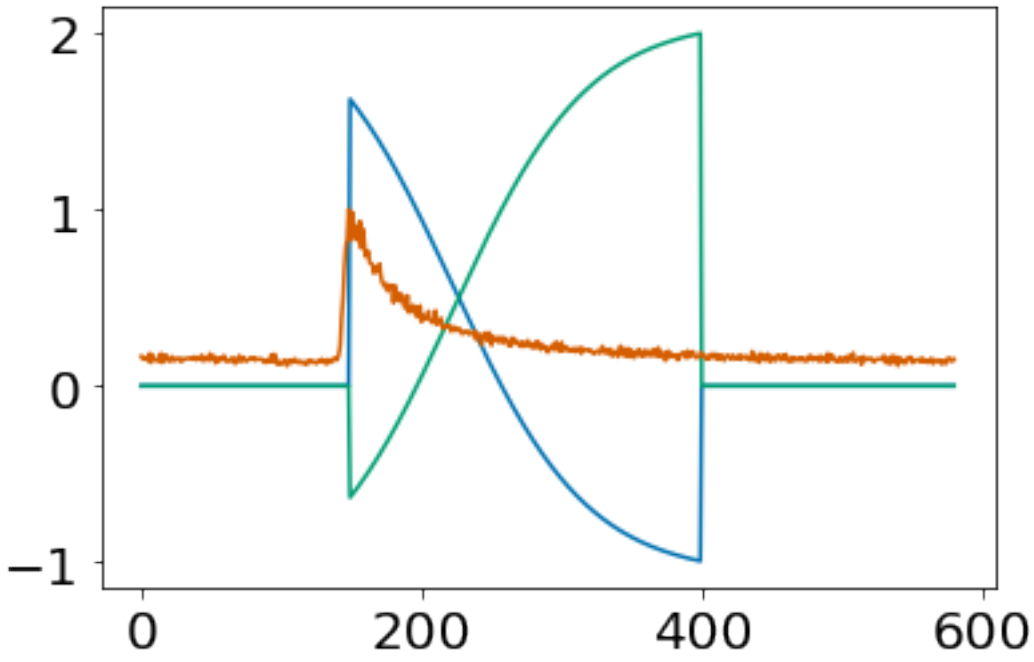

2
1
0
−1
0
50
100
150
200

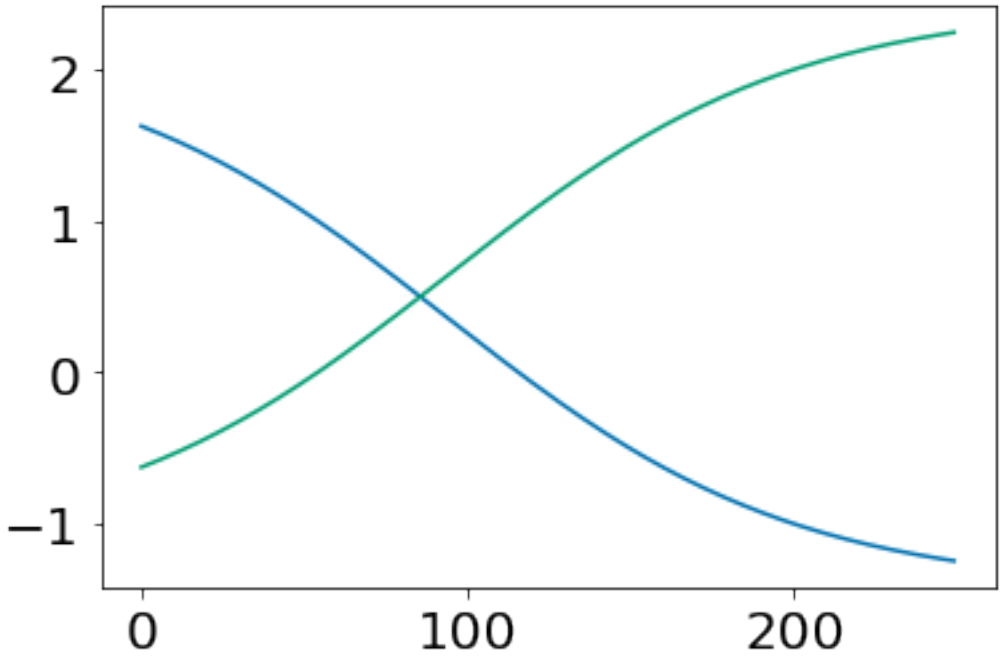

2
1
0
−1
0
200
400
600

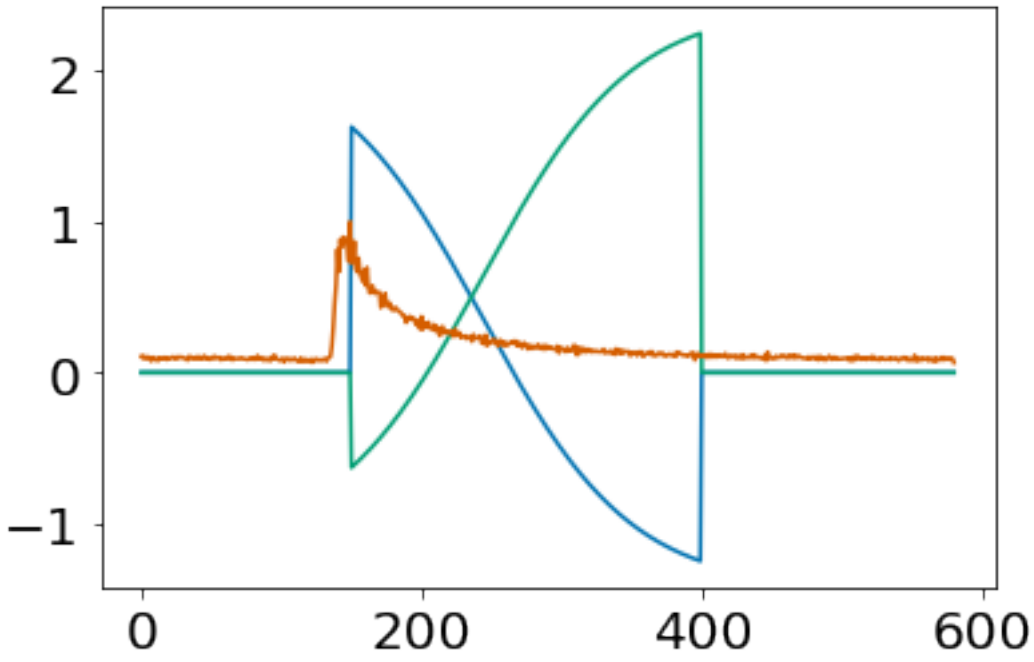

2
1
0
−1
0
100
200

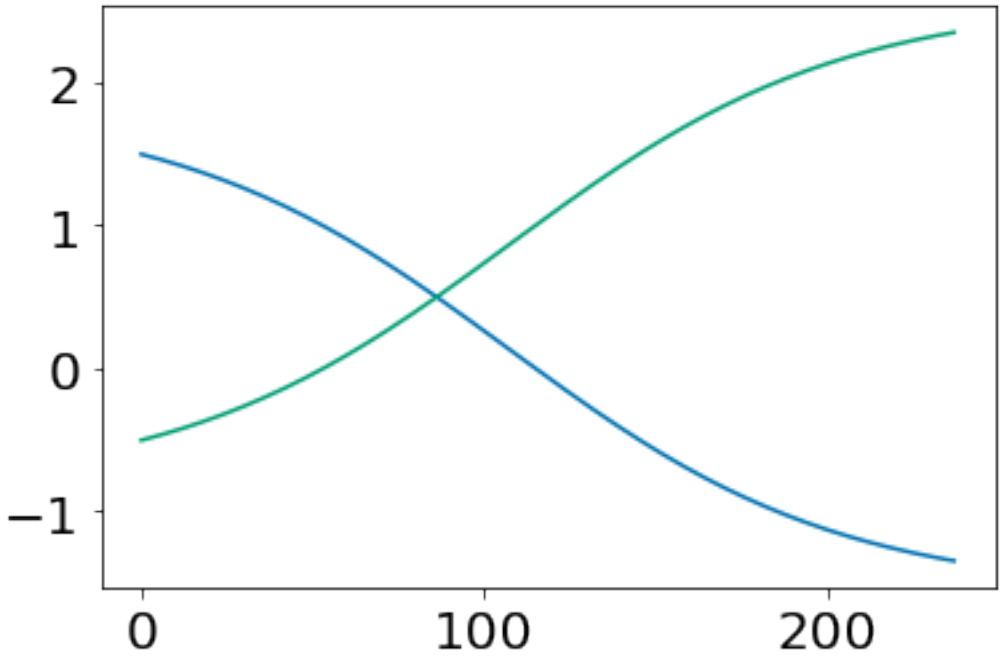

2
1
0
−1
0
200
400
600

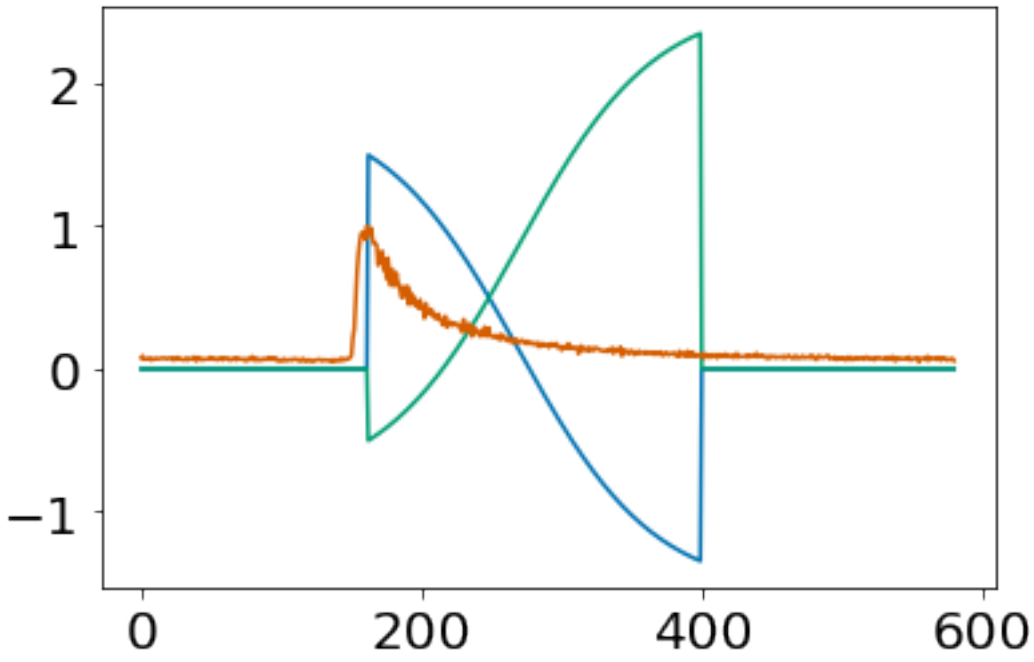

2
1
0
−1
0
100
200

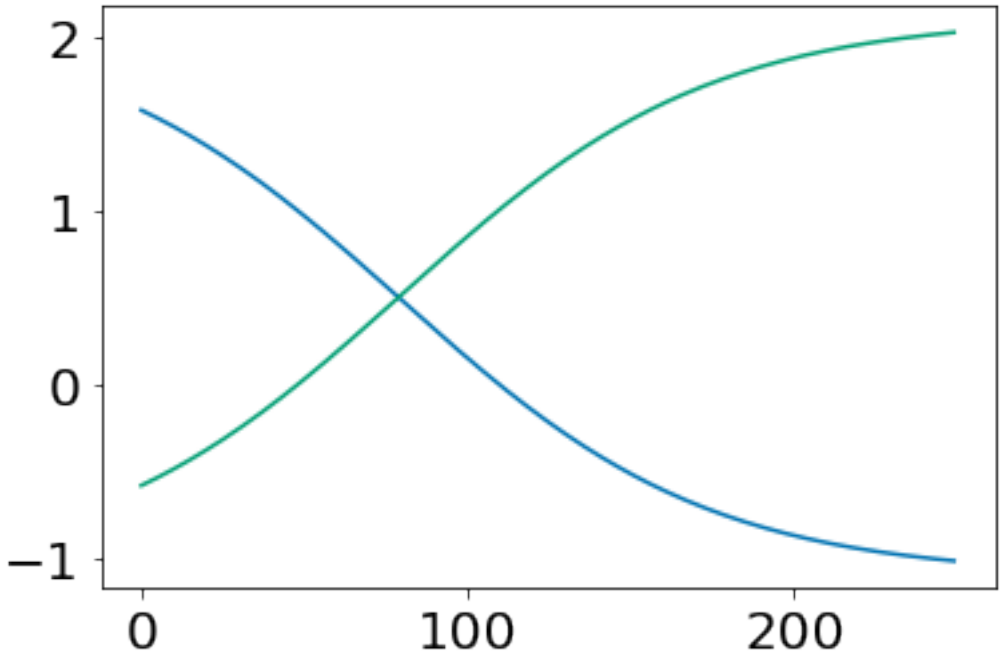

2
1
0
−1
0
200
400
600

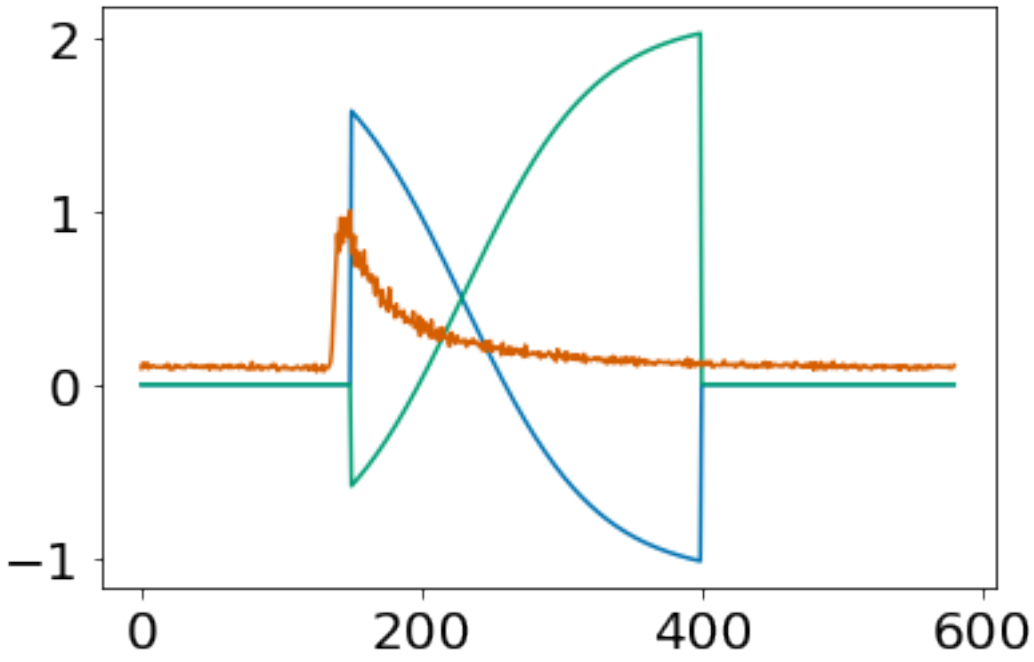

2
1
0
−1
0
100
200

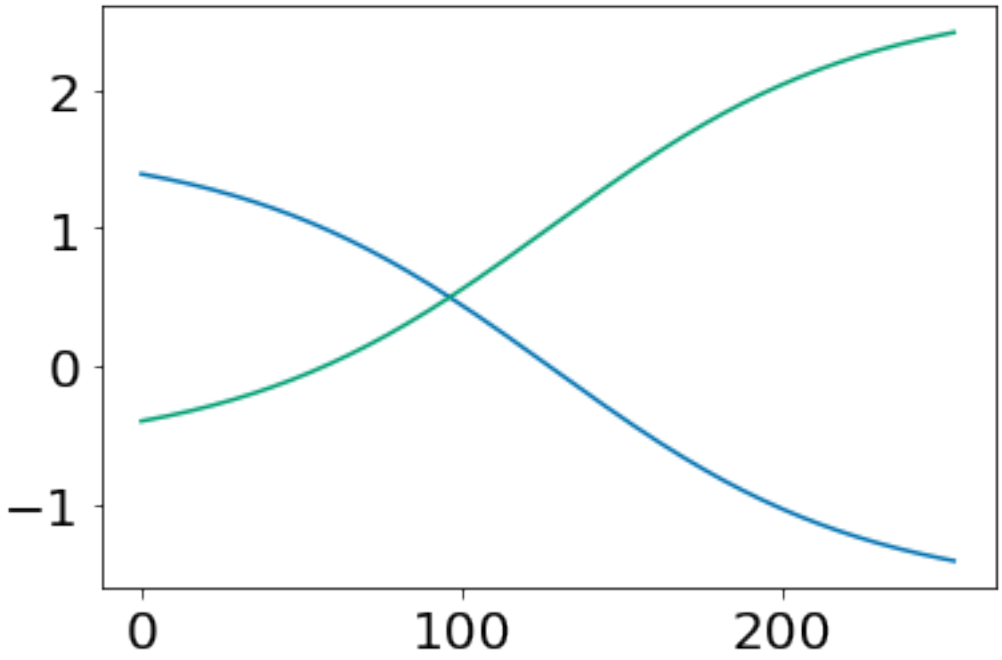

2
1
0
−1
0
200
400
600

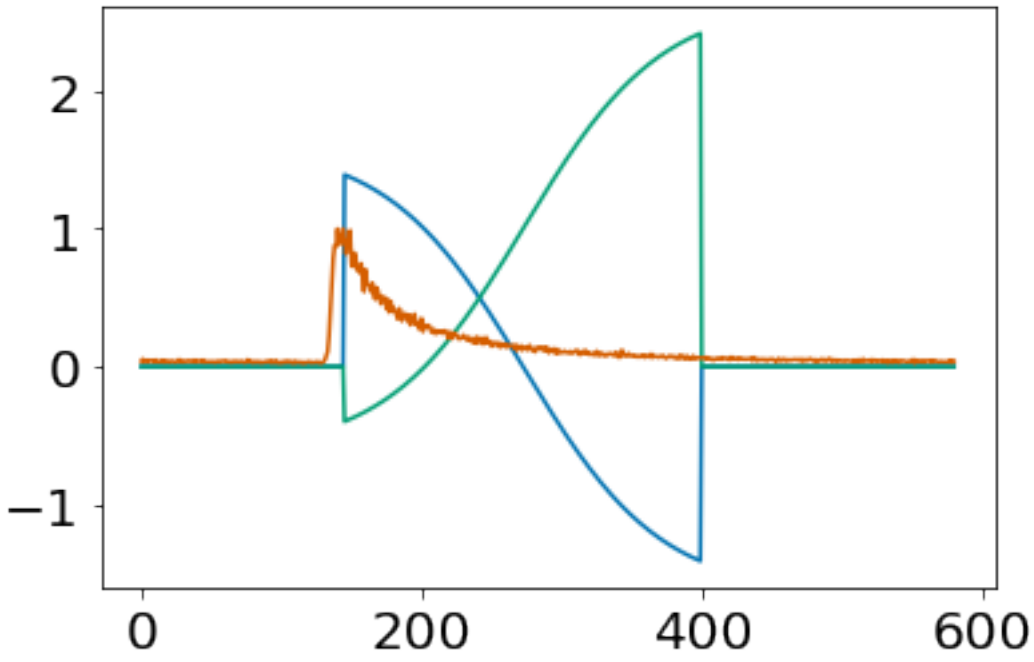

2
1
0
−1
0
100
200

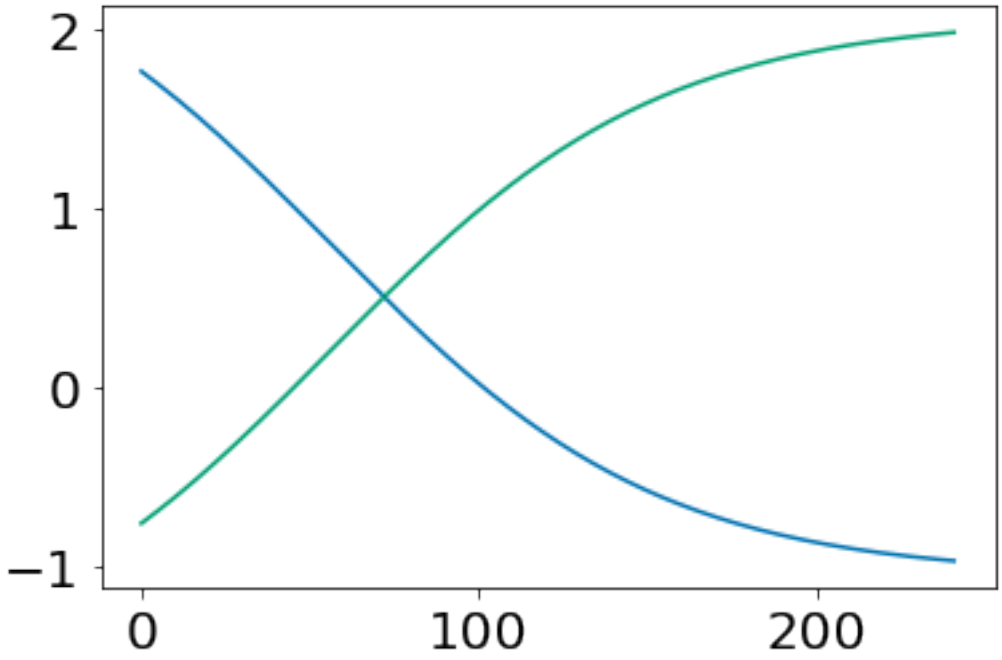

2
1
0
−1
0
200
400
600

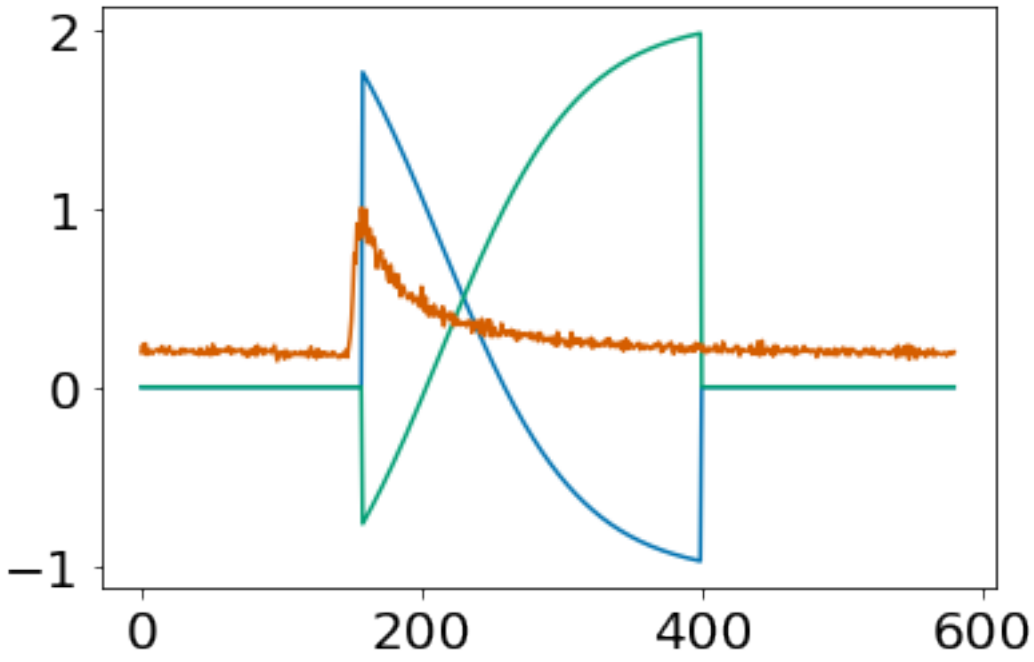

2
1
0
−1
0
100
200

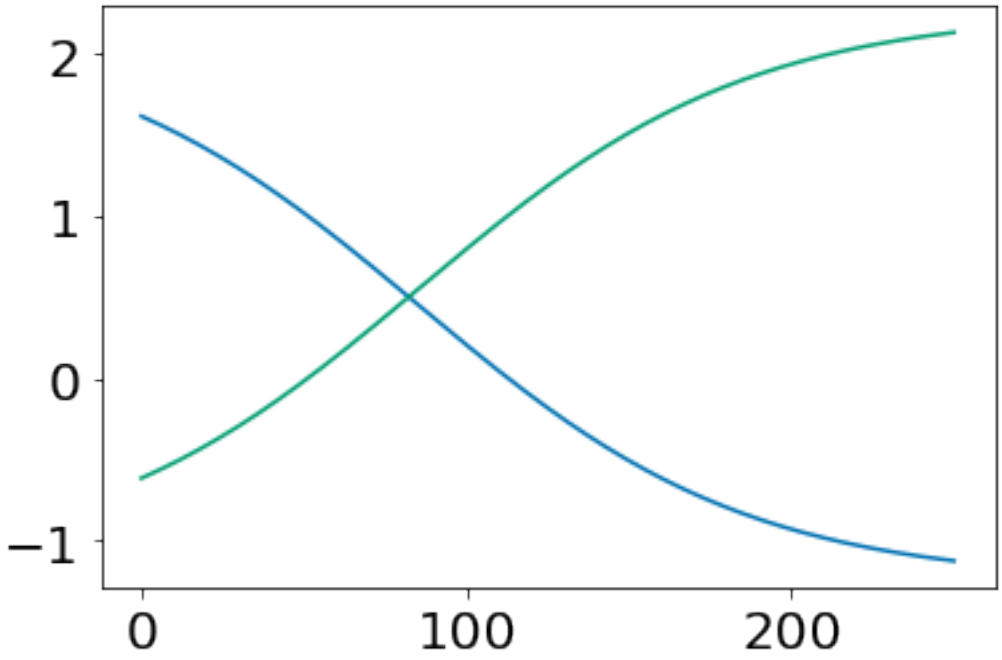

2
1
0
−1
0
200
400
600

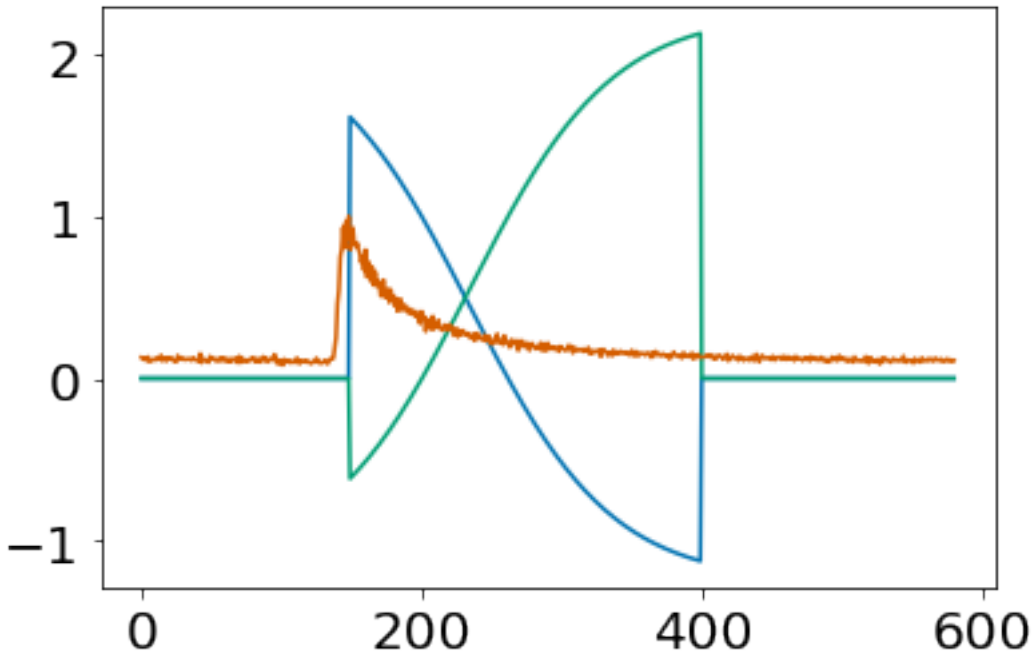

2
1
0
−1
0
100
200

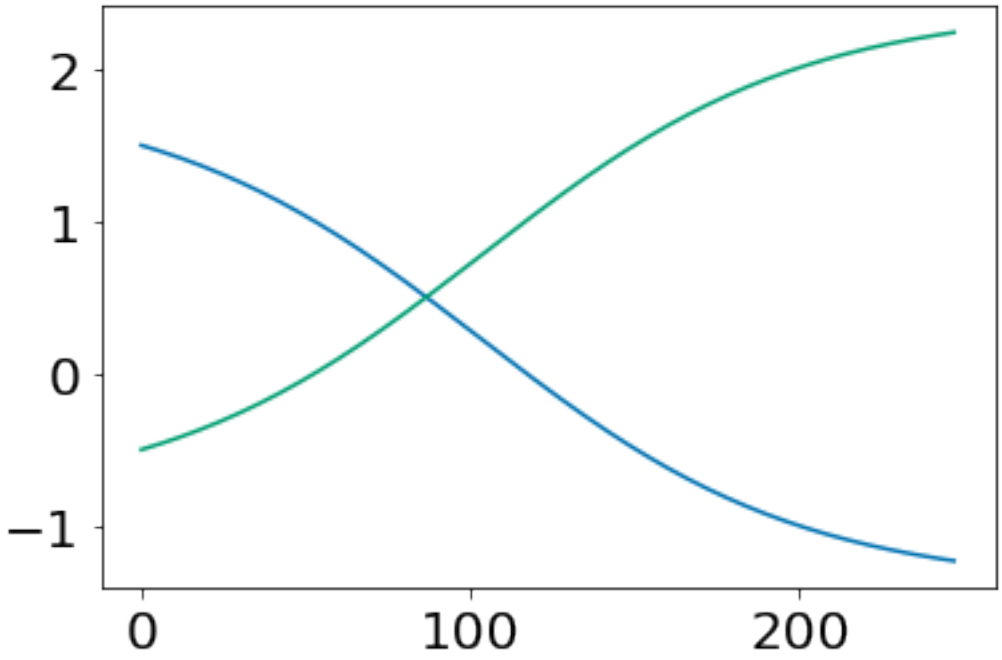

2
1
0
−1
0
200
400
600

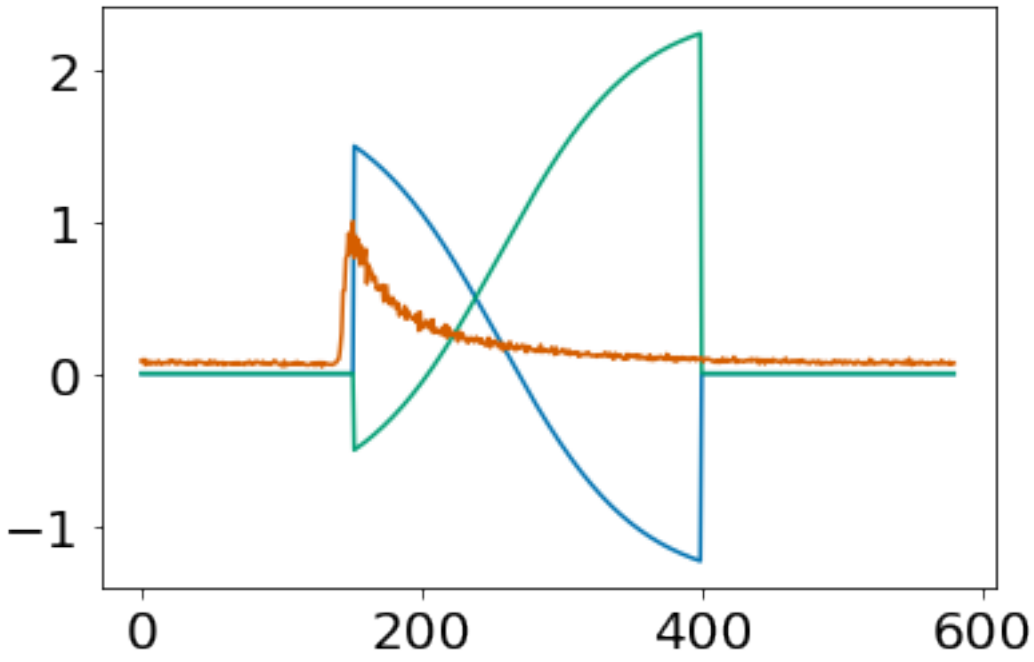

2
1
0
−1
0
100
200

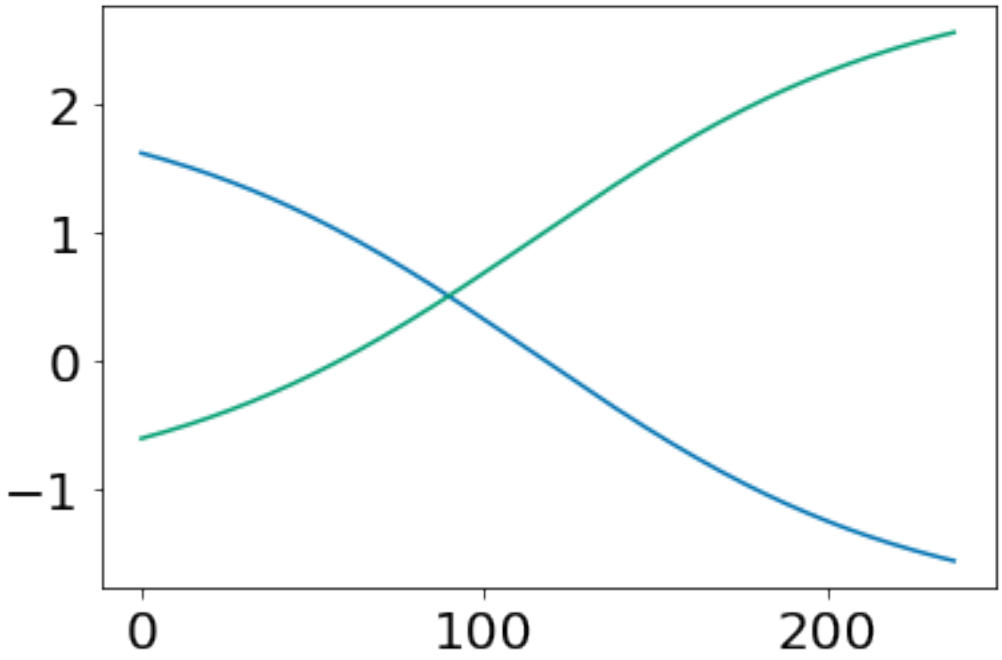

2
1
0
−1
0
200
400
600

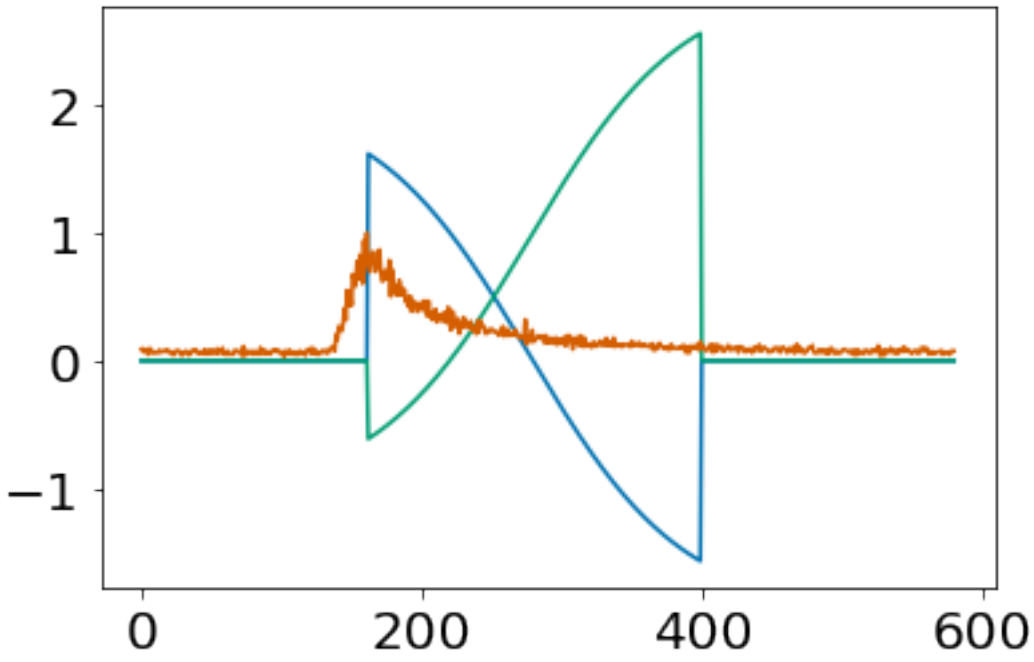

2
1
0
−1
0
100
200


2
1
0
−1
0
200
400
600


2
1
0
−1
0
100
200

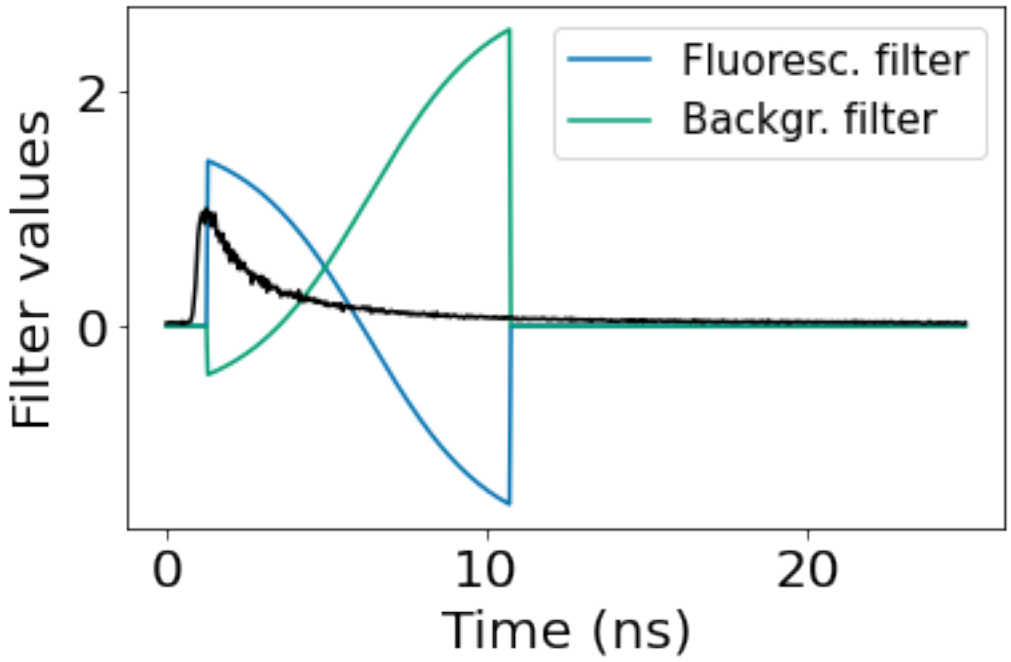

2
1
0
−1
0
200
400
600





[26]:
```
lifetimeBins_plot = lifetimeBins*binTime
```

[27]:
```
histSingle = getattr(data, 'hist' + str(10))
histSingle[:,1] /= np.max(histSingle[:,1])

plt.figure()
roll = -150
plt.plot(1e-3*lifetimeBins_plot, np.roll(filtersTheo[10, :, 0], roll),␣
 ↪label='Fluoresc. filter')
plt.plot(1e-3*lifetimeBins_plot, np.roll(filtersTheo[10, :, 1], roll),␣
 ↪label='Backgr. filter')
plt.plot(1e-3*lifetimeBins_plot, np.roll(histSingle[:,1], roll), color='black')
plt.legend(fontsize=15)
plt.xlabel("Time (ns)")
plt.ylabel('Filter values')
# plt.axis([0, 25, -1.4, 2.4])
plt.rcParams['svg.fonttype'] = 'none'
plt.tight_layout()
plt.savefig('FLFS_filters.svg', bbox_inches='tight')
```



1.6.2 Filter using theoretical filter functions

[28]:
```
data = aTimesFiltered(data, filtersTheo, False)
```

```
Calculating filtered photon streams det0
Calculating filtered photon streams det1
Calculating filtered photon streams det2
```

```
Calculating filtered photon streams det3
Calculating filtered photon streams det4
Calculating filtered photon streams det5
Calculating filtered photon streams det6
Calculating filtered photon streams det7
Calculating filtered photon streams det8
Calculating filtered photon streams det9
Calculating filtered photon streams det10
Calculating filtered photon streams det11
Calculating filtered photon streams det12
Calculating filtered photon streams det13
Calculating filtered photon streams det14
Calculating filtered photon streams det15
Calculating filtered photon streams det16
Calculating filtered photon streams det17
Calculating filtered photon streams det18
Calculating filtered photon streams det19
Calculating filtered photon streams det20
```

[29]:
```
G = aTimes2CorrsParallel(data, list(range(0, nchannel)), accuracy=100, taumax=1/
↪data.macrotime, split=10)
```

```
Calculating correlation 0
   Filter 0
   Filter 1
   Filter 2
Calculating correlation 1
   Filter 0
   Filter 1
   Filter 2
Calculating correlation 2
   Filter 0
   Filter 1
   Filter 2
Calculating correlation 3
   Filter 0
   Filter 1
   Filter 2
Calculating correlation 4
   Filter 0
   Filter 1
   Filter 2
Calculating correlation 5
   Filter 0
   Filter 1
   Filter 2
Calculating correlation 6
   Filter 0
```

```
   Filter 1
   Filter 2
Calculating correlation 7
   Filter 0
   Filter 1
   Filter 2
Calculating correlation 8
   Filter 0
   Filter 1
   Filter 2
Calculating correlation 9
   Filter 0
   Filter 1
   Filter 2
Calculating correlation 10
   Filter 0
   Filter 1
   Filter 2
Calculating correlation 11
   Filter 0
   Filter 1
   Filter 2
Calculating correlation 12
   Filter 0
   Filter 1
   Filter 2
Calculating correlation 13
   Filter 0
   Filter 1
   Filter 2
Calculating correlation 14
   Filter 0
   Filter 1
   Filter 2
Calculating correlation 15
   Filter 0
   Filter 1
   Filter 2
Calculating correlation 16
   Filter 0
   Filter 1
   Filter 2
Calculating correlation 17
   Filter 0
   Filter 1
   Filter 2
Calculating correlation 18
   Filter 0
```

```
   Filter 1
   Filter 2
Calculating correlation 19
   Filter 0
   Filter 1
   Filter 2
Calculating correlation 20
   Filter 0
   Filter 1
   Filter 2
```

Store correlations to .csv files

[30]:
```
corr2csv(G, fname[0:-3], limits=[0, 0], chunks=0)
```

[31]:
```
data_head_raw, data_filename_raw = os.path.split(fname)
```

Load correlations from .csv files

[32]:
```
G = FCSLoadG(data_filename_raw[0:-3] + '_', folderName=data_head_raw,␣
↪printFileNames=True)
```

```
det11F0_average
det14F2_average
det2F0_average
det5F2_average
det0F0_average
det0F1_average
det0F2_average
det10F0_average
det10F1_average
det10F2_average
det2F1_average
det2F2_average
det3F0_average
det3F1_average
det3F2_average
det4F0_average
det4F1_average
det4F2_average
det5F0_average
det5F1_average
det6F0_average
det6F1_average
det6F2_average
det7F0_average
det7F1_average
det7F2_average
det8F0_average
```

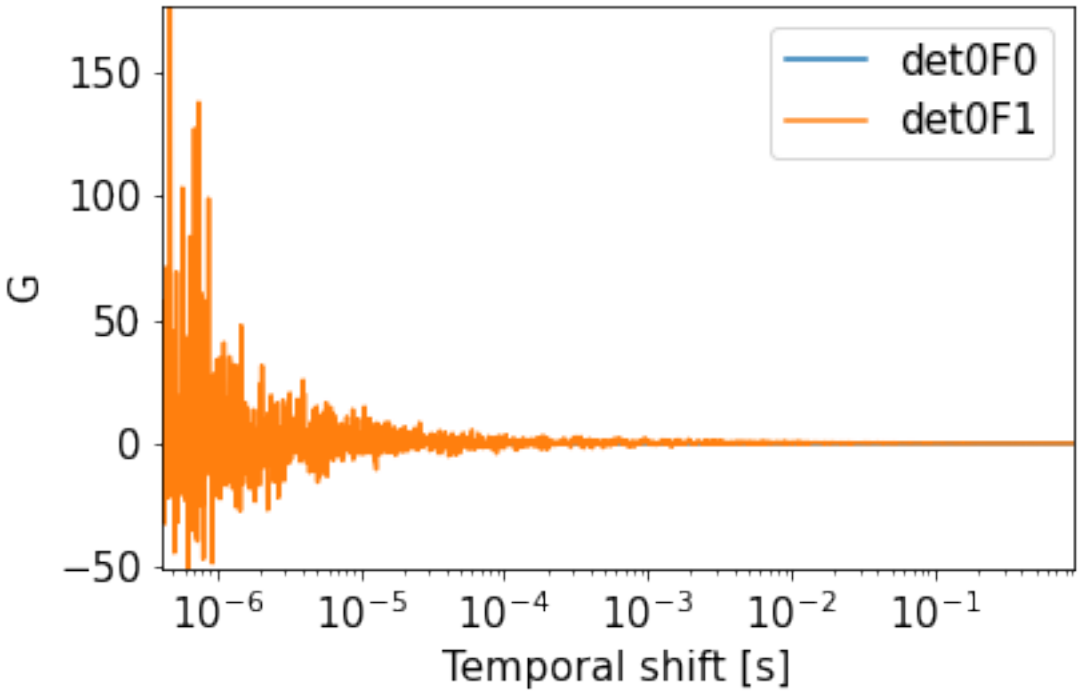
```
det8F1_average
det8F2_average
det9F0_average
det9F1_average
det9F2_average
det11F1_average
det11F2_average
det12F0_average
det12F1_average
det12F2_average
det13F0_average
det13F1_average
det13F2_average
det14F0_average
det14F1_average
det15F0_average
det15F1_average
det15F2_average
det16F0_average
det16F1_average
det16F2_average
det17F0_average
det17F1_average
det17F2_average
det18F0_average
det18F1_average
det18F2_average
det19F0_average
det19F1_average
det19F2_average
det1F0_average
det1F1_average
det1F2_average
det20F0_average
det20F1_average
det20F2_average
--------------------------
63 files found.
--------------------------
```

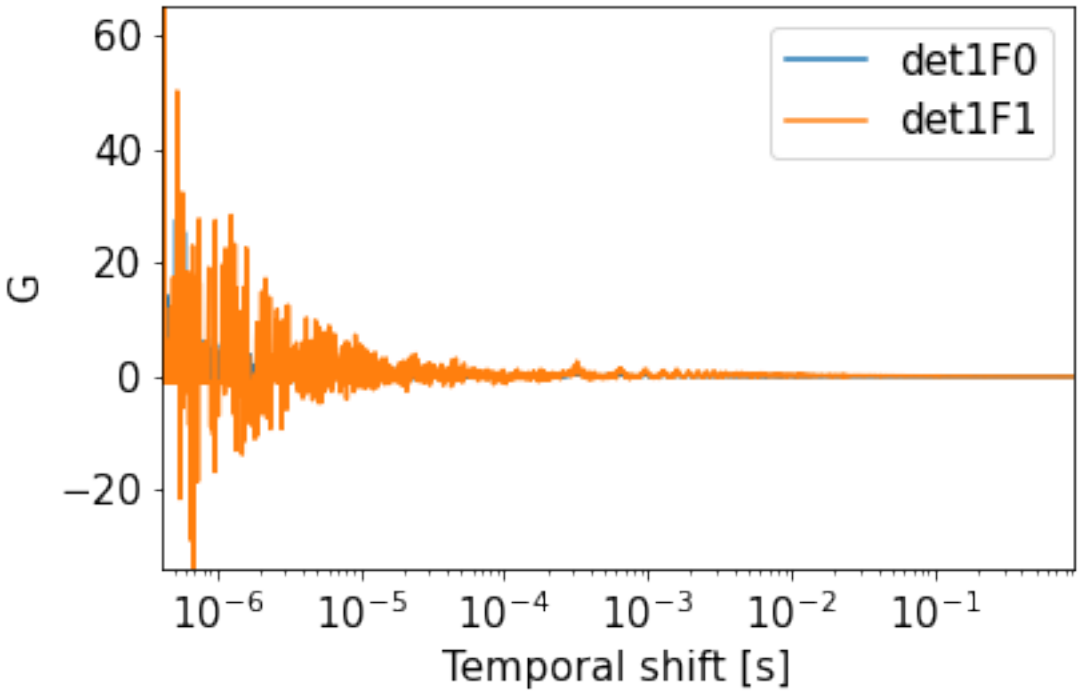
1.7 Plot correlations

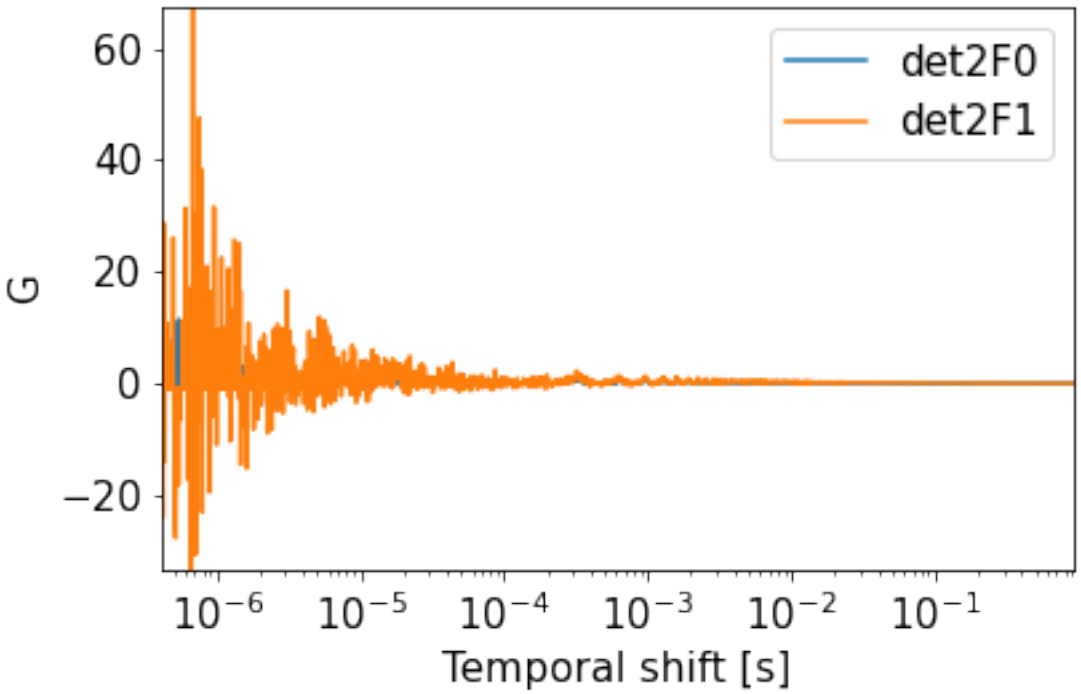
[33]:
```
start = 1200
stop = -10
for i in range(nchannel):
```

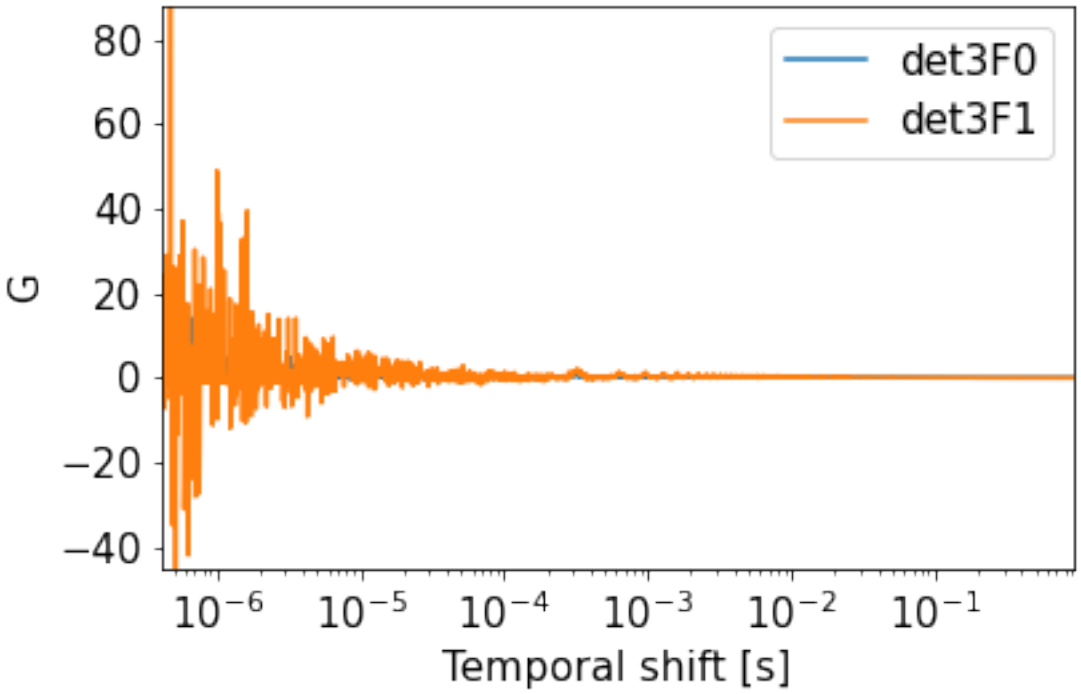
```
    h = plotFCScorrelations(G,␣
↪['det'+str(i)+'F0_average','det'+str(i)+'F1_average'], limits=[start, stop],␣
↪pColors=[0, 1, 2])
```

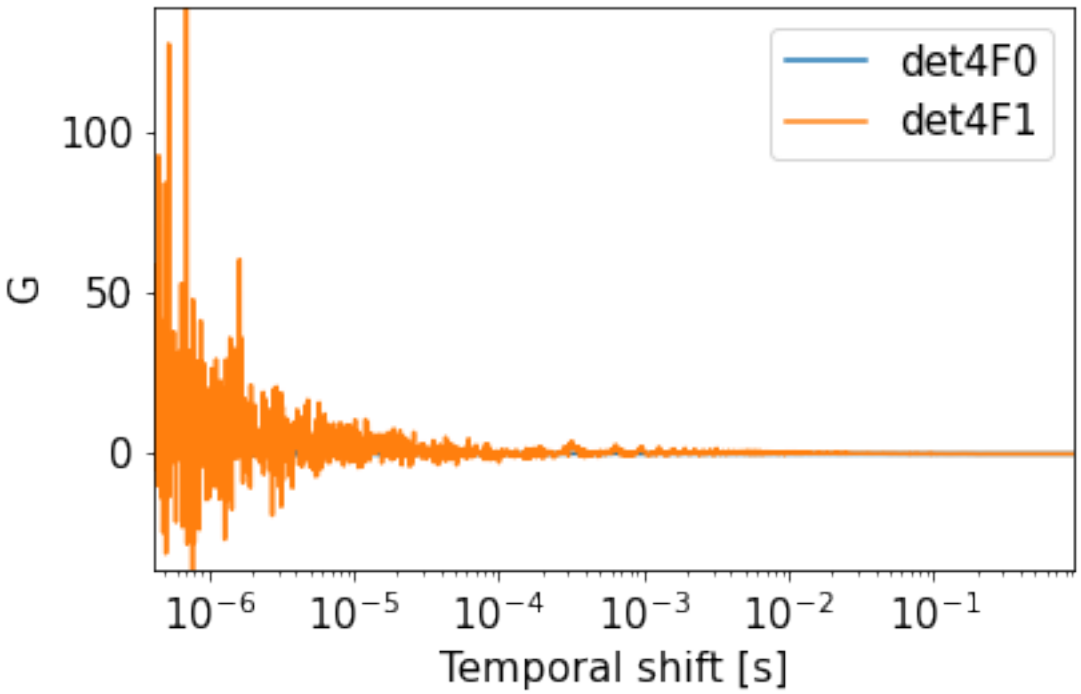
```
/home/labuser/myDev/timetaggingplatform/dataProcessing/libs/spad_ffs/spad_fcs/FC
S2Corr.py:568: RuntimeWarning: More than 20 figures have been opened. Figures
created through the pyplot interface (`matplotlib.pyplot.figure`) are retained
until explicitly closed and may consume too much memory. (To control this
warning, see the rcParam `figure.max_open_warning`).
  h = plt.figure()
```

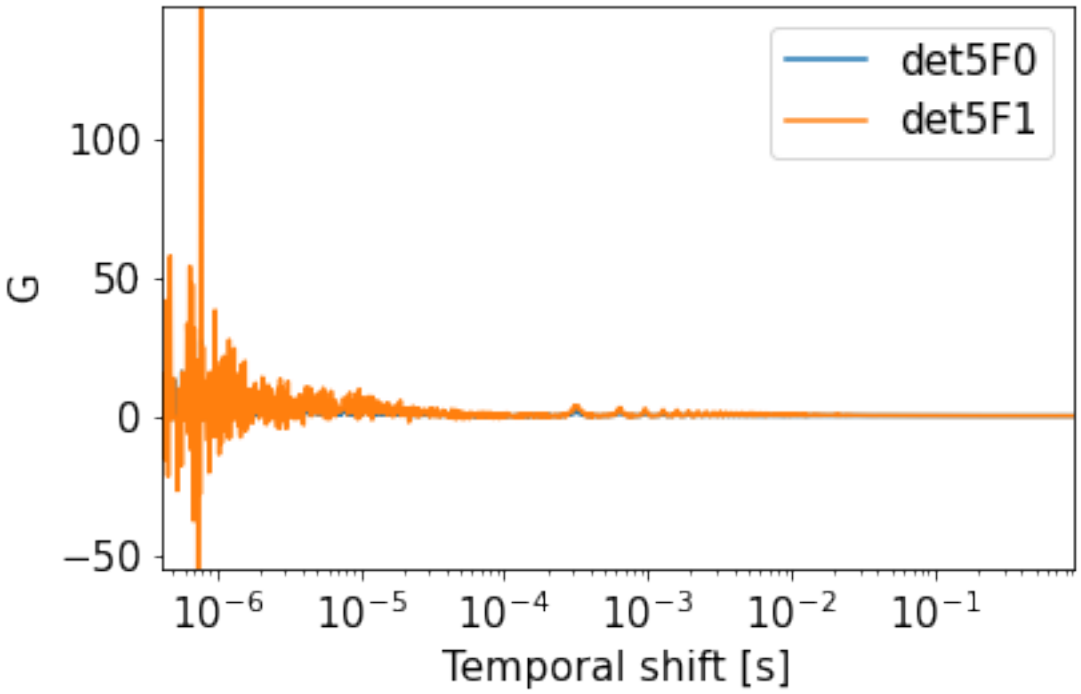


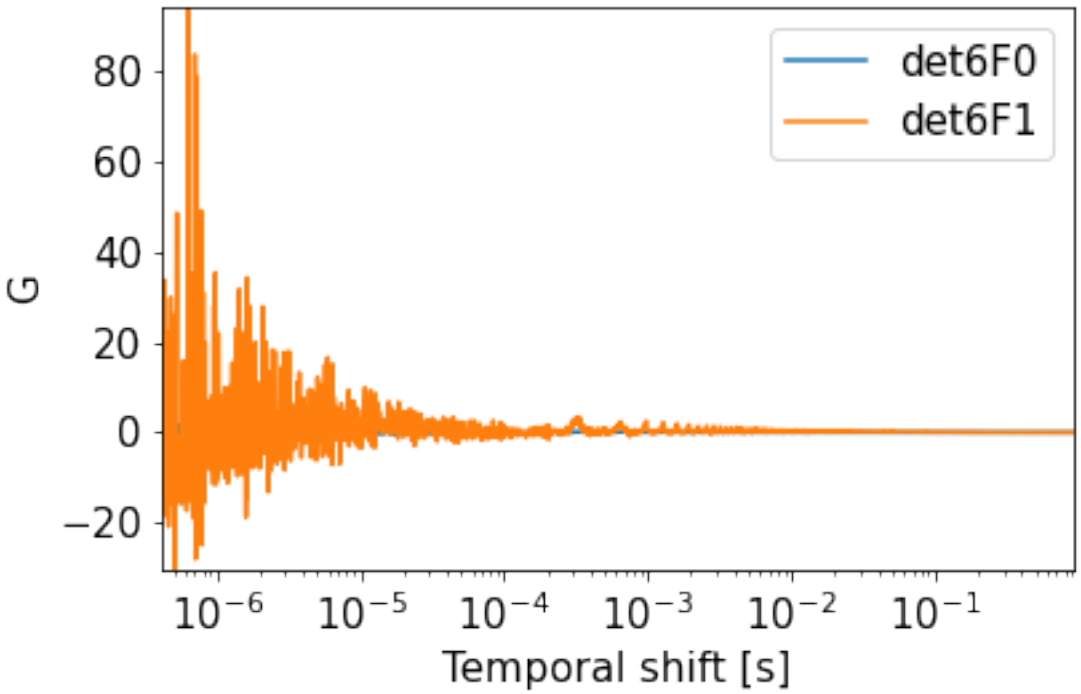

det1F0
det1F1
G
60
40
20
0
−20
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
Temporal shift [s]

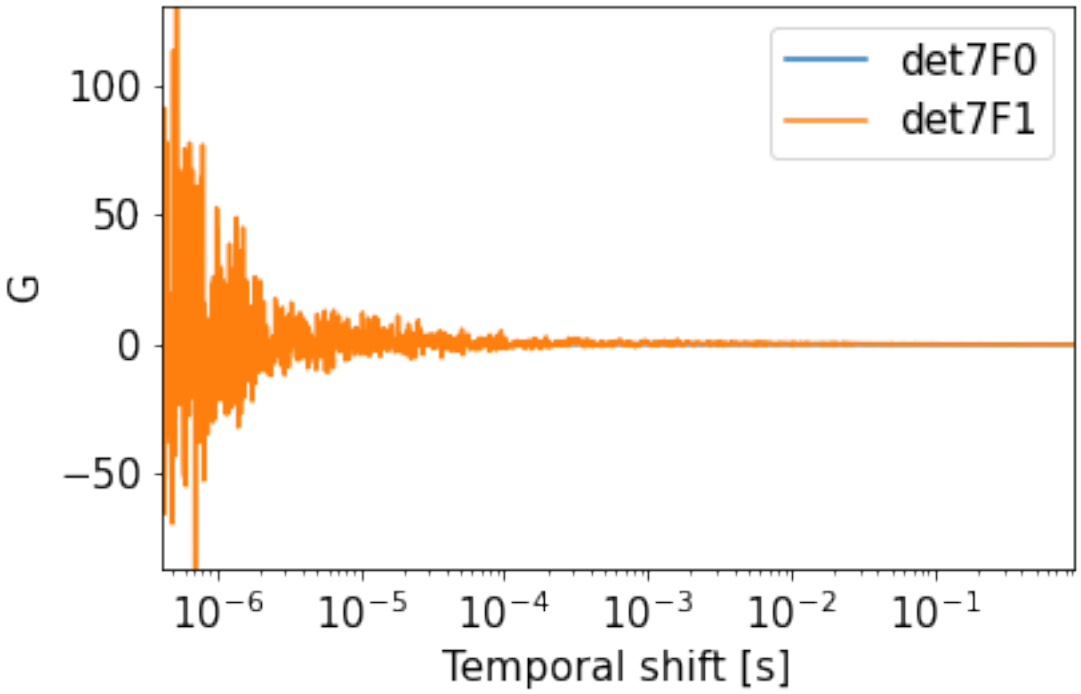

det2F0
det2F1
G
60
40
20
0
−20
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
Temporal shift [s]

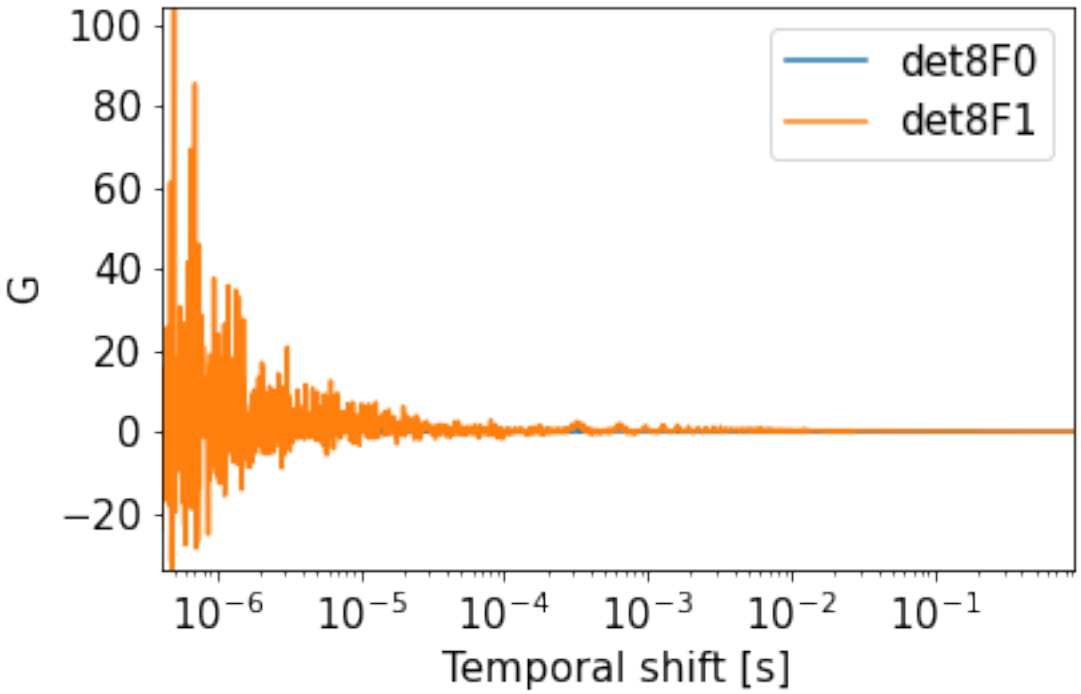

det3F0
det3F1
80
60
40
20
0
−20
−40
G
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
Temporal shift [s]

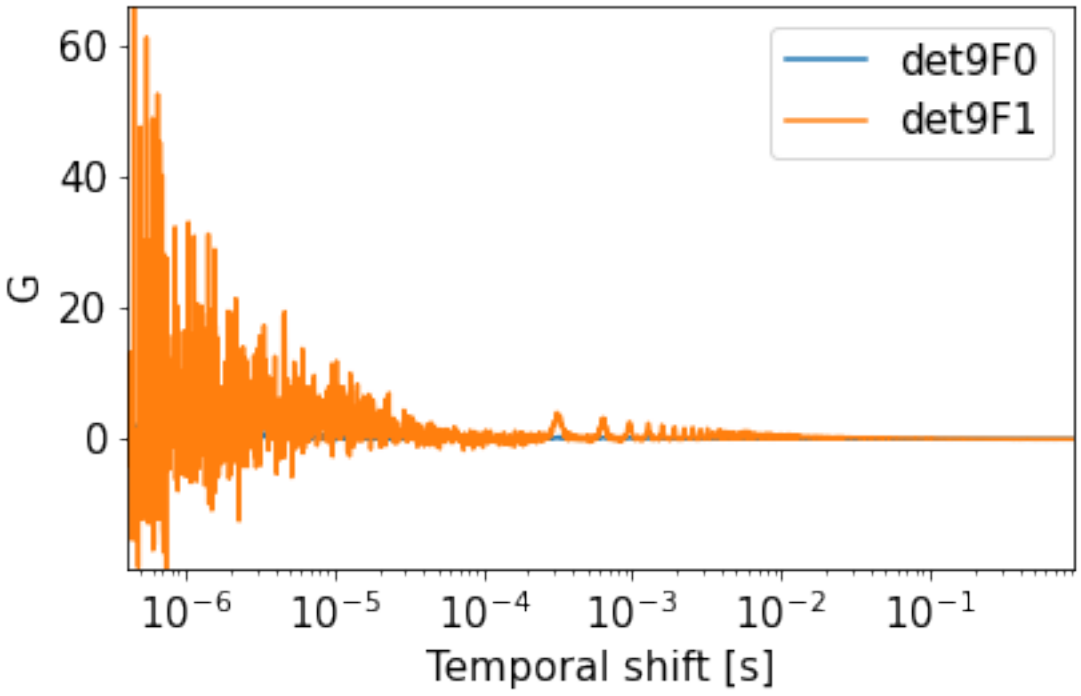

det4F0
det4F1
100
50
0
G
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
Temporal shift [s]

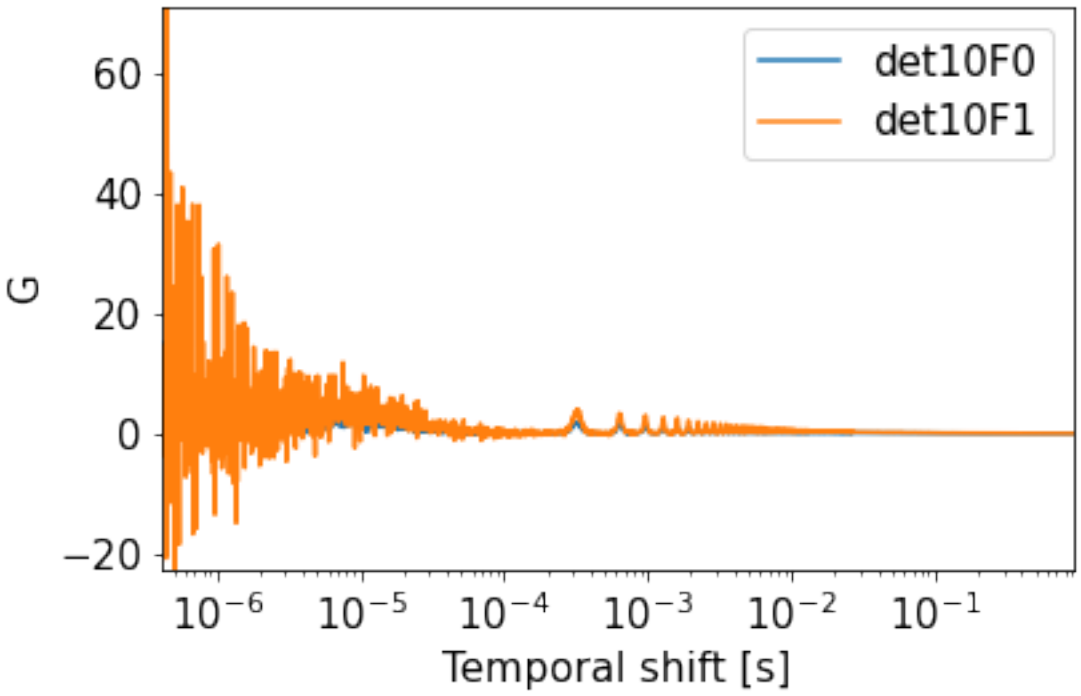

det5F0
det5F1
100
50
0
−50
G
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
Temporal shift [s]

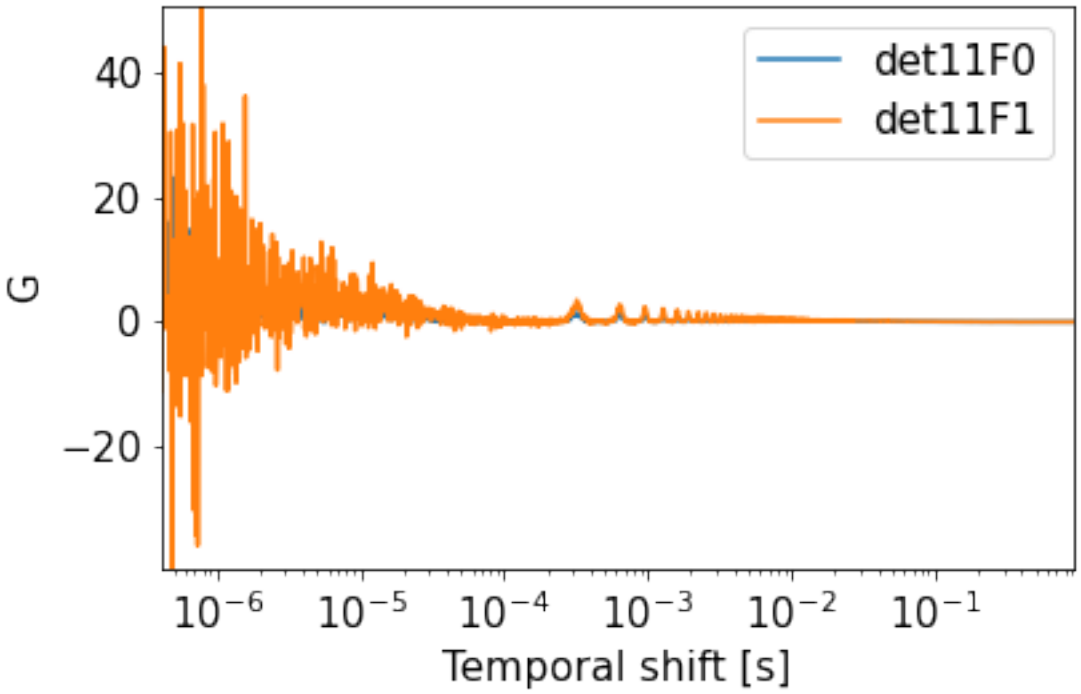

det6F0
det6F1
80
60
40
20
0
−20
G
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
Temporal shift [s]

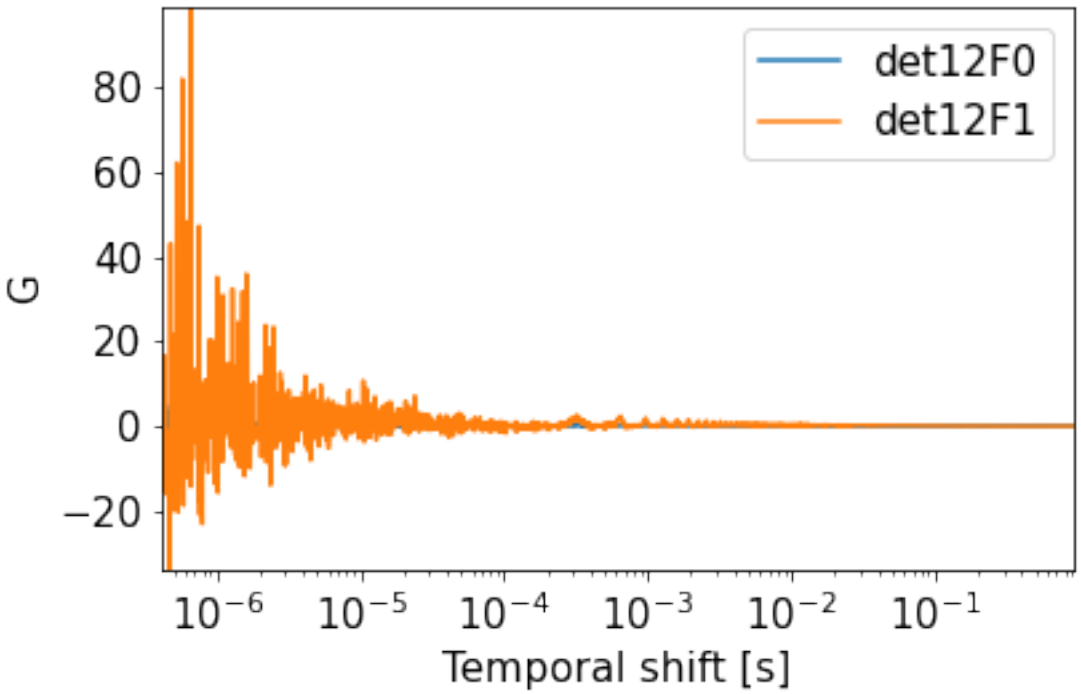

det7F0
det7F1
100
50
0
−50
G
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
Temporal shift [s]

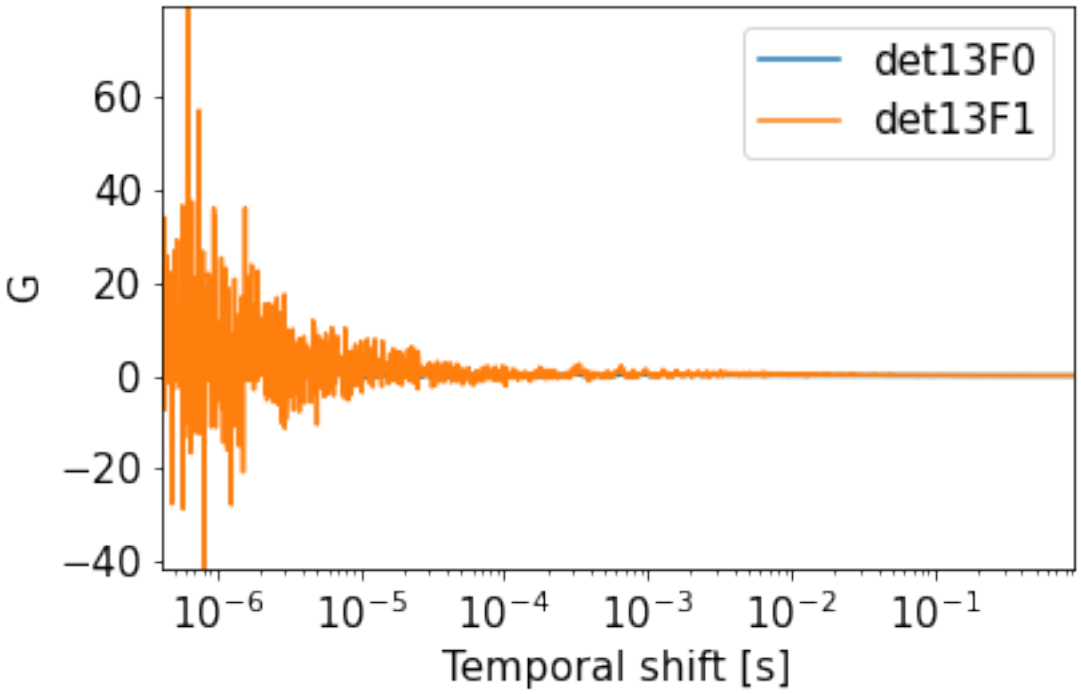

det8F0
det8F1
100
80
60
40
20
0
−20
G
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
Temporal shift [s]

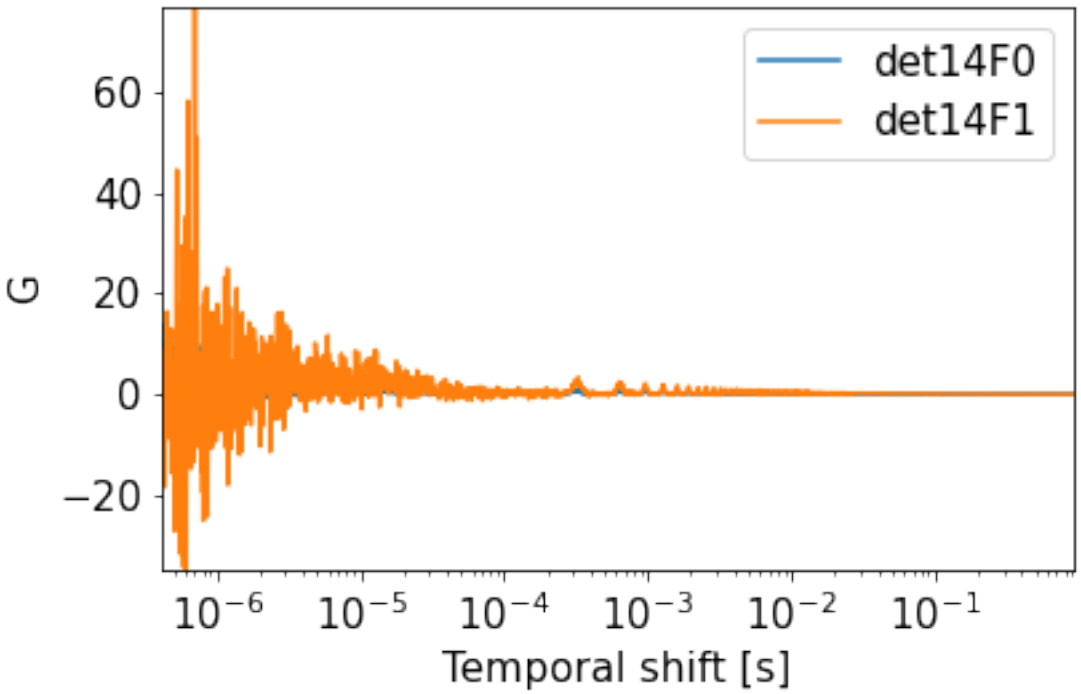

det9F0
det9F1
60
40
20
0
G
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
Temporal shift [s]

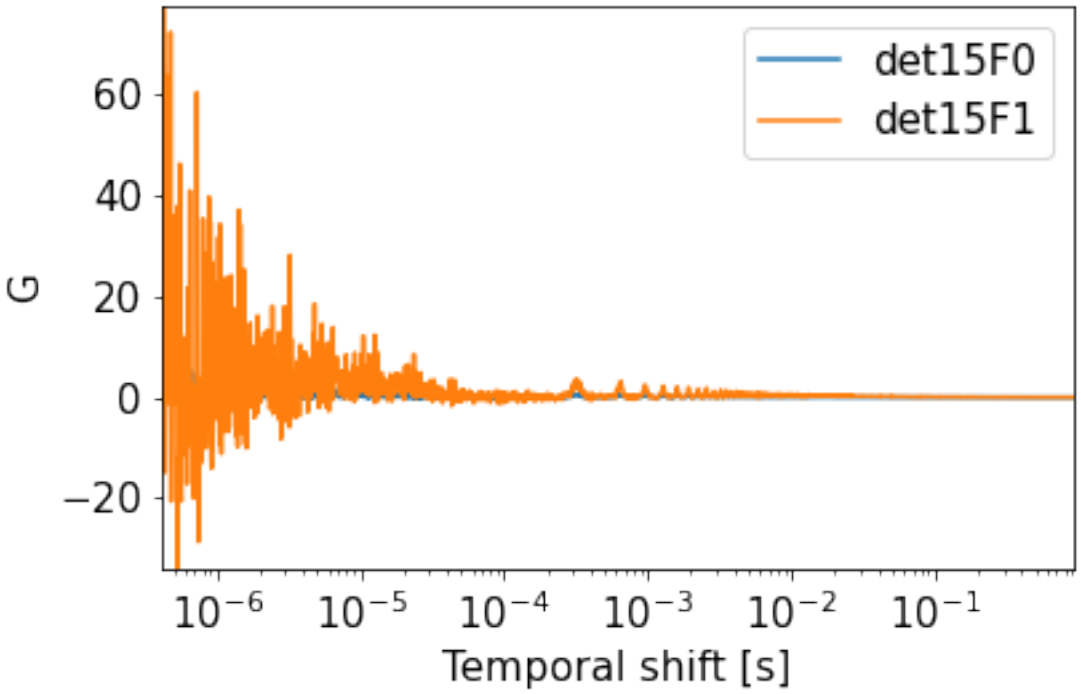

det10F0
det10F1
60
40
20
0
−20
G
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
Temporal shift [s]

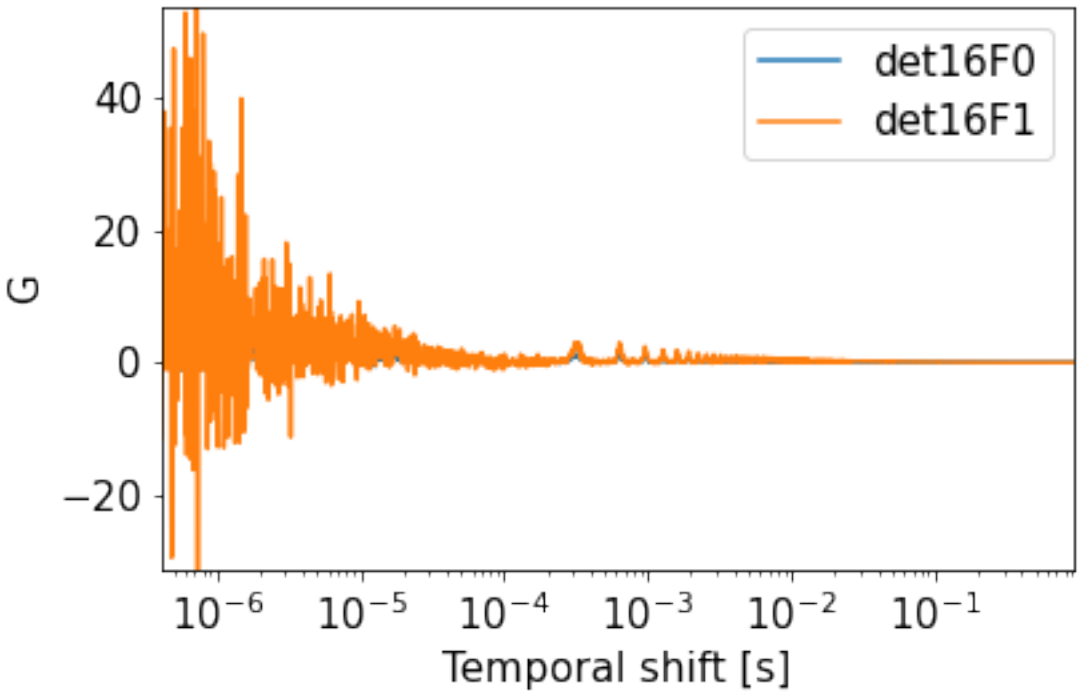

det11F0
det11F1
40
20
0
−20
G
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
Temporal shift [s]

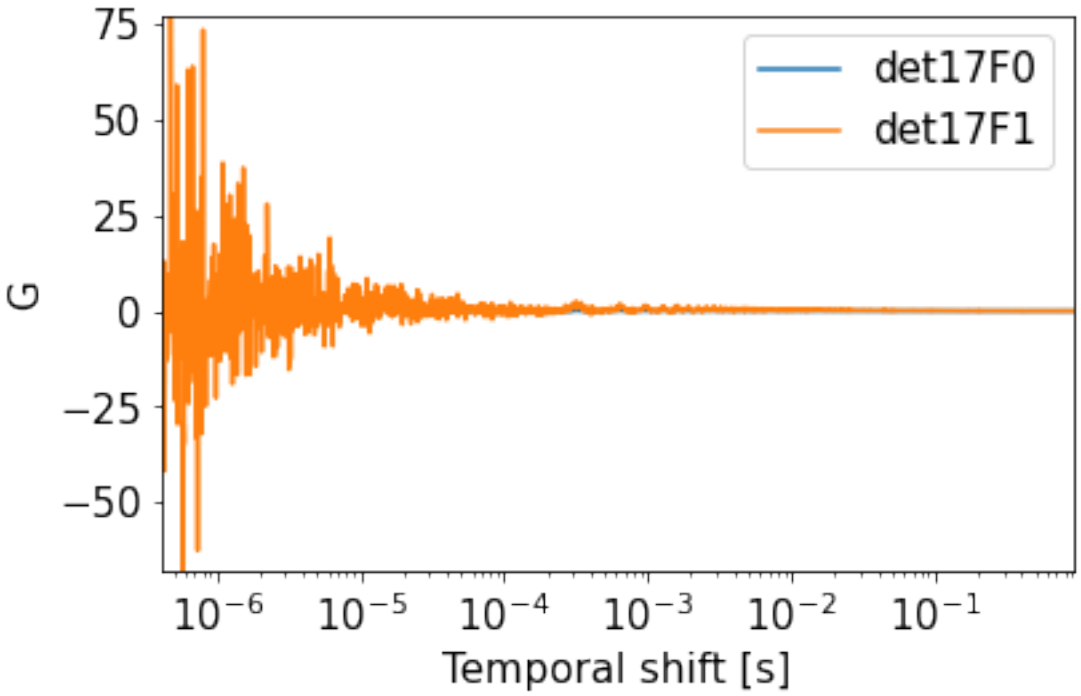

det12F0
det12F1
80
60
40
20
0
−20
G
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
Temporal shift [s]

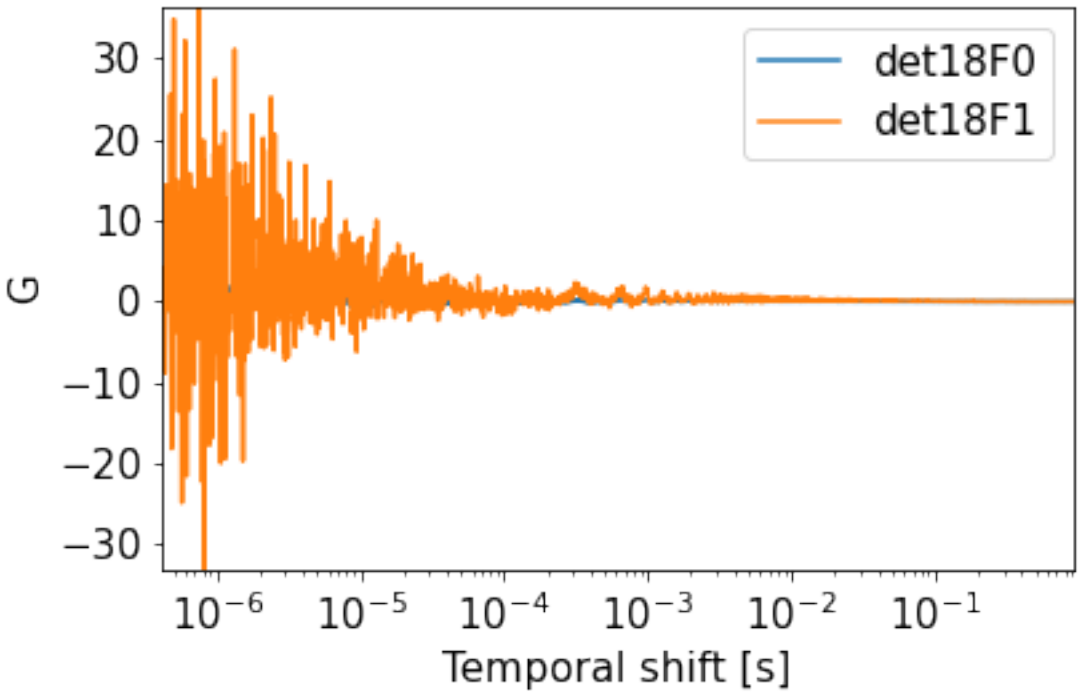

det13F0
det13F1
60
40
20
0
−20
−40
G
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
Temporal shift [s]

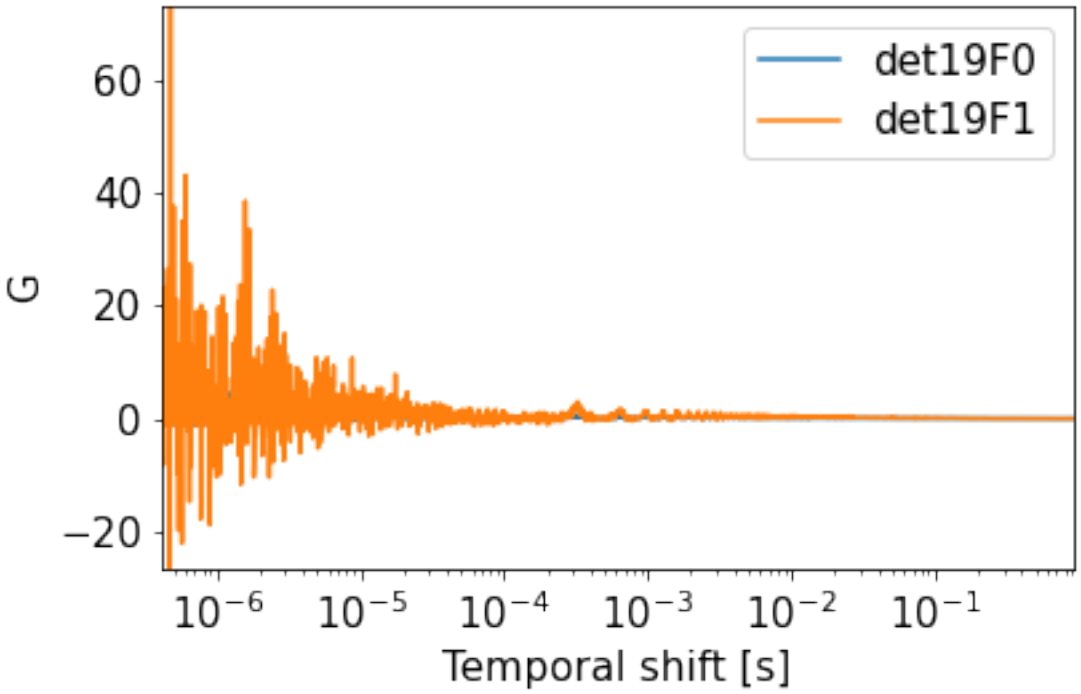

det14F0
det14F1
60
40
20
0
−20
G
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
Temporal shift [s]

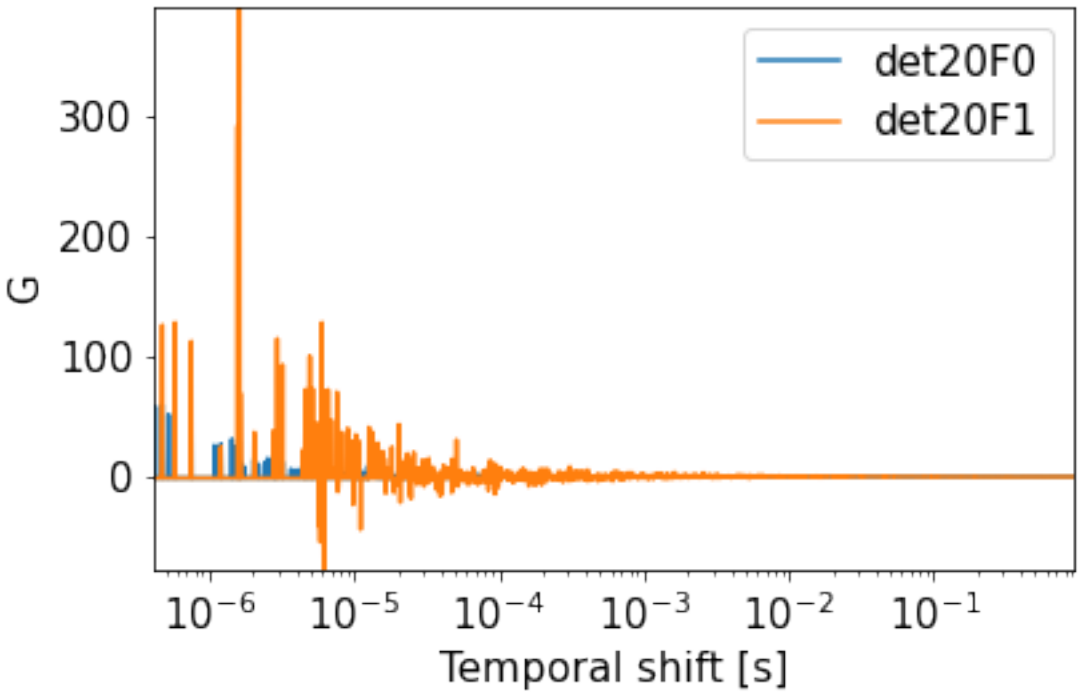

det15F0
det15F1
60
40
20
0
−20
G
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
Temporal shift [s]


det16F0
det16F1
40
20
0
−20
G
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
Temporal shift [s]


det17F0
det17F1
G
Temporal shift [s]


det18F0
det18F1
G
Temporal shift [s]

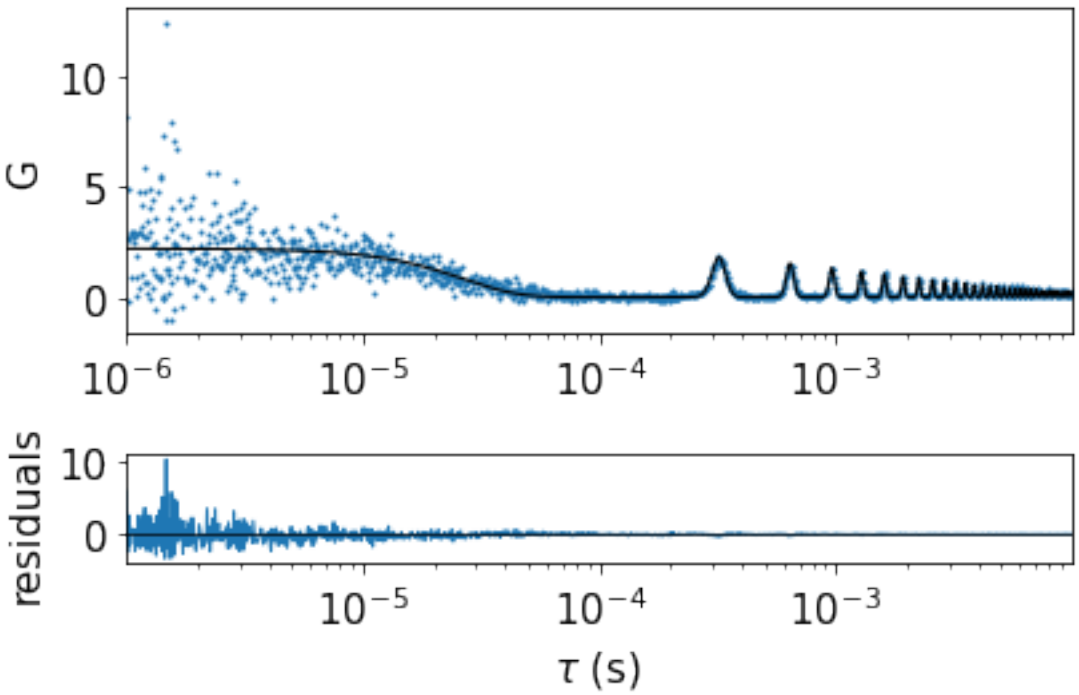

det19F0
det19F1
60
40
20
0
−20
G
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
Temporal shift [s]


det20F0
det20F1
300
200
100
0
G
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
Temporal shift [s]

1.8 Fit correlations unfiltered data

[34]:
```
SF = 4.5 # shape parameter for the PSF
amp = 1 # start value for the amplitude of the correlation function
w0 = 220e-9 # start value for the beam waist
Gfit = "det10F0_average"

fitresults = np.zeros((1, 3))
color = "C0"
color = 0

dataSingle = getattr(G, Gfit)
dataSingleF = getattr(G, Gfit)
tau = dataSingle[:,0]
Gexp = dataSingle[:,1]
GexpF = dataSingleF[:,1]

[dummy, start] = findNearest(tau,1e-6)
[dummy, stop] = findNearest(tau,1e-2)

fitarray = np.array([1, 1, 1, 0, 0, 0, 0])
paramStart = np.array([amp, 2.4, 300e-9, SF, 0.5e-6, 320e-6, 0])
paramMin = np.array([0, 5e-6, 100e-9, 3, 0, 0, -1])
paramMax = np.array([1e6, 10, 10000e-9, 6, 10, 10, 1])
                            #  [N, tauD, w, SF, Rcirc, Tcirc, offset]
fitresult = FCSfit(Gexp[start:stop],  tau[start:stop], 'fitfunCircFCS',␣
 ↪fitarray, paramStart,  paramMin, paramMax, color, 0, savefig=Gfit+'.svg',␣
 ↪plotTau=False)
for l in range(sum(fitarray)):
    print("fitresult[" + str(l) + "] = " + str(fitresult.x[l]))
D = fitresult.x[2]**2 / 4 / (fitresult.x[1]*1e-3)
print(D)
print("diameter: " + str(StokesEinstein(D)))
color += 1
fitresults[0, :] = fitresult.x
```

```
------
0.009932111871999995
saving figure
tauD = 1.5304485396306164 ms
chi2 = 78.91191552823342
fitresult[0] = 0.45407320819597397
fitresult[1] = 1.5304485396306164
fitresult[2] = 2.6116775067284766e-07
1.1141928693658869e-11
diameter: 3.852292453303357e-08
```


G
10
5
0
10^{-6}
10^{-5}
10^{-4}
10^{-3}
residuals
10
0
10^{-5}
10^{-4}
10^{-3}
τ (s)

[ ]:

[ ]:

[ ]: