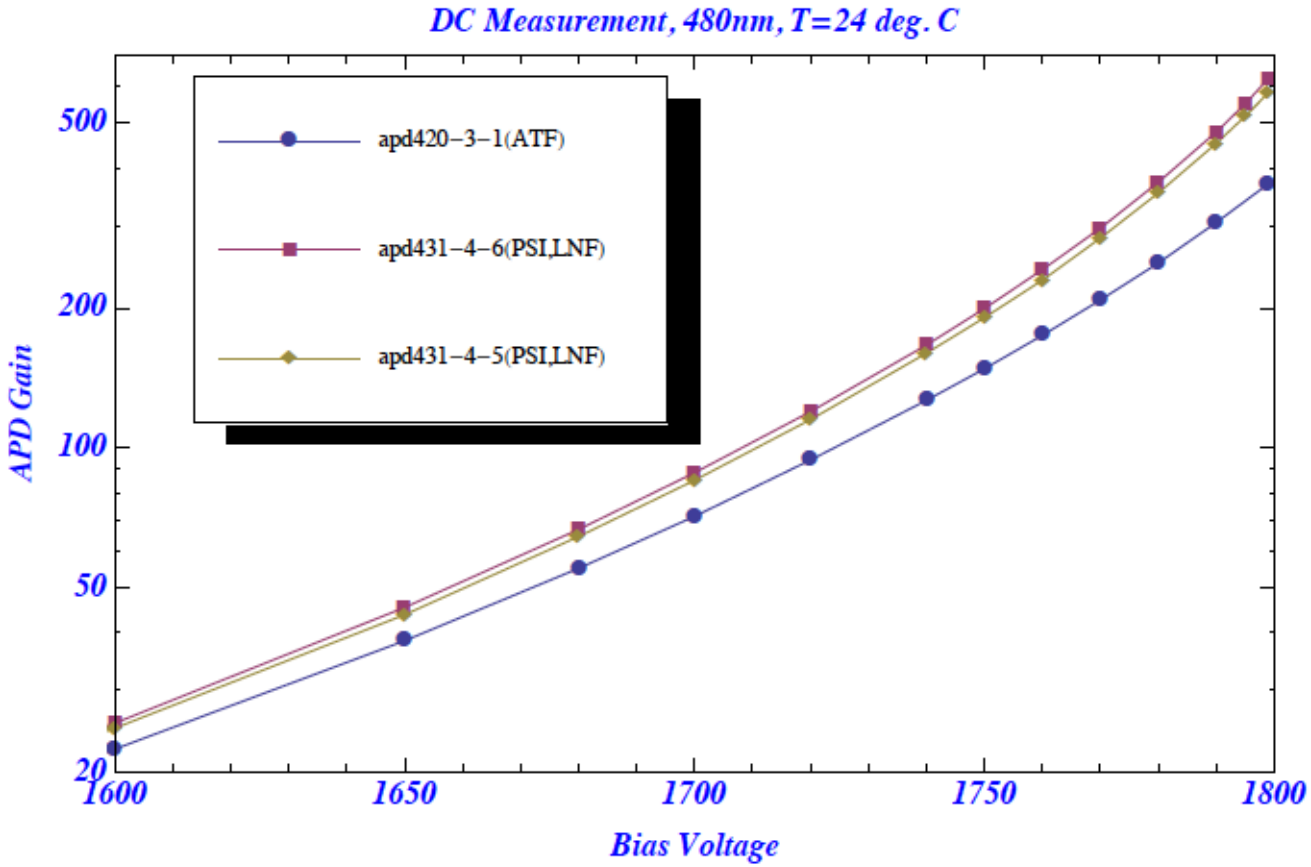
Analysis of DESY testbem data starting from 16:45 local time on March 1.
Scope inputs are:
1)front APD- 200 mV/div,
2)front APD- 10 mV/div,
3) back APD -200 mV/div
4) back APD -10 mV/div
trigger is 4mm**2 APD(trigger) with mean amplitude~600 mV and threshold of 40 mV.
Preamps are Cividec(2 GHz, 40 dB) for channels 1 and 2 and Wenteq (1GHz,50dB) for 3.
For all data the APD bias voltage was 1795 volts.(most likely gain ~520)
I noted on the morning of the 2nd that (one of) the APDs was starting to draw current and that , as a result, the effective HV on all detectors was reduced.
Tracing this down in the data themselves I find:
-runs 32-37 are all ok with 10k events/file
-run 38 is a short run (skip it)
-runs 39 and 40 seem also good with 10k events
-but 41 and 42 (and possibly end of 40) look like gain has dropped.
=>So estimate ~80k mip event to work with.

SNW-Oct. 29th, 2014

First calculate expected signal amplitude for min. ionizing particle in the 8 x8 mm^2 APDs (area enters because this has large capacitance leading to a loss in peak amplitude into a 50 Ω voltage amplifier - by a factor of ~5 in simulation).



Uncertainties on expected pulse amplitude :

70 e - h/micron is taken from paper of Meroli et al. 40 Micron is given by Dick Farrell as effective thickness of high field region. From the above Gain curve, the expected gain at 1795 V is in range ~320 to 520 but private communication from Dick Farrel says one device (464-6-1) would correspond to higher gain curve above.

In[190]:=
```
ehpairs = 70; thick = 40; G_APD = 520.; t_pulse = 5 * 10^-9; Qe = 1.6 * 10^-19; R = 50.;
Q = ehpairs * thick * G_APD * Qe;
i_peak = Q / (t_pulse / 2);
v_peak = i_peak * R
```

Out[193]= 0.0046592

In[194]:=
```
dbamp = 20 Log[10, 100];
gamp_Cividec = 10^(40/20)
gamp_wenteq = N[10^(50/20)]
```

Out[195]= 100

Out[196]= 316.228

Conclusion : We should expect an input signal with peak amplitude of about 4.5 mV into the preamp and should take into account a factor of about ~5 reduction for the capacitance of the 64mm^2 detectors..
Inspection of the DESY data below shows a peak amplitude of about 600 mV, roughly as expected for the low capacitance 2*2mm^2 detector (using the Wenteq spec sheet w. 50 dB nominal gain we would actually expect ~1.5V).
The data for the 64 mm^2 detectors have a most probable p.h. of about 120 mV.
Since Wenteq gain is about 3 times Cividec the pulse height difference is not as significant as expected (5:1 instead of 3:1).

The 64 mm^2 detector has lower amplitude(but unlikely more than *a* factor of 2 reduction) &

slower risetime than the 4 mm^2 detector.

This is in qualitative agreement with the circuit simulation we did.

It also means that, with a 2 nsec risetime signal the preamp noise at the output, in order to have less than 20 picosecond contribution to the time jitter, should be less than 4 mV- ie SNR=100:1.

"iorder" specifies level of filtering. ie iorder=5 reduces 5 GHz to 1 GHz average of 5 pts. Expect noise reduction from sampling to be 1/sqrt(5).

Start by looking at dataset with trigger apd in chan3 to get full ph distributions.

In[77]:=

4 channels of LRS 1 GHz scope with 5 GHz sampling. Ch1= front, Ch2=back 64mm**2 APD. Ch3= 4 mm**2 APD, triggers scope at 40 mV.

In[78]:=
```
SetDirectory["~bastian/Desktop/desydata/"];
Namelist = FileNames[];
Namelist // Length;
nfiles = %;
iorder = 1;
```

In[83]:=
```
filename = Namelist[[6]]
scopedata = Import[filename, "csv"];
Dimensions[scopedata]
```

Out[83]= LeCroy-RTW-2014-03-01-002.csv

Out[85]= {8000, 255}

In[86]:=
```
t0 = scopedata[[1, 2]];
dt = scopedata[[1, 3]] * 10^9
time = Range[0, 251] * dt;
```

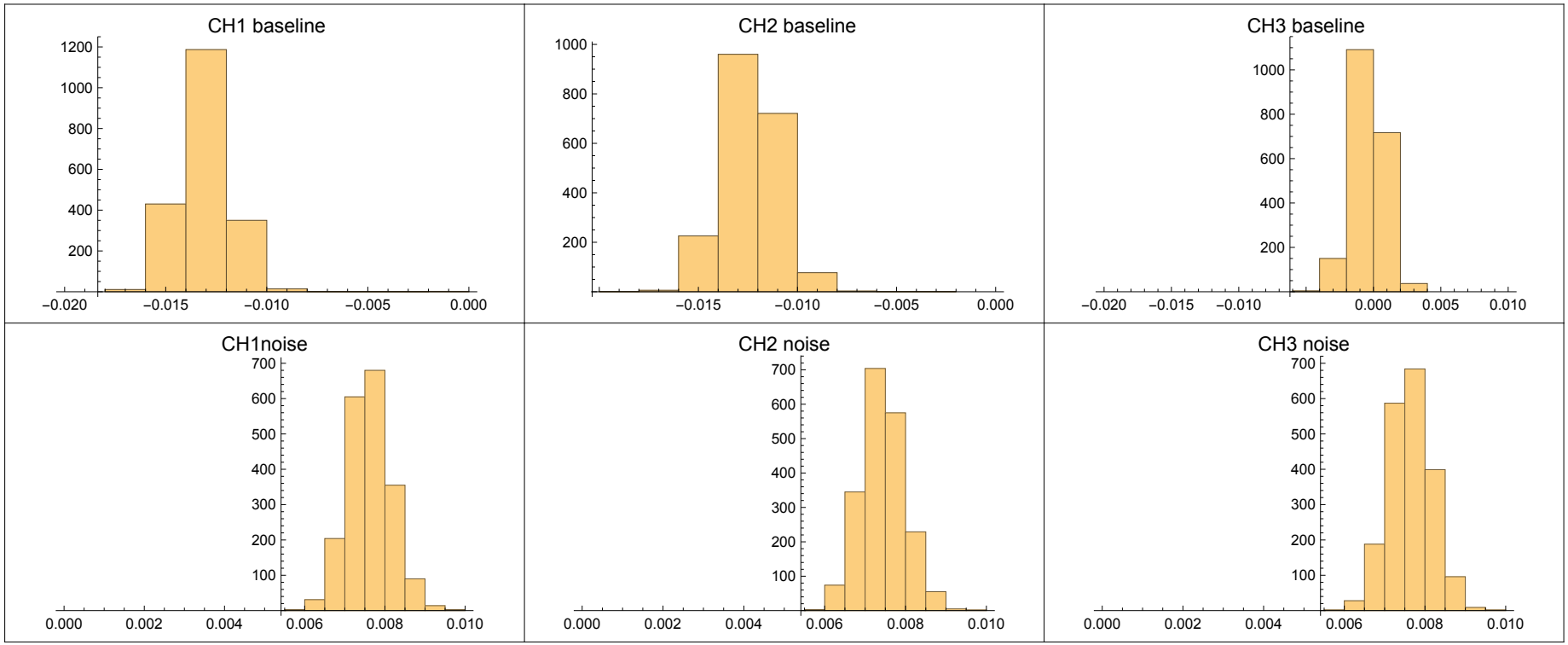
Out[87]= 0.2

In[89]:=
```
v1 = Table[Take[scopedata[[4 * (i - 1) + 1]], {4, 255}], {i, 2000}];
v2 = Table[Take[scopedata[[4 * (i - 1) + 2]], {4, 255}], {i, 2000}];
v3 = Table[Take[scopedata[[4 * (i - 1) + 3]], {4, 255}], {i, 2000}];
```

In[92]:=
```
inbase1 = Table[Mean[Take[v1[[i]], 120]], {i, 2000}];
innoise1 = Table[RootMeanSquare[Take[v1[[i]], 120] - inbase1[[i]]], {i, 2000}];
inbase2 = Table[Mean[Take[v2[[i]], 120]], {i, 2000}];
innoise2 = Table[RootMeanSquare[Take[v2[[i]], 120] - inbase2[[i]]], {i, 2000}];
inbase3 = Table[Mean[Take[v3[[i]], 120]], {i, 2000}];
innoise3 = Table[RootMeanSquare[Take[v3[[i]], 120] - inbase3[[i]]], {i, 2000}];
restt1 = Table[(Take[v1[[i]], {116, 165}] - inbase1[[i]]), {i, 2000}];
restt2 = Table[(Take[v2[[i]], {116, 165}] - inbase2[[i]]), {i, 2000}];
restt3 = Table[-(Take[v3[[i]], {116, 165}] - inbase3[[i]]), {i, 2000}];
```

In[101]:=
```
GraphicsGrid[{{Histogram[inbase1, {-.02, 0, .002}, PlotLabel → "CH1 baseline"], Histogram[inbase2, {-.02, 0, .002},
    PlotLabel → "CH2 baseline"], Histogram[inbase3, {-.02, 0.01, .002}, PlotLabel → "CH3 baseline"]},
  {Histogram[innoise1, {0., 0.01, .0005}, PlotLabel → "CH1noise"], Histogram[innoise2, {0., .01, .0005},
    PlotLabel → "CH2 noise"], Histogram[innoise3, {0., 0.01, .0005}, PlotLabel → "CH3 noise"]}}, Frame → All, ImageSize → Full]
```

Out[101]=

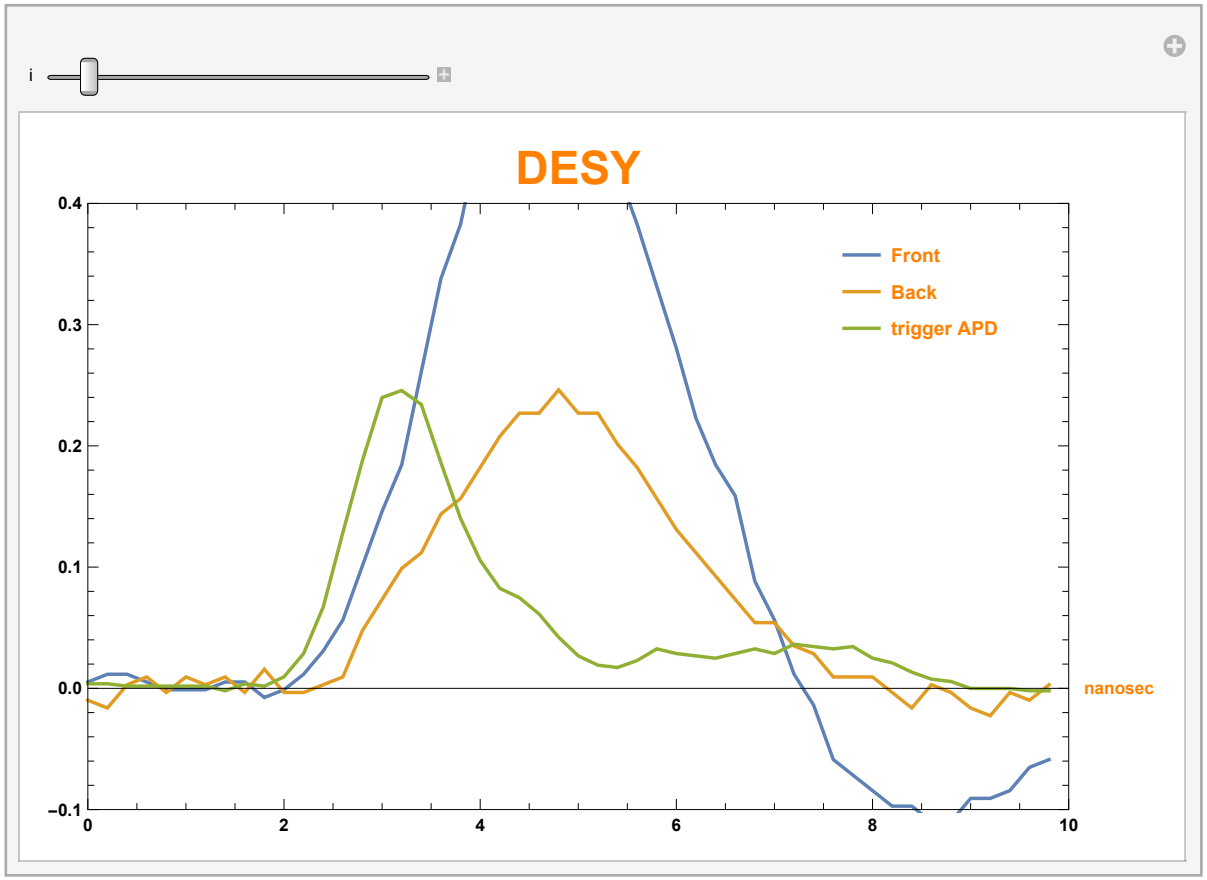


In[102]:=
```
Print[Style[{" noise1= ", Mean[innoise1], " noise2= ", Mean[innoise2], " noise3= ", Mean[innoise3], " volts"}, Large, Red]]
```

{ noise1= , 0.00781447, noise2= ,
 0.00764281, noise3= , 0.00781248, volts}

In[103]:=
```
timerest = Range[0, 49] * dt;
Manipulate[
 ListPlot[{Transpose[{timerest, restt1[[i]]}], Transpose[{timerest, restt2[[i]]}], Transpose[{timerest, 0.3 * restt3[[i]]}]},
  PlotStyle → Directive[AbsoluteThickness[1.8]], LabelStyle → Directive[Orange, Bold],
  PlotLabel → Style[DESY, Large], AxesLabel → {nanosec, V}, PlotStyle → {Red, Blue, Green},
  PlotLegends → Placed[{"Front", "Back", "trigger APD"}, {0.85, 0.85}], PlotRange → {{0, 10.}, {-.1, .4}},
  Joined → True, ImageSize → Large, Frame → True, FrameStyle → Black], {i, 1, 200, 1}, SaveDefinitions → True]
```

Out[104]=

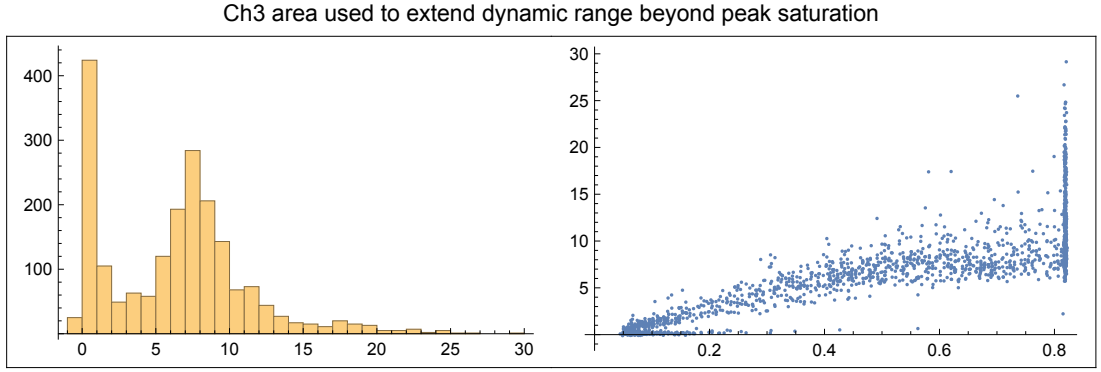


Now look at peak amplitude on all 3 APDs expecting to see a Landau distribution.

In[105]:=
```
peak1 = Table[Max[restt1[[i]]], {i, 2000}]; peak2 = Table[Max[restt2[[i]]], {i, 2000}];
peak3 = Table[Max[restt3[[i]]], {i, 2000}];
area3 = Table[Sum[restt3[[i, j]], {j, 1, 50}], {i, 2000}];
```

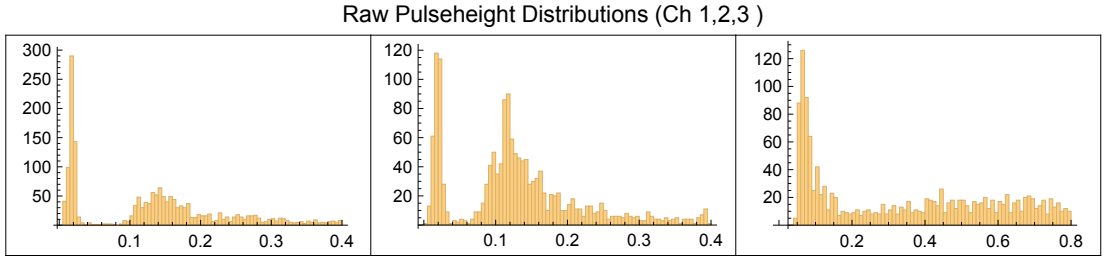
In[107]:=
```
GraphicsRow[{Histogram[area3],
  ListPlot[Transpose[{peak3, area3}]]}, Frame → True,
 PlotLabel → "Ch3 area used to extend dynamic range beyond peak saturation", ImageSize → Large]
```

Out[107]=



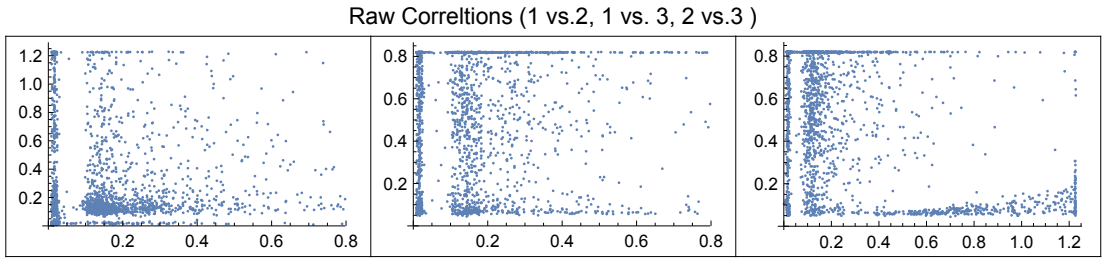
In[108]:=
```
GraphicsRow[{Histogram[peak1, {0, .4, .005}], Histogram[peak2, {0, 0.4, .005}], Histogram[peak3, {0, .8, .01}]},
 Frame → All, PlotLabel → "Raw Pulseheight Distributions (Ch 1,2,3 )", ImageSize → Large]
```

Out[108]=



In[109]:=
```
GraphicsRow[{ListPlot[Transpose[{peak1, peak2}]], ListPlot[Transpose[{peak1, peak3}]], ListPlot[Transpose[{peak2, peak3}]]},
 Frame → All, PlotLabel → "Raw Correltions (1 vs.2, 1 vs. 3, 2 vs.3 )", ImageSize → Large]
```

Out[109]=

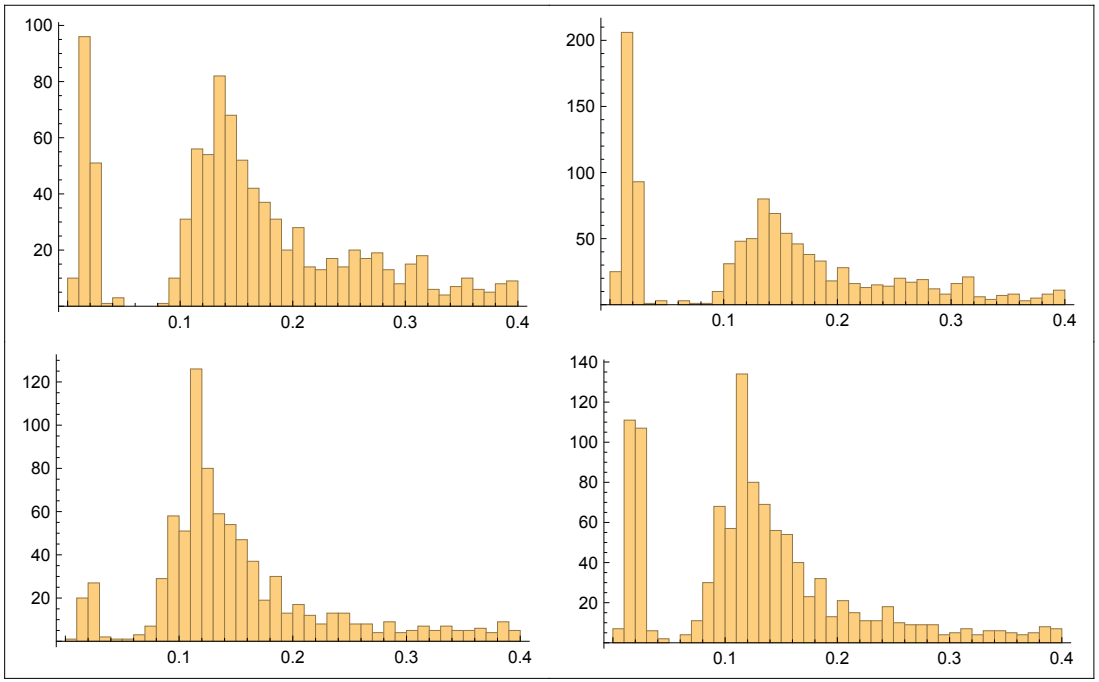


Now create an nTuple with the 3 peak values for each event.

In[110]:=
```
peakntup = Transpose[{peak1, peak2, peak3, area3, restt1, restt2, restt3}];
```

In[111]:=
```
GraphicsGrid[{{Histogram[Select[peakntup, (#[[4]] > 5 && #[[2]] > 0.06) &][[All, 1]], {0, .4, .01}],
   Histogram[Select[peakntup, (#[[4]] > 5 && #[[4]] < 15) &][[All, 1]], {0, .4, .01}]},
  {Histogram[Select[peakntup, (#[[4]] > 5 && #[[1]] > 0.06) &][[All, 2]], {0, .4, .01}],
   Histogram[Select[peakntup, (#[[4]] > 5 && #[[4]] < 15) &][[All, 2]], {0, .4, .01}]}}, Frame → True,
 PlotLabel → "Ch1 with min cut on 2&3 and only cut on 3, then same for ch2", ImageSize → Large]
```

Out[111]=



Conclusion : Front and Back APD have most probable p.h. of about 120 mV and 110 mV. Small trigger APD has peak at ~600 mV but not well defined.

Below is for the data where large APDs were recorded in "dual-range", but trigger APD not recorded.

In[112]:=
```
SetDirectory["~bastian/Desktop/data_analysis_Mathematica/LabVIEW_Data/"];
Namelist = FileNames[];
Namelist // Length;
nfiles = %;
```

In[116]:=
```
filename = Namelist[[36]]
scopedata = Import[filename, "csv"];
```

Out[116]= LeCroy-RTW-2014-03-01-032.csv

In[118]:=
```
Dimensions[scopedata]
```

Out[118]= {40 000, 255}

In[119]:=
```
t0 = scopedata[[1, 2]];
dt = scopedata[[1, 3]] * 10^9
time = Range[0, 251] * dt;
```

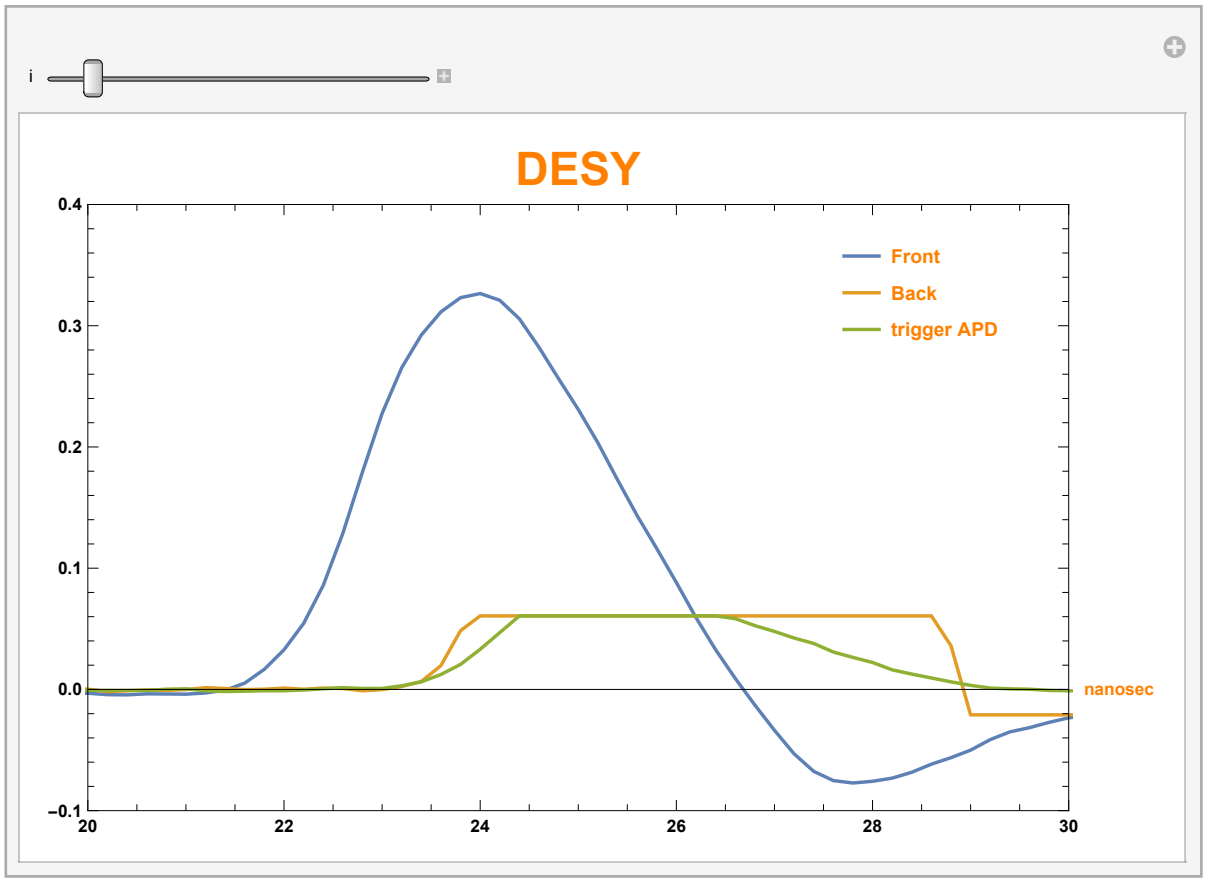
Out[120]= 0.2

In[122]:=
```
v1 = Table[Take[scopedata[[4 * (i - 1) + 1]], {4, 255}], {i, 10 000}];
v2 = Table[Take[scopedata[[4 * (i - 1) + 2]], {4, 255}], {i, 10 000}];
v3 = Table[Take[scopedata[[4 * (i - 1) + 3]], {4, 255}], {i, 10 000}];
```

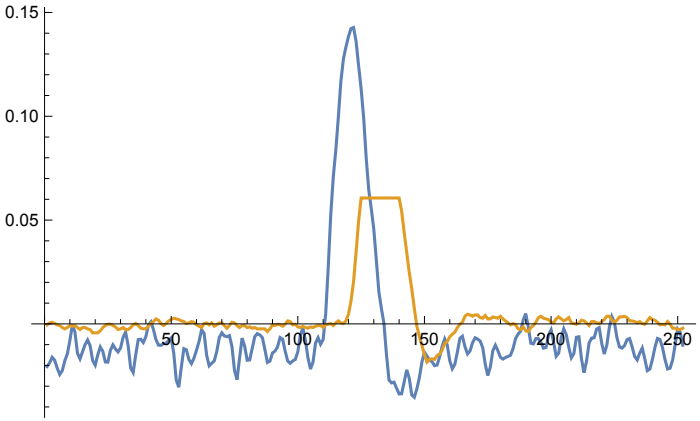
In[125]:=
```
v4 = Table[Take[scopedata[[4 * (i - 1) + 4]], {4, 255}], {i, 10 000}];
```

In[197]:=
```
Manipulate[ListPlot[{Transpose[{time, 0.3 * v1[[i]]}], Transpose[{time, v2[[i]]}], Transpose[{time, v4[[i]]}]},
  PlotStyle → Directive[AbsoluteThickness[1.8]], LabelStyle → Directive[Orange, Bold],
  PlotLabel → Style[DESY, Large], AxesLabel → {nanosec, V}, PlotStyle → {Red, Blue, Green},
  PlotLegends → Placed[{"Front", "Back", "trigger APD"}, {0.85, 0.85}], PlotRange → {{20., 30.}, {-.1, .4}},
  Joined → True, ImageSize → Large, Frame → True, FrameStyle → Black], {i, 1, 200, 1}, SaveDefinitions → True]
ListPlot[{v1[[4]], v2[[4]]}, PlotRange → Full, Joined → True]
```

Out[197]=


i
DESY
Front
Back
trigger APD
nanosec
0.4
0.3
0.2
0.1
0.0
−0.1
20
22
24
26
28
30

Out[198]=

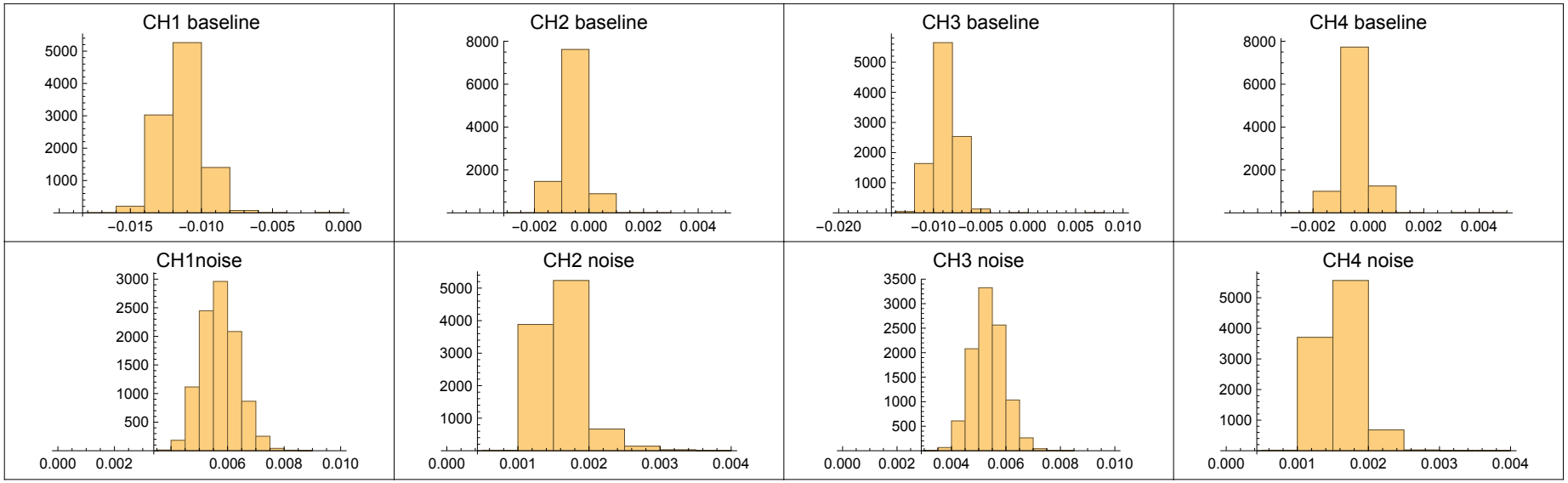


In[136]:=

```
inbase1 = Table[Mean[Take[v1[[i]], 100]], {i, 10 000}];
innoise1 = Table[RootMeanSquare[Take[v1[[i]], 100] - inbase1[[i]]], {i, 10 000}];
inbase2 = Table[Mean[Take[v2[[i]], 100]], {i, 10 000}];
innoise2 = Table[RootMeanSquare[Take[v2[[i]], 100] - inbase2[[i]]], {i, 10 000}];
inbase3 = Table[Mean[Take[v3[[i]], 100]], {i, 10 000}];
innoise3 = Table[RootMeanSquare[Take[v3[[i]], 100] - inbase3[[i]]], {i, 10 000}];
inbase4 = Table[Mean[Take[v4[[i]], 100]], {i, 10 000}];
innoise4 = Table[RootMeanSquare[Take[v4[[i]], 100] - inbase4[[i]]], {i, 10 000}];
(*rest1=Table[(Take[v1[[i]],{116,165}]-inbase1[[i]]),{i,2000}];
rest2=Table[(Take[v2[[i]],{116,165}]-inbase2[[i]]),{i,2000}];
rest3=Table[-(Take[v3[[i]],{116,165}]-inbase3[[i]]),{i,2000}];*)
```

In[144]:=
```
GraphicsGrid[{{Histogram[inbase1, {-.02, 0, .002}, PlotLabel → "CH1 baseline"],
    Histogram[inbase2, {-.005, .005, .001}, PlotLabel → "CH2 baseline"], Histogram[inbase3, {-.02, 0.01, .002},
     PlotLabel → "CH3 baseline"], Histogram[inbase4, {-.005, .005, .001}, PlotLabel → "CH4 baseline"]},
   {Histogram[innoise1, {0., 0.01, .0005}, PlotLabel → "CH1noise"], Histogram[innoise2, {0, .004, .0005},
     PlotLabel → "CH2 noise"], Histogram[innoise3, {0., 0.01, .0005}, PlotLabel → "CH3 noise"],
    Histogram[innoise4, {0, .004, .0005}, PlotLabel → "CH4 noise"]}}, Frame → All, ImageSize → Full]
```

Out[144]=



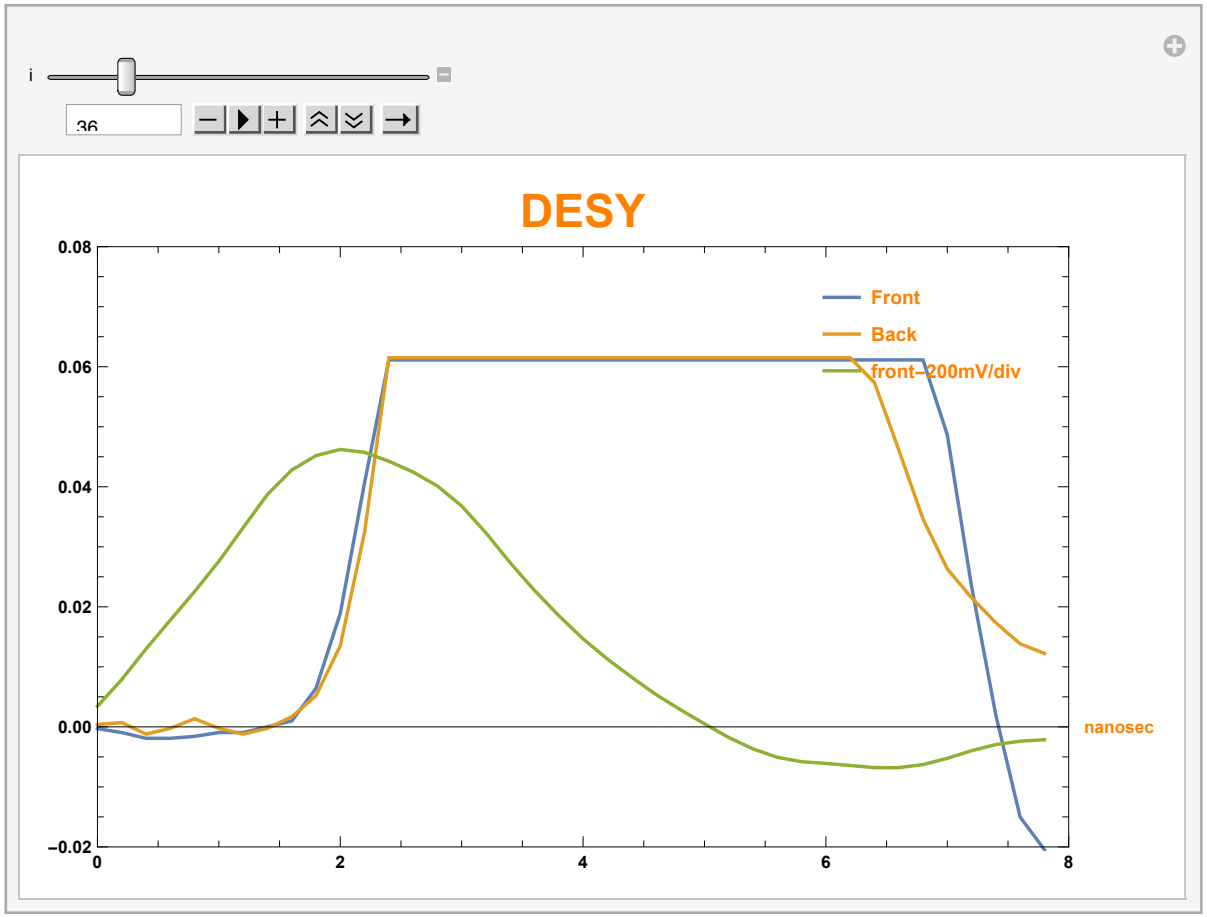
In[145]:=
```
Print[Style[{" noise2= ", Mean[innoise2], " noise4= ", Mean[innoise4], " noise3= ", Mean[innoise3], " volts"}, Large, Red]]
```

```
{ noise2= , 0.00163819, noise4= ,
 0.0016146, noise3= , 0.00539582, volts}
```

In[146]:=
```
Clear[rest1, rest2, rest3, rest4];
rest1 = Table[(Take[v1[[i]], {111, 150}] - inbase1[[i]]), {i, 10000}];
rest2 = Table[(Take[v2[[i]], {111, 150}] - inbase2[[i]]), {i, 10000}];
rest3 = Table[(Take[v3[[i]], {111, 150}] - inbase3[[i]]), {i, 10000}];
rest4 = Table[(Take[v4[[i]], {111, 150}] - inbase4[[i]]), {i, 10000}];
```

In[151]:=
```
timerestt = Range[0, 39] * dt;
Manipulate[
 ListPlot[{Transpose[{timerestt, rest2[[i]]}], Transpose[{timerestt, rest4[[i]]}], Transpose[{timerestt, 0.1 * rest1[[i]]}]},
  PlotStyle → Directive[AbsoluteThickness[1.8]], LabelStyle → Directive[Orange, Bold],
  PlotLabel → Style[DESY, Large], AxesLabel → {nanosec, V}, PlotStyle → {Red, Blue, Green},
  PlotLegends → Placed[{"Front", "Back", "front-200mV/div"}, {0.85, 0.85}], PlotRange → {{0, 8}, {-.02, .08}},
  Joined → True, ImageSize → Large, Frame → True, FrameStyle → Black], {i, 1, 200, 1}, SaveDefinitions → True]
```

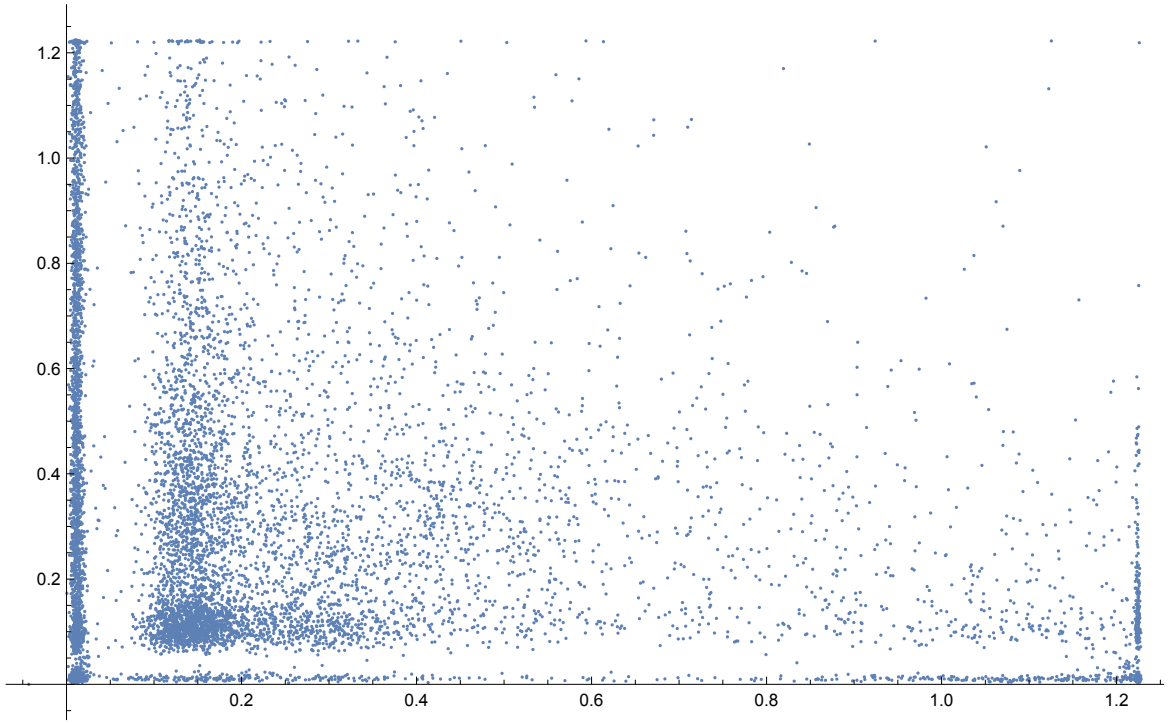
Out[152]=



In[153]:=
```
pkapd1 = Table[Max[rest1[[i]]], {i, 1, 10 000}];
pkapd3 = Table[Max[rest3[[i]]], {i, 1, 10 000}];
```

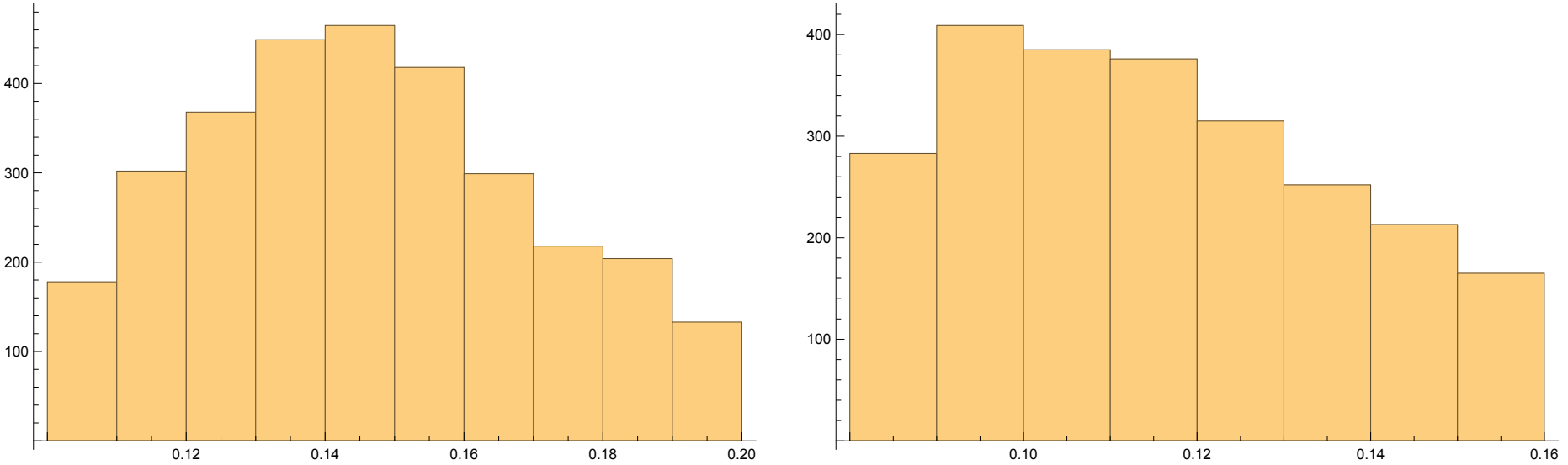
In[155]:=
```
ListPlot[Transpose[{pkapd1, pkapd3}], PlotRange → Full]
```

Out[155]=



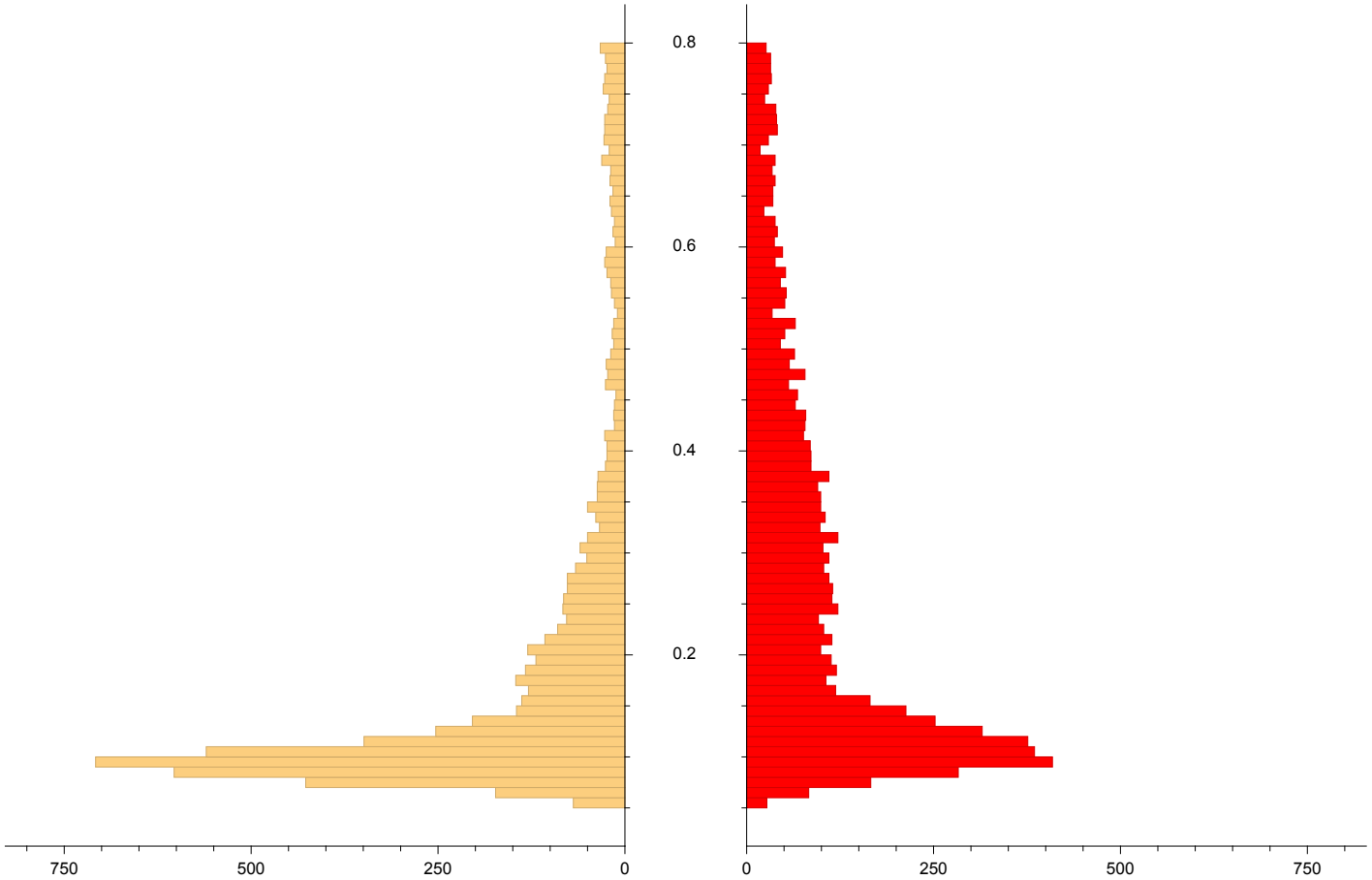
In[156]:=
```
GraphicsRow[{Histogram[pkapd1, {0.1, .2, .01}], Histogram[pkapd3, {0.08, .16, .01}]}, ImageSize → Full]
```

Out[156]=



In[157]:=
```
PairedHistogram[.66 * pkapd1, Style[pkapd3, Red], {0.05, .8, .01}, AxesOrigin → {0, 0}]
```

Out[157]=



In[158]:=
```
Clear[peakntup]
peakntup = Transpose[{pkapd1, pkapd3, rest2, rest4}];
```

Cut data to select only events with both detector signals in MIP peak

In[161]:=
```
Clear[peakcut]
peakcut = Select[peakntup, (#[[1]] > .1 && #[[1]] < .2 && #[[2]] > 0.08 && #[[2]] < 0.16) &];
```

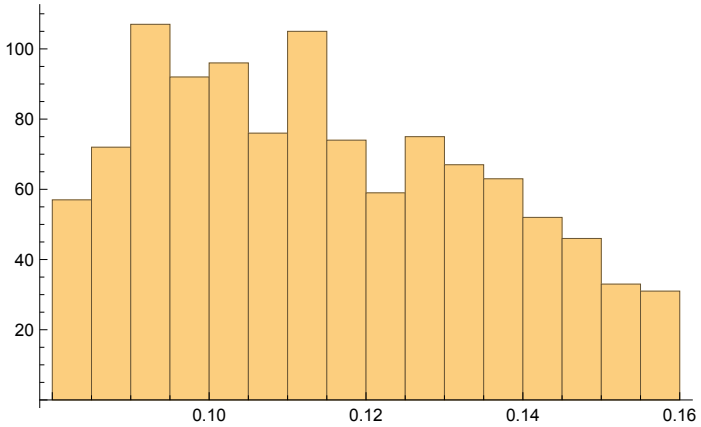
In[163]:=
```
Dimensions[peakcut]
```

Out[163]=
```
{1105, 4}
```

In[164]:=
```
Clear[p1, p3, r2, r4]
{p1, p3, r2, r4} = Transpose[peakcut];
```

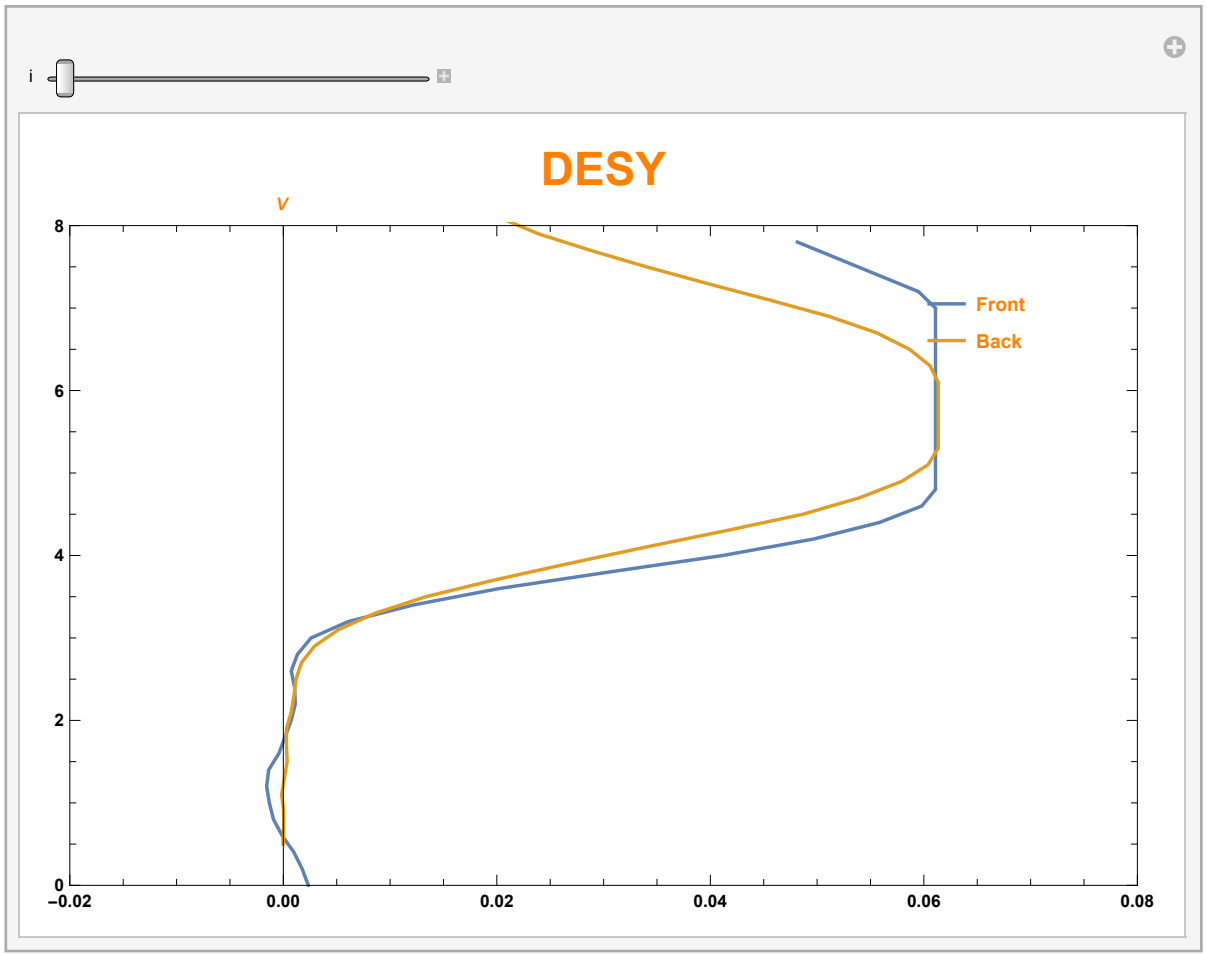
In[166]:=
```
Histogram[p3]
```

Out[166]=



In[167]:=
```
timerest1 = Range[0, 39] * dt;
Manipulate[
 ListPlot[{Transpose[{WienerFilter[r2[[i]], 2, .1], timerest1}], Transpose[{WienerFilter[r4[[i]], 2, .1], timerest1 + .5}]},
  PlotStyle → Directive[AbsoluteThickness[1.8]], LabelStyle → Directive[Orange, Bold],
  PlotLabel → Style[DESY, Large], AxesLabel → {nanosec, V}, PlotStyle → {Red, Blue, Green},
  PlotLegends → Placed[{"Front", "Back", "front-200mV/div"}, {0.85, 0.85}], PlotRange → {{-.02, .08}, {0, 8}},
  Joined → True, ImageSize → Large, Frame → True, FrameStyle → Black], {i, 1, 200, 1}, SaveDefinitions → True]
```

Out[168]=



In[208]:=
```
Clear[a2, a4];
a2 = Table[Transpose[{WienerFilter[r2[[i]], 2, .1], timerest1}], {i, 1, 1105}];
a4 = Table[Transpose[{WienerFilter[r4[[i]], 2, .1], timerest1 + .5}], {i, 1, 1105}];
```

In[211]:=
```
Dimensions[a2]
```

Out[211]=
```
{1105, 40, 2}
```

In[212]:=
```
b2 = ConstantArray[0, {1105, 9, 2}]; b4 = ConstantArray[0, {1105, 9, 2}];
Do[
  done = False;
  Do[
   If[done, Goto[vvv]];
   If[a2[[k, i, 1]] > 0.06, Goto[ppp], Goto[vvv]];
   Label[ppp];
   done = True;
   j = i - 1;
   Label[vvv];
   , {i, 40}];
  b2[[k]] = Take[a2[[k]], {j - 8, j}];
  b4[[k]] = Take[a4[[k]], {j - 10, j - 2}];
  , {k, 1105}];
```

In[215]:=
```
fitset = Transpose[{b2, b4}];
Export["DESY_10mV_fitset", fitset, "csv"];
```

In[177]:=
```
f = Interpolation[b2]
f[0.03]
```

Out[177]=
```
InterpolatingFunction[ 

]
```

Out[178]=
```
3.92348
```

Now create an nTuple with the 3 peak values for each event.

In[189]:=
```
Dimensions[pkapd1]
```

Out[189]=
```
{10 000}
```

In[179]:=

In[180]:=
```
peakntup = Transpose[{peak1, peak2, peak3, area3, rest1, rest2, rest3}];
```

Transpose::nmtx : The first two levels of
{{0.017226 , 0.144421 , 0.105596 , 0.263509 , 1.22347 , 0.284789 , 0.0248523 , 0.0145594 , 0.136901 , 0.162767 , 0.124955 , 0.0153594 , 0.224951 , 0.222498 , ≪23≫,
0.122448 , 0.112742 , 0.016426 , 0.136795 , 0.203778 , 0.104103 , 0.136155 , 0.00911964 , 0.208312 , 0.144101 , 0.0180259 , 0.299721 , 0.0237857 , ≪1950≫}, ≪5≫, {
≪1≫}} cannot be transposed . ≫

In[181]:=
```
GraphicsGrid[{{Histogram[Select[peakntup, (#[[4]] > 5 && #[[2]] > 0.06) &][[All, 1]], {0.05, .4, .01}],
   Histogram[Select[peakntup, (#[[4]] > 5 && #[[4]] < 15) &][[All, 1]], {0.05, .4, .01}]},
  {Histogram[Select[peakntup, (#[[4]] > 5 && #[[1]] > 0.06) &][[All, 2]], {0.05, .4, .01}],
   Histogram[Select[peakntup, (#[[4]] > 5 && #[[4]] < 15) &][[All, 2]], {0.05, .4, .01}]}}, Frame → True,
 PlotLabel → "Ch1 with min cut on 2&3 and only cut on 3, then same for ch2", ImageSize → Large]
```

Transpose::argt : Transpose called with 0 arguments ; 1 or 2 arguments are expected . ≫

Transpose::argt : Transpose called with 0 arguments ; 1 or 2 arguments are expected . ≫

Histogram ::ldata : Transpose[] is not a valid dataset or list of datasets . ≫

Transpose::argt : Transpose called with 0 arguments ; 1 or 2 arguments are expected . ≫

General::stop : Further output of Transpose::argt will be suppressed during this calculation . ≫

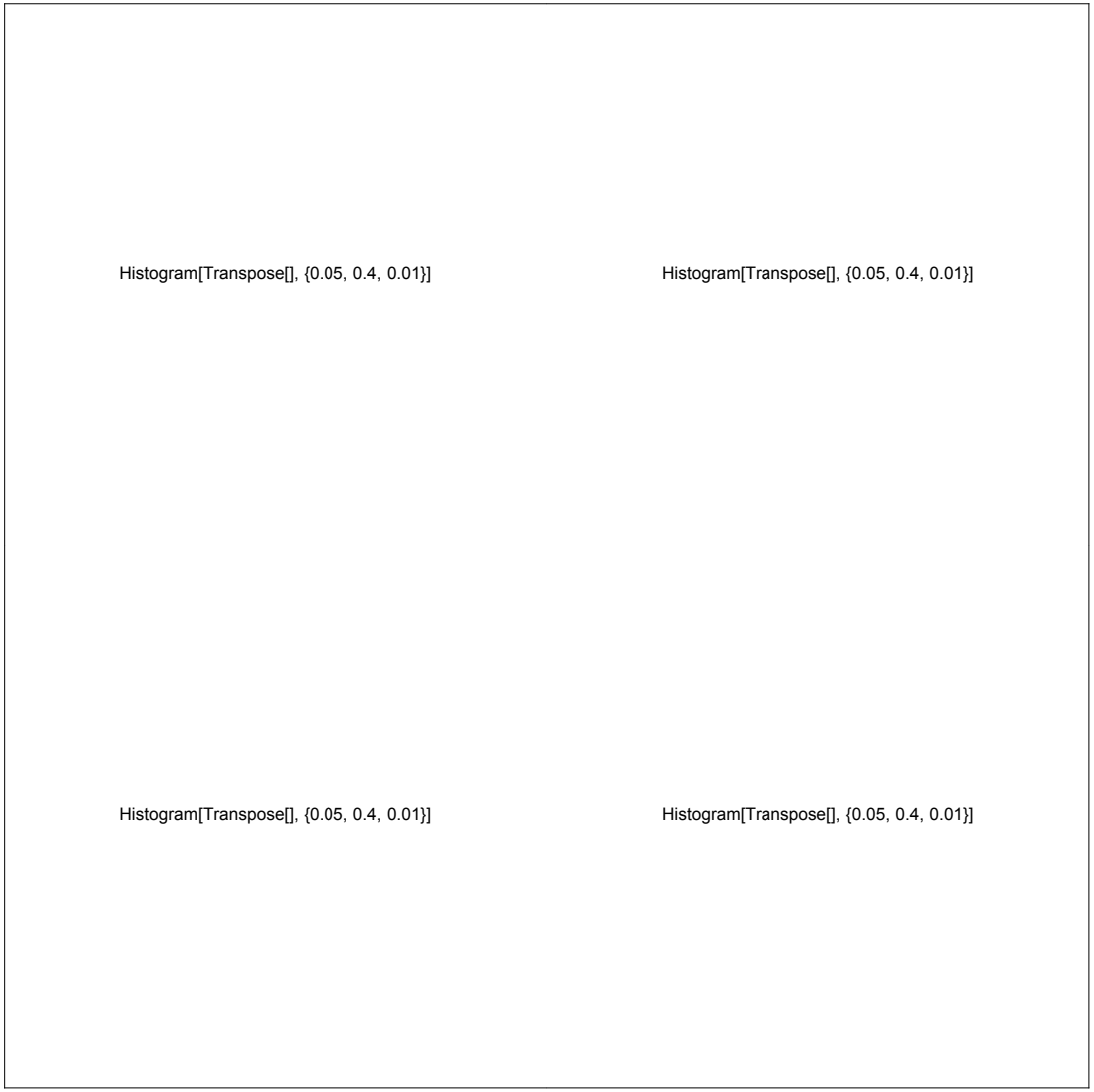
Histogram ::ldata : Transpose[] is not a valid dataset or list of datasets . ≫

Histogram ::ldata : Transpose[] is not a valid dataset or list of datasets . ≫

General::stop : Further output of Histogram ::ldata will be suppressed during this calculation . ≫

Ch1 with min cut on 2&3 and only cut on 3, then same for ch2

Out[181]=



In[182]:=
```
peakcut = Select[peakntup, (#[[1]] > .3 && #[[2]] > .3 && #[[1]] < 0.4 && #[[2]] < 0.4 && #[[4]] > 5 && #[[4]] < 15) &];
```

Transpose::argt : Transpose called with 0 arguments ; 1 or 2 arguments are expected . >>

In[183]:=
```
Dimensions[peakcut]
```

Out[183]= {0}

In[184]:=
```
Dimensions[peakcut[[1, 7]]]
```

Part::partw : Part 1 of Transpose[] does not exist . >>

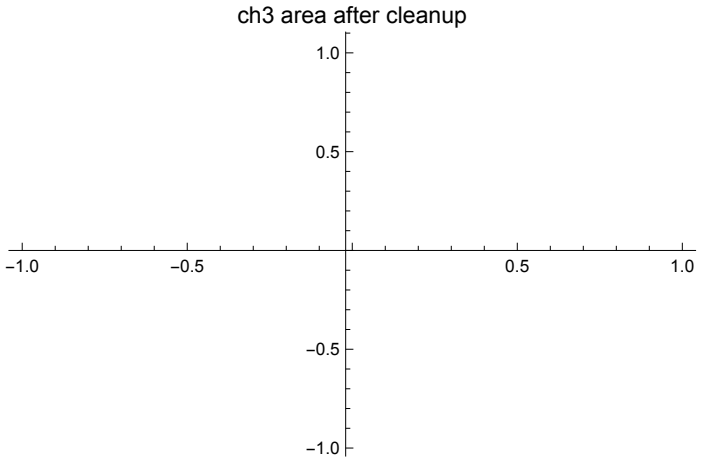
Out[184]= {3}

In[185]:=
```
{p1, p2, p3, a3, r1, r2, r3} = Transpose[peakcut];
```

Set::shape : Lists {p1, p2, p3, a3, r1, r2, r3} and Transpose[Transpose[]] are not the same shape. >>

In[186]:=
```
Histogram[{(*area3,*)a3}, {-5, 20, .1}, PlotLabel → "ch3 area after cleanup"]
```

Out[186]=



In[187]:=
```
GraphicsRow[{ListPlot[Transpose[{p1, p2}], PlotRange → {{0, .4}, {0, .4}}],
   PairedHistogram[p1, Style[p2, Red], {0, 0.4, .02}, AxesOrigin → {0, 0}]}, Frame → True,
 PlotLabel → "Compare ch1 and ch2 Amplitudes, difference most likely due to preamp gain diff.", ImageSize → Full]
```

Transpose ::nmtx : The first two levels of
{{0.167286 , 0.186795 , 0.121195 , 0.121219 , 0.144183 , 0.19597 , 0.108598 , 0.139746 , 0.154459 , 0.144126 , 0.152409 , 0.11798 , 0.177705 , 0.199333 , 0.126106 ,
0.150509 , 0.103835 , ≪18≫, 0.128825 , 0.135897 , 0.187977 , 0.159758 , 0.148246 , 0.120504 , 0.152746 , 0.127533 , 0.126122 , 0.139901 , 0.153159 , 0.153482 ,
0.136593 , 0.10229 , 0.106807 , ≪1055≫}, p2} cannot be transposed . ≫

ListPlot ::lpn :
Transpose [{{0.167286 , 0.186795 , 0.121195 , 0.121219 , 0.144183 , 0.19597 , 0.108598 , 0.139746 , 0.154459 , 0.144126 , 0.152409 , 0.11798 , 0.177705 , 0.199333 , 0.126106 ,
0.150509 , ≪19≫, 0.128825 , 0.135897 , 0.187977 , 0.159758 , 0.148246 , 0.120504 , 0.152746 , 0.127533 , 0.126122 , 0.139901 , 0.153159 , 0.153482 , 0.136593 ,
0.10229 , 0.106807 , ≪1055≫}, p2}] is not a list of numbers or pairs of numbers . ≫

ListPlot ::lpn :
Transpose [{{0.167286 , 0.186795 , 0.121195 , 0.121219 , 0.144183 , 0.19597 , 0.108598 , 0.139746 , 0.154459 , 0.144126 , 0.152409 , 0.11798 , 0.177705 , 0.199333 , 0.126106 ,
0.150509 , ≪19≫, 0.128825 , 0.135897 , 0.187977 , 0.159758 , 0.148246 , 0.120504 , 0.152746 , 0.127533 , 0.126122 , 0.139901 , 0.153159 , 0.153482 , 0.136593 ,
0.10229 , 0.106807 , ≪1055≫}, p2}] is not a list of numbers or pairs of numbers . ≫

PairedHistogram ::ldata :
{{0.167286 , 0.186795 , 0.121195 , 0.121219 , 0.144183 , 0.19597 , 0.108598 , 0.139746 , 0.154459 , 0.144126 , 0.152409 , 0.11798 , 0.177705 , 0.199333 , 0.126106 , 0.150509 ,
0.103835 , ≪18≫, 0.128825 , 0.135897 , 0.187977 , 0.159758 , 0.148246 , 0.120504 , 0.152746 , 0.127533 , 0.126122 , 0.139901 , 0.153159 , 0.153482 , 0.136593 ,
0.10229 , 0.106807 , ≪1055≫}, p2} is not a valid dataset or list of datasets . ≫

PairedHistogram ::ldata :
{{0.167286 , 0.186795 , 0.121195 , 0.121219 , 0.144183 , 0.19597 , 0.108598 , 0.139746 , 0.154459 , 0.144126 , 0.152409 , 0.11798 , 0.177705 , 0.199333 , 0.126106 , 0.150509 ,
0.103835 , ≪18≫, 0.128825 , 0.135897 , 0.187977 , 0.159758 , 0.148246 , 0.120504 , 0.152746 , 0.127533 , 0.126122 , 0.139901 , 0.153159 , 0.153482 , 0.136593 ,
0.10229 , 0.106807 , ≪1055≫}, p2} is not a valid dataset or list of datasets . ≫

Compare ch1 and ch2 Amplitudes, difference most likely due to preamp gain diff.

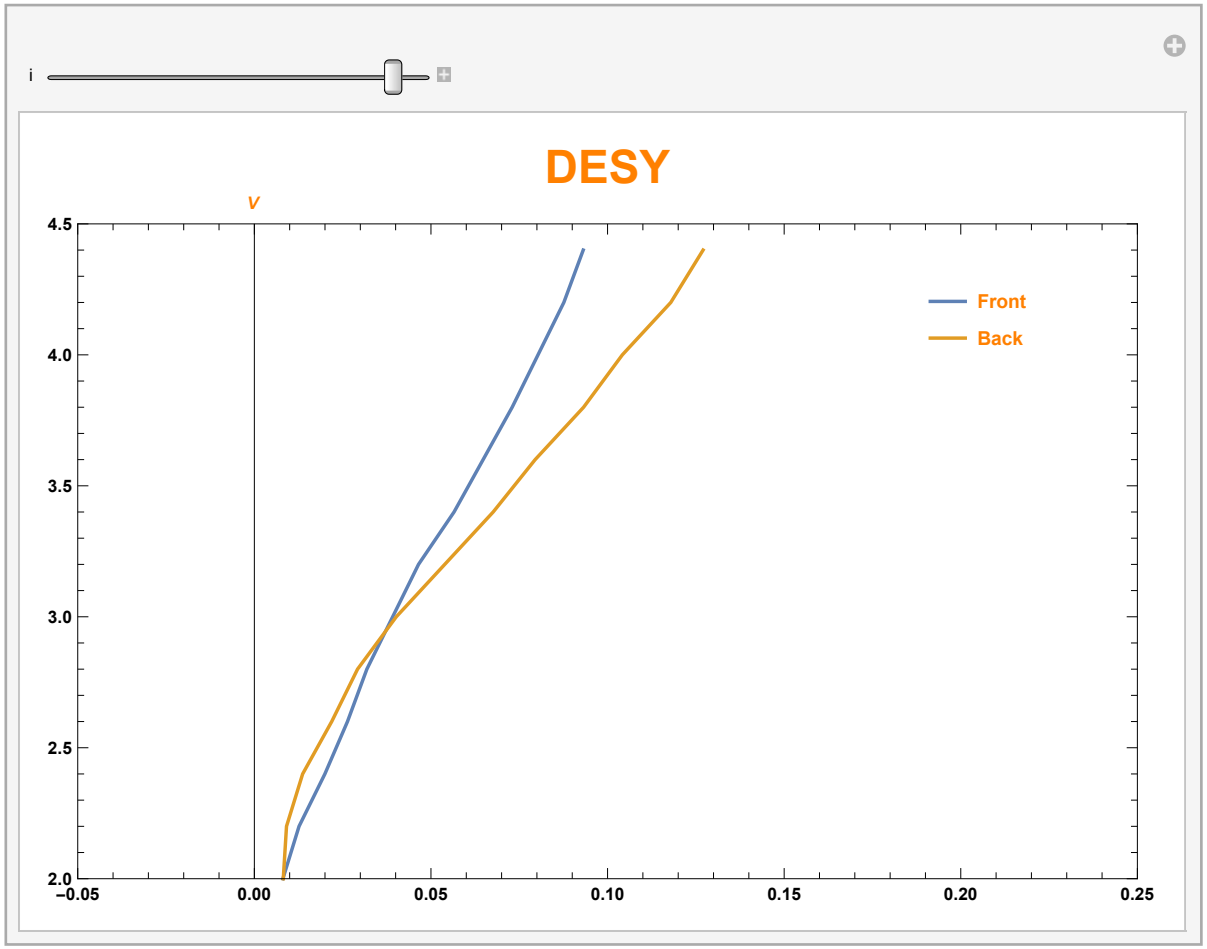
Out[187]=

0.116686, 0.105134, 0.175596, 0.149799, 0.151811, 0.196601, 0.132215, 0.107305, 0.129416, 0.164155, 0.122632, 0.156725, 0.138701, 0.152049, 0.154229, 0.149342, 0.182583, 0.161137, 0.146073, 0.191967, 0.17922, 0.102547, 0.110438, 0.172476, 0.146312, 0.157531, 0.156511, 0.166175, 0.179236, 0.148081, 0.118249, 0.131664, 0.181426, 0.122267, 0.146523, 0.107049, 0.119974, 0.178987, 0.178279, 0.167669, 0.164207, 0.119744, 0.134674, 0.153927, 0.147103, 0.130989, 0.149203, 0.140624, 0.137025, 0.113733, 0.12185, 0.15654, 0.185461, 0.10734, 0.108496, 0.13674, 0.1213, 0.156176, 0.145715, 0.144634, 0.12923, 0.149382, 0.171581, 0.163773, 0.130167, 0.161753, 0.196741, 0.135349, 0.188276, 0.177834, 0.105513, 0.170399, 0.185024, 0.166943, 0.12561, 0.136912, 0.172743, 0.166209, 0.176496, 0.159639, 0.127034, 0.138796, 0.155439, 0.121199, 0.131033, 0.183462, 0.121751, 0.170484, 0.140689, 0.173071, 0.165421, 0.14144, 0.143549, 0.176819, 0.112872, 0.134807, 0.113133, 0.111634, 0.199664, 0.173855, 0.145485, 0.157335, 0.18468, 0.111001, 0.113867, 0.140718, 0.149842, 0.1918, 0.195314, 0.15078, 0.128252, 0.146472, 0.141243, 0.148744, 0.112126, 0.146432, 0.175629, 0.165882, 0.118838, 0.107421, 0.161565, 0.186192, 0.117929, 0.128943, 0.177659, 0.15453, 0.117754, 0.159934, 0.170263, 0.14836, 0.168089, 0.143223, 0.129125, 0.156607, 0.123384, 0.123092, 0.153307, 0.157115, 0.164939, 0.142073, 0.120004, 0.101652, 0.161773, 0.139548, 0.129511, 0.174268, 0.150085, 0.191469, 0.12896, 0.155739, 0.119894, 0.137916, 0.139141, 0.182422, 0.153982, 0.150203, 0.152736, 0.164182, 0.137247, 0.133308, 0.104383, 0.144292, 0.144337, 0.121636, 0.156137, 0.187678, 0.163662, 0.172882, 0.164198, 0.110574, 0.147511, 0.151859, 0.185749, 0.17179, 0.16103, 0.123777, 0.129703, 0.114563, 0.136959, 0.145465, 0.152543, 0.140468, 0.121829, 0.116946, 0.157518, 0.149082, 0.142459, 0.140384, 0.182794, 0.152405, 0.130022, 0.102556, 0.14556, 0.123694, 0.111693, 0.168793, 0.147478, 0.181095, 0.148545, 0.11653, 0.13485, 0.131564, 0.16774, 0.119315, 0.166119, 0.127956, 0.118395, 0.112949, 0.133052, 0.105633, 0.118385, 0.127618, 0.157509, 0.168637, 0.16877, 0.118439, 0.143142, 0.188748, 0.168644, 0.17161, 0.142234, 0.167704, 0.103036, 0.181115, 0.135923, 0.154673, 0.181463, 0.114379, 0.10664, 0.134735, 0.165939, 0.125717, 0.138115, 0.170264, 0.161829, 0.126873, 0.148035, 0.103053, 0.114396, 0.145942, 0.177841, 0.17266, 0.154854, 0.121393, 0.192076, 0.141209, 0.140493, 0.126906, 0.136515, 0.139778, 0.125102, 0.149351, 0.112063, 0.138352, 0.118146, 0.146391, 0.116241, 0.136327, 0.128585, 0.160833, 0.19104, 0.121177, 0.150513, 0.169018, 0.141303, 0.103316, 0.101252, 0.159201, 0.113326, 0.151376, 0.152984, 0.183923, 0.164161, 0.100408, 0.146784, 0.146669, 0.120338, 0.142696, 0.123206, 0.13868, 0.196874, 0.12801, 0.120508, 0.105181, 0.162383, 0.190222, 0.18807, 0.165046, 0.188199, 0.135254, 0.126452, 0.157162, 0.152802, 0.145986, 0.110817, 0.106999, 0.121128, 0.12627, 0.186453, 0.127887, 0.121937, 0.19327, 0.158314, 0.164065, 0.19341, 0.158999, 0.144591, 0.160387, 0.128412, 0.134909, 0.141929, 0.142561, 0.134568, 0.165065, 0.144447, 0.141045, 0.145018, 0.140302, 0.14267, 0.15905, 0.16935, 0.140006, 0.181524, 0.143036, 0.12316, 0.124235, 0.183976, 0.143277, 0.183733, 0.134501, 0.134615, 0.122472, 0.127038, 0.139043, 0.162864,

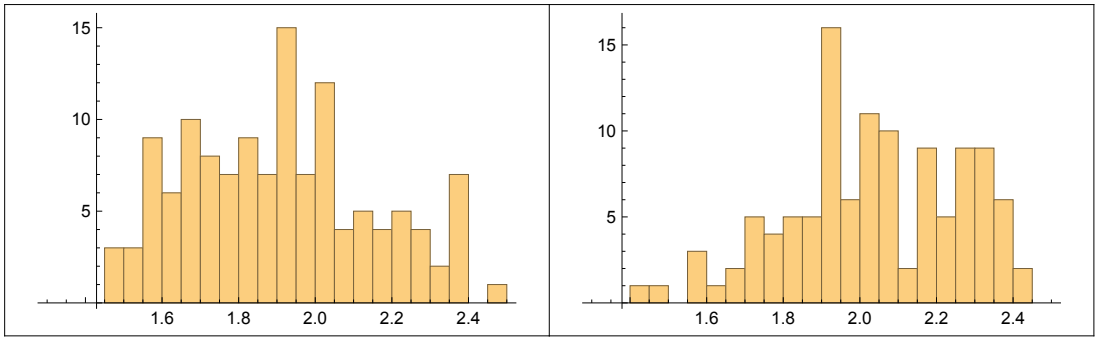
0.175448, 0.157406, 0.151214, 0.160108, 0.166089, 0.150531, 0.176466, 0.136689, 0.116686, 0.105134, 0.175596, 0.149799, 0.151811, 0.196601, 0.132215, 0.107305, 0.129416, 0.164155, 0.122632, 0.156725, 0.138701, 0.152049, 0.154229, 0.149342, 0.182583, 0.161137, 0.146073, 0.191967, 0.17922, 0.102547, 0.110438, 0.172476, 0.146312, 0.157531, 0.156511, 0.166175, 0.179236, 0.148081, 0.118249, 0.131664, 0.181426, 0.122267, 0.146523, 0.107049, 0.119974, 0.178987, 0.178279, 0.167669, 0.164207, 0.119744, 0.134674, 0.153927, 0.147103, 0.130989, 0.149203, 0.140624, 0.137025, 0.113733, 0.12185, 0.15654, 0.185461, 0.10734, 0.108496, 0.13674, 0.1213, 0.156176, 0.145715, 0.144634, 0.12923, 0.149382, 0.171581, 0.163773, 0.130167, 0.161753, 0.196741, 0.135349, 0.188276, 0.177834, 0.105513, 0.170399, 0.185024, 0.166943, 0.12561, 0.136912, 0.172743, 0.166209, 0.176496, 0.159639, 0.127034, 0.138796, 0.155439, 0.121199, 0.131033, 0.183462, 0.121751, 0.170484, 0.140689, 0.173071, 0.165421, 0.14144, 0.143549, 0.176819, 0.112872, 0.134807, 0.113133, 0.111634, 0.199664, 0.173855, 0.145485, 0.157335, 0.18468, 0.111001, 0.113867, 0.140718, 0.149842, 0.1918, 0.195314, 0.15078, 0.128252, 0.146472, 0.141243, 0.148744, 0.112126, 0.146432, 0.175629, 0.165882, 0.118838, 0.107421, 0.161565, 0.186192, 0.117929, 0.128943, 0.177659, 0.15453, 0.117754, 0.159934, 0.170263, 0.14836, 0.168089, 0.143223, 0.129125, 0.156607, 0.123384, 0.123092, 0.153307, 0.157115, 0.164939, 0.142073, 0.120004, 0.101652, 0.161773, 0.139548, 0.129511, 0.174268, 0.150085, 0.191469, 0.12896, 0.155739, 0.119894, 0.137916, 0.139141, 0.182422, 0.153982, 0.150203, 0.152736, 0.164182, 0.137247, 0.133308, 0.104383, 0.144292, 0.144337, 0.121636, 0.156137, 0.187678, 0.163662, 0.172882, 0.164198, 0.110574, 0.147511, 0.151859, 0.185749, 0.17179, 0.16103, 0.123777, 0.129703, 0.114563, 0.136959, 0.145465, 0.152543, 0.140468, 0.121829, 0.116946, 0.157518, 0.149082, 0.142459, 0.140384, 0.182794, 0.152405, 0.130022, 0.102556, 0.14556, 0.123694, 0.111693, 0.168793, 0.147478, 0.181095, 0.148545, 0.11653, 0.13485, 0.131564, 0.16774, 0.119315, 0.166119, 0.127956, 0.118395, 0.112949, 0.133052, 0.105633, 0.118385, 0.127618, 0.157509, 0.168637, 0.16877, 0.118439, 0.143142, 0.188748, 0.168644, 0.17161, 0.142234, 0.167704, 0.103036, 0.181115, 0.135923, 0.154673, 0.181463, 0.114379, 0.10664, 0.134735, 0.165939, 0.125717, 0.138115, 0.170264, 0.161829, 0.126873, 0.148035, 0.103053, 0.114396, 0.145942, 0.177841, 0.17266, 0.154854, 0.121393, 0.192076, 0.141209, 0.140493, 0.126906, 0.136515, 0.139778, 0.125102, 0.149351, 0.112063, 0.138352, 0.118146, 0.146391, 0.116241, 0.136327, 0.128585, 0.160833, 0.19104, 0.121177, 0.150513, 0.169018, 0.141303, 0.103316, 0.101252, 0.159201, 0.113326, 0.151376, 0.152984, 0.183923, 0.164161, 0.100408, 0.146784, 0.146669, 0.120338, 0.142696, 0.123206, 0.13868, 0.196874, 0.12801, 0.120508, 0.105181, 0.162383, 0.190222, 0.18807, 0.165046, 0.188199, 0.135254, 0.126452, 0.157162, 0.152802, 0.145986, 0.110817, 0.106999, 0.121128, 0.12627, 0.186453, 0.127887, 0.121937, 0.19327, 0.158314, 0.164065, 0.19341, 0.158999, 0.144591, 0.160387, 0.128412, 0.134909, 0.141929, 0.142561, 0.134568, 0.165065, 0.144447, 0.141045, 0.145018, 0.140302, 0.14267, 0.15905, 0.16935, 0.140006, 0.181524, 0.143036, 0.12316, 0.124235, 0.183976, 0.143277, 0.183733, 0.134501, 0.134615, 0.122472, 0.127038, 0.139043,

```
r1f = Table[WienerFilter[r1[[i]], 3, .1], {i, 132}];
r2f = Table[WienerFilter[r2[[i]], 3, .1], {i, 132}];
r3f = Table[WienerFilter[r3[[i]], 3, .1], {i, 132}];

r1fit = Table[Transpose[{Take[r1f[[i]], {11, 23}], Take[timerest, {11, 23}]}], {i, 132}];
r2fit = Table[Transpose[{Take[r2f[[i]], {11, 23}], Take[timerest, {11, 23}]}], {i, 132}];
```

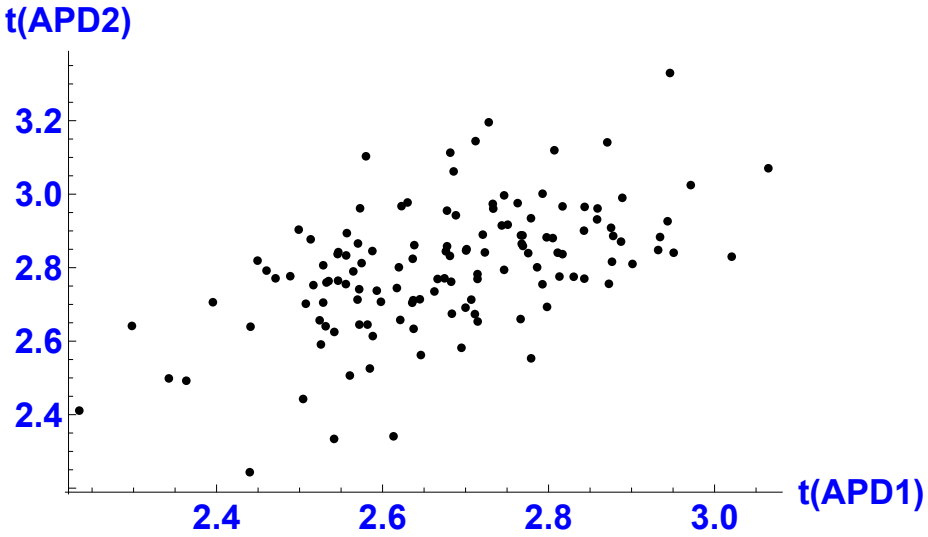
```
Manipulate[ListPlot[{r1fit[[i]], r2fit[[i]](*,Transpose[{timerest,0.3*r3[[i]]}]*)},
  PlotStyle → Directive[AbsoluteThickness[1.8]], LabelStyle → Directive[Orange, Bold],
  PlotLabel → Style[DESY, Large], AxesLabel → {nanosec, V}, PlotStyle → {Red, Blue, Green},
  PlotLegends → Placed[{"Front", "Back", "trigger APD"}, {0.85, 0.85}], PlotRange → {{-.05, .25}, {2, 4.5}},
  Joined → True, ImageSize → Large, Frame → True, FrameStyle → Black], {i, 1, 132, 1}, SaveDefinitions → True]
```



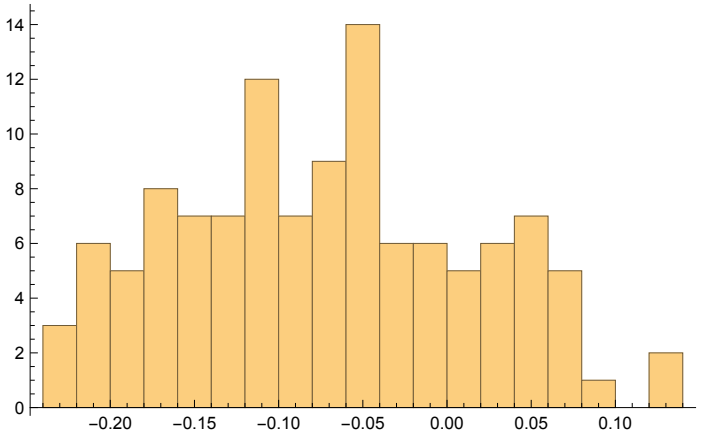
```
difftr = ConstantArray[0, 4];

trig2 = ConstantArray[0, 132]; trig3 = ConstantArray[0, 132];
tr2 = ConstantArray[0, 132]; tr3 = ConstantArray[0, 132];
ithr = 2; timethreshold = {.02, .03, .04, .05};
Do[
  Clear[fun];
  fun[x_] := Evaluate[Fit[r1fit[[i]], {1, x}, x]];
  tr2[[i]] = Evaluate[fun[.11]] - Evaluate[fun[.015]];
  trig2[[i]] = Evaluate[fun[timethreshold[[ithr]]]];
  , {i, 132}];
Do[
  Clear[fun];
  fun[x_] := Evaluate[Fit[r2fit[[i]], {1, x}, x]];
  tr3[[i]] = Evaluate[fun[.11]] - Evaluate[fun[.015]];
  trig3[[i]] = Evaluate[fun[timethreshold[[ithr]]]];
  , {i, 132}];

GraphicsRow[{Histogram[tr2, {1.3, 2.5, .05}], Histogram[tr3, {1.3, 2.5, .05}]}, Frame → All, ImageSize → Large]
```



```
ListPlot[Transpose[{trig2, trig3}], PlotStyle → Directive[Black, PointSize[Medium]],
 AxesLabel → {"t(APD1)", "t(APD2)"}, LabelStyle → Directive[Blue, Bold, 18], AxesStyle → Directive[Bold]]
```




```
trigdif = (trig2 - trig3) / Sqrt[2];
Histogram[trigdif, {-0.24, .14, .02}]
```



Now apply the same cuts to all 29 runs and compress them into a single data file.

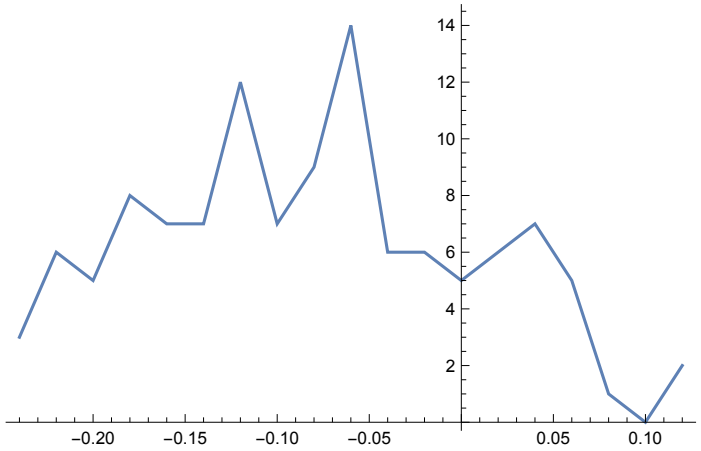
```
bc = BinCounts[trigdif, {-0.24, .14, .02}];
Length[bc]
```

```
19
```

```
bins = Range[-0.24, .14 - .02, .02]
```

```
{-0.24, -0.22, -0.2, -0.18, -0.16, -0.14, -0.12, -0.1, -0.08, -0.06, -0.04, -0.02, 0., 0.02, 0.04, 0.06, 0.08, 0.1, 0.12}
```

```
bc1 = Transpose[{bins, bc}];
ListPlot[bc1, Joined -> True]
```




```
ff = FindFit[bc1, {amp1 * s1 * PDF[NormalDistribution[m1, s1], x],
    -.15 < m1 < 0}, {{m1, -.06}, {s1, .06}, {amp1, 60}}, x]
```

```
{m1 → -0.0929363, s1 → 0.107231, amp1 → 23.833}
```

```
{mean, sig} = {m1, s1} /. ff
```

```
{-0.0929363, 0.107231}
```

```
difftr[[ithr]] = sig
```

```
0.107231
```

```
difftr
```

```
{0.0829559, 0.0833169, 0.0841702, 0.0846339}
```

```
dst1 = r1;
dst2 = r2;
dst3 = r3;
Dimensions[dst1]
```

```
{442, 50}
```

```
Do[
  filename = Namelist[[irun]];
  Print[filename];
scopedata = Import[filename, "csv"];
  v1 = Table[Take[scopedata[[4 * (i - 1) + 1]], {4, 255}], {i, 2000}];
  v2 = Table[Take[scopedata[[4 * (i - 1) + 2]], {4, 255}], {i, 2000}];
  v3 = Table[Take[scopedata[[4 * (i - 1) + 3]], {4, 255}], {i, 2000}];
  inbase1 = Table[Mean[Take[v1[[i]], 120]], {i, 2000}];
  innoise1 = Table[RootMeanSquare[Take[v1[[i]], 120] - inbase1[[i]]], {i, 2000}];
  inbase2 = Table[Mean[Take[v2[[i]], 120]], {i, 2000}];
  innoise2 = Table[RootMeanSquare[Take[v2[[i]], 120] - inbase2[[i]]], {i, 2000}];
  inbase3 = Table[Mean[Take[v3[[i]], 120]], {i, 2000}];
  innoise3 = Table[RootMeanSquare[Take[v3[[i]], 120] - inbase3[[i]]], {i, 2000}];
  Print[
   Style[{" noise1= ", Mean[innoise1], " noise2= ", Mean[innoise2], " noise3= ", Mean[innoise3], " volts"} , Medium, Red]];
  rest1 = Table[(Take[v1[[i]], {116, 165}] - inbase1[[i]]), {i, 2000}];
  rest2 = Table[(Take[v2[[i]], {116, 165}] - inbase2[[i]]), {i, 2000}];
  rest3 = Table[-(Take[v3[[i]], {116, 165}] - inbase3[[i]]), {i, 2000}];
  peak1 = Table[Max[rest1[[i]]], {i, 2000}];
peak2 = Table[Max[rest2[[i]]], {i, 2000}];
  peak3 = Table[Max[rest3[[i]]], {i, 2000}];
  area3 = Table[Sum[rest3[[i, j]], {j, 1, 50}], {i, 2000}];
  peakntup = Transpose[{peak1, peak2, peak3, area3, rest1, rest2, rest3}];
  peakcut = Select[peakntup, (#[[1]] > .050 && #[[2]] > .050 && #[[1]] < 0.4 && #[[2]] < 0.4 && #[[4]] > 5 && #[[4]] < 15) &];
  Print[Dimensions[peakcut]];
  Clear[p1, p2, p3, a3, r1, r2, r3]; {p1, p2, p3, a3, r1, r2, r3} = Transpose[peakcut];
  temp = Join[dst1, r1]; dst1 = temp;
  Print[Dimensions[dst1]];
  temp = Join[dst2, r2]; dst2 = temp;
  temp = Join[dst3, r3]; dst3 = temp;
  , {irun, 4, 31, 1}];
```

```
LeCroy-RTW-2014-03-01-004.csv
```

```
{ noise1= , 0.00311984,  noise2= , 0.00281714,  noise3= , 0.00517762,  volts}
```

```
{439, 7}
```

```
{881, 50}
LeCroy-RTW-2014-03-01-005.csv
{ noise1= , 0.00294786,  noise2= , 0.00282301,  noise3= , 0.00514624,  volts}
{503, 7}
{1384, 50}
LeCroy-RTW-2014-03-01-006.csv
{ noise1= , 0.00312083,  noise2= , 0.00283603,  noise3= , 0.0051452,  volts}
{473, 7}
{1857, 50}
LeCroy-RTW-2014-03-01-007.csv
{ noise1= , 0.00305255,  noise2= , 0.00292348,  noise3= , 0.00517372,  volts}
{512, 7}
{2369, 50}
LeCroy-RTW-2014-03-01-008.csv
{ noise1= , 0.00298552,  noise2= , 0.00299295,  noise3= , 0.00517331,  volts}
{473, 7}
{2842, 50}
LeCroy-RTW-2014-03-01-009.csv
{ noise1= , 0.00290895,  noise2= , 0.00278221,  noise3= , 0.00516971,  volts}
{481, 7}
{3323, 50}
LeCroy-RTW-2014-03-01-010.csv
{ noise1= , 0.00298493,  noise2= , 0.00284811,  noise3= , 0.00516356,  volts}
{500, 7}
{3823, 50}
```

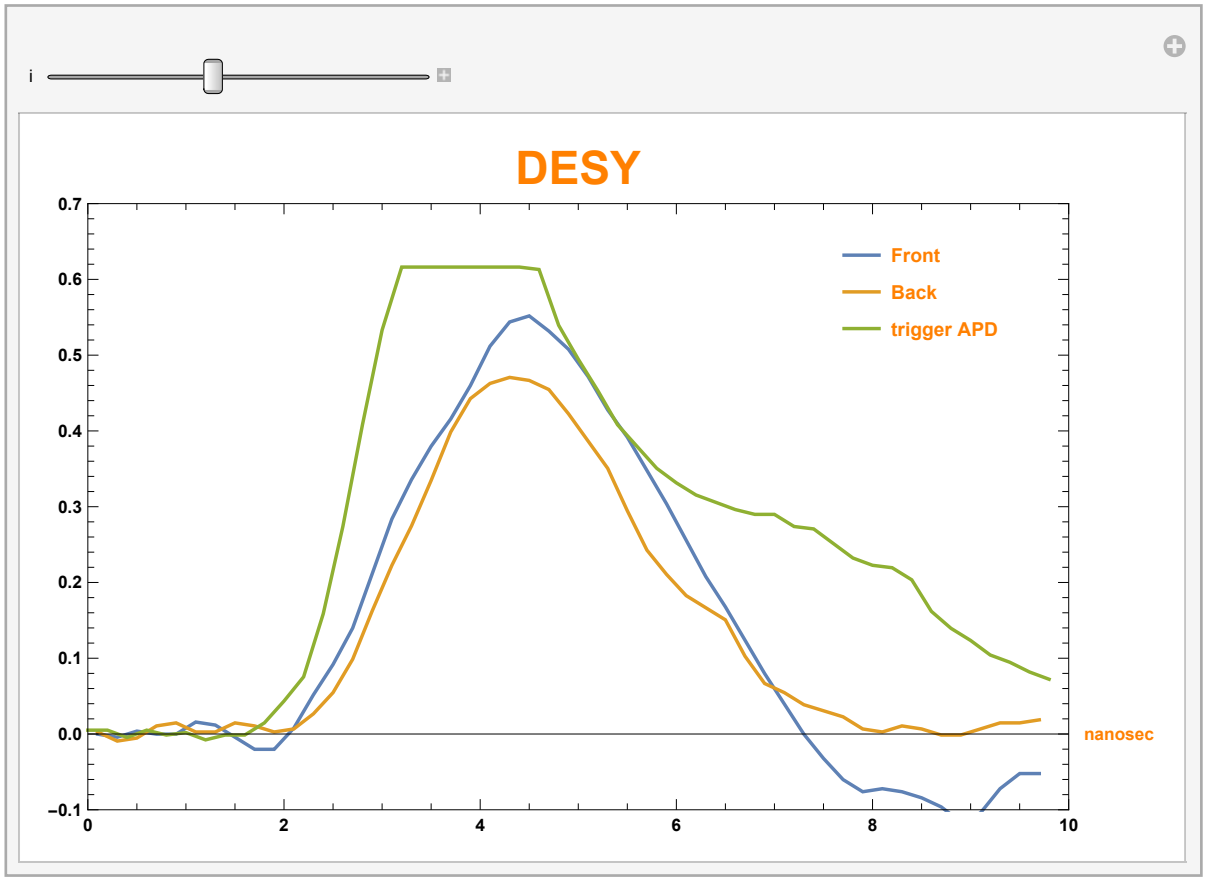
```
LeCroy-RTW-2014-03-01-011.csv
{ noise1= , 0.00318095,  noise2= , 0.00293835,  noise3= , 0.0051644,  volts}
{497, 7}
{4320, 50}
LeCroy-RTW-2014-03-01-012.csv
{ noise1= , 0.00291051,  noise2= , 0.00282119,  noise3= , 0.0051604,  volts}
{493, 7}
{4813, 50}
LeCroy-RTW-2014-03-01-013.csv
{ noise1= , 0.00311221,  noise2= , 0.00288509,  noise3= , 0.00517132,  volts}
{472, 7}
{5285, 50}
LeCroy-RTW-2014-03-01-014.csv
{ noise1= , 0.00308155,  noise2= , 0.00283809,  noise3= , 0.00516245,  volts}
{537, 7}
{5822, 50}
LeCroy-RTW-2014-03-01-015.csv
{ noise1= , 0.00304498,  noise2= , 0.00278098,  noise3= , 0.00515215,  volts}
{510, 7}
{6332, 50}
LeCroy-RTW-2014-03-01-016.csv
{ noise1= , 0.00277865,  noise2= , 0.00282392,  noise3= , 0.00513785,  volts}
{451, 7}
{6783, 50}
LeCroy-RTW-2014-03-01-017.csv
```

```
{ noise1= , 0.00306938,  noise2= , 0.00278624,  noise3= , 0.00515188,  volts}
{482, 7}
{7265, 50}
LeCroy-RTW-2014-03-01-018.csv
{ noise1= , 0.00299438,  noise2= , 0.0028053,  noise3= , 0.00516719,  volts}
{492, 7}
{7757, 50}
LeCroy-RTW-2014-03-01-019.csv
{ noise1= , 0.00315807,  noise2= , 0.00274332,  noise3= , 0.0051945,  volts}
{492, 7}
{8249, 50}
LeCroy-RTW-2014-03-01-020.csv
{ noise1= , 0.00305053,  noise2= , 0.00276729,  noise3= , 0.00515833,  volts}
{421, 7}
{8670, 50}
LeCroy-RTW-2014-03-01-021.csv
{ noise1= , 0.00322069,  noise2= , 0.0029164,  noise3= , 0.00516756,  volts}
{498, 7}
{9168, 50}
LeCroy-RTW-2014-03-01-022.csv
{ noise1= , 0.00288874,  noise2= , 0.00272246,  noise3= , 0.00517095,  volts}
{512, 7}
{9680, 50}
LeCroy-RTW-2014-03-01-023.csv
{ noise1= , 0.00301725,  noise2= , 0.00276976,  noise3= , 0.00514335,  volts}
```

```
{481, 7}
{10 161, 50}
LeCroy-RTW-2014-03-01-024.csv
{ noise1= , 0.00296826,  noise2= , 0.00280479,  noise3= , 0.00513266,  volts}
{475, 7}
{10 636, 50}
LeCroy-RTW-2014-03-01-025.csv
{ noise1= , 0.0029687,  noise2= , 0.00275753,  noise3= , 0.00514908,  volts}
{496, 7}
{11 132, 50}
LeCroy-RTW-2014-03-01-026.csv
{ noise1= , 0.00302132,  noise2= , 0.00292,  noise3= , 0.00516115,  volts}
{508, 7}
{11 640, 50}
LeCroy-RTW-2014-03-01-027.csv
{ noise1= , 0.0031349,  noise2= , 0.00285633,  noise3= , 0.00517059,  volts}
{483, 7}
{12 123, 50}
LeCroy-RTW-2014-03-01-028.csv
{ noise1= , 0.00308687,  noise2= , 0.00283672,  noise3= , 0.00513436,  volts}
{479, 7}
{12 602, 50}
LeCroy-RTW-2014-03-01-029.csv
{ noise1= , 0.00304686,  noise2= , 0.00275455,  noise3= , 0.00514187,  volts}
{475, 7}
```

```
{13 077, 50}
LeCroy-RTW-2014-03-01-030.csv
{ noise1= , 0.0031579,  noise2= , 0.00275324,  noise3= , 0.00514946,  volts}
{490, 7}
{13 567, 50}
LeCroy-RTW-2014-03-01-031.csv
{ noise1= , 0.00286406,  noise2= , 0.00274669,  noise3= , 0.00516735,  volts}
{500, 7}
{14 067, 50}
Export["DESY_50mV.dat", {timerest, dst1, dst2, dst3}, "csv"]
```

```
iorder = 2;
timerest = Range[0, 49] * dt;
Manipulate[ListPlot[{Transpose[{MovingAverage[timerest, iorder], MovingAverage[(5 * dst1[[i]]), iorder]}],
    Transpose[{MovingAverage[timerest, iorder], MovingAverage[(5 * dst2[[i]]), iorder]}], Transpose[{timerest, dst3[[i]]}]},
  PlotStyle → Directive[AbsoluteThickness[1.8]], LabelStyle → Directive[Orange, Bold],
  PlotLabel → Style[DESY, Large], AxesLabel → {nanosec, V}, PlotStyle → {Red, Blue, Green},
  PlotLegends → Placed[{"Front", "Back", "trigger APD"}, {0.85, 0.85}], PlotRange → {{0, 10.}, {-.1, .7}},
  Joined → True, ImageSize → Large, Frame → True, FrameStyle → Black], {i, 1, 200, 1}, SaveDefinitions → True]
```



```
Export["DESY_50mV_ch1.dat", dst1, "csv"]
Export["DESY_50mV_ch2.dat", dst2, "csv"]
Export["DESY_50mV_ch3.dat", dst3, "csv"]
```

```
DESY_50mV_ch1.dat
```

```
DESY_50mV_ch2.dat
```

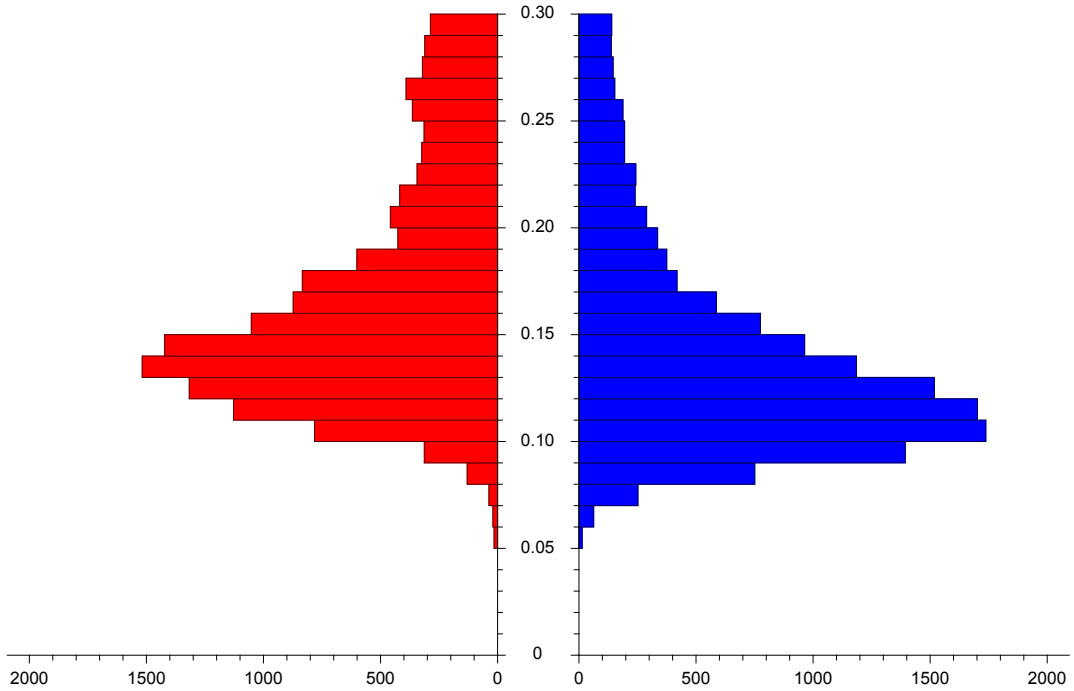
```
DESY_50mV_ch3.dat
```

```
ExpandFileName["DESY_50mV.dat"]
```

```
/Users/bastian/Desktop/desydata/DESY_50mV.dat
```

```
peak1 = Table[Max[dst1[[i]]], {i, 14 000}]; peak2 = Table[Max[dst2[[i]]], {i, 14 000}];
peak3 = Table[Max[dst3[[i]]], {i, 14 000}];
area3 = Table[Sum[dst3[[i, j]], {j, 1, 50}], {i, 14 000}];
```

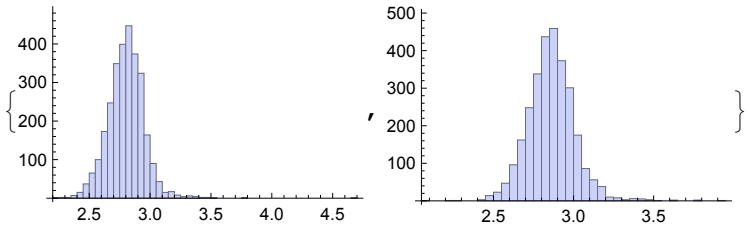
```
PairedHistogram[peak1, peak2, {0, 0.3, .01}, AxesOrigin → {0, 0}, ChartStyle → {{Red, Blue}, None}, ImageSize → Large]
```



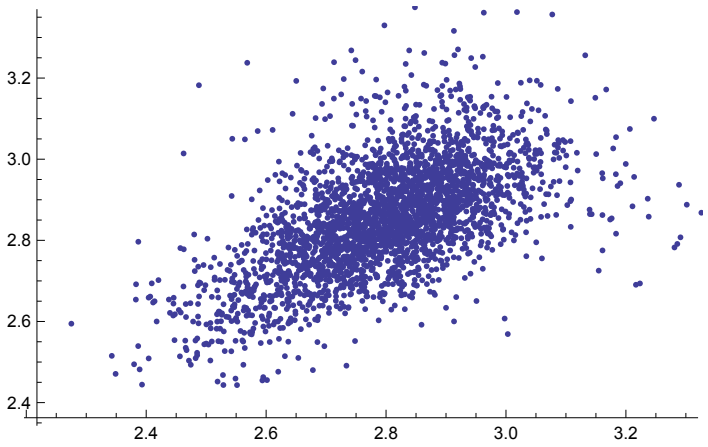

```
tenpercen1 = ConstantArray[0, 2900]; tenpercen2 = ConstantArray[0, 2900];
iorder = 3; thresh = 0.2;
Do[
  f =
   Interpolation[Transpose[{MovingAverage[timerest, iorder], MovingAverage[(dst1[[iev]] - thresh * peak1[[iev]]), iorder]}]];
  a = FindRoot[f[x], {x, 1, 5}, Method → "Brent"];
  tenpercen1[[iev]] = Flatten[a /. Rule → List][[2]];
  , {iev, 1, 2900, 1}];
```

```
Do[
  f =
   Interpolation[Transpose[{MovingAverage[timerest, iorder], MovingAverage[(dst2[[iev]] - thresh * peak2[[iev]]), iorder]}]];
  If[iev == 1031 || iev == 1520 || iev == 1917 || iev == 2704 || iev == 2894,
   a = FindRoot[f[x], {x, 2, 3}, Method → "Brent"], a = FindRoot[f[x], {x, 1, 5}, Method → "Brent"]];
  tenpercen2[[iev]] = Flatten[a /. Rule → List][[2]];
  , {iev, 1, 2900, 1}];
```

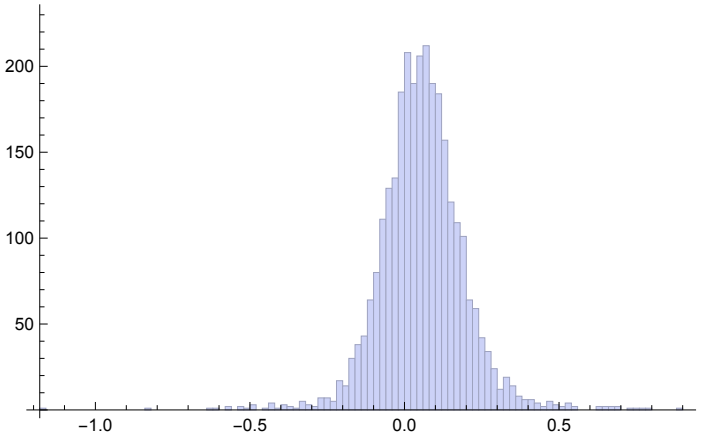
```
{Histogram[tenpercen1], Histogram[tenpercen2]}
```




```
ListPlot[Transpose[{tenpercen1, tenpercen2}]]
```



```
Histogram[tenpercen2 - tenpercen1]
```

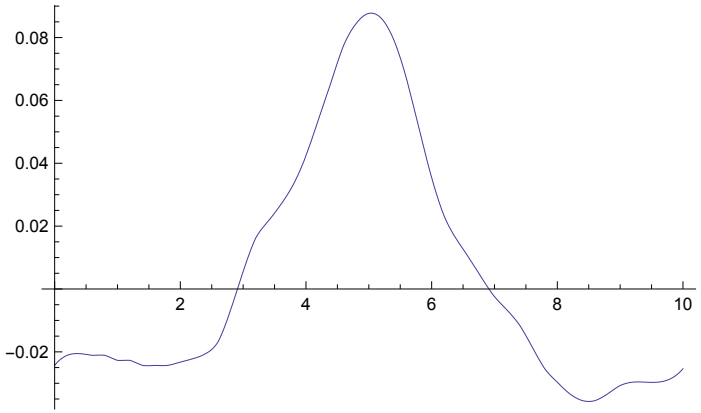



```
m = Mean[tenpercen2 - tenpercen1]; rm = RootMeanSquare[(tenpercen2 - tenpercen1) - m] / Sqrt[2]
```

```
0.0985594
```

```
Plot[f[x], {x, 0, 10}]
```

InterpolatingFunction ::dmval : Input value {0.000204286 } lies outside the range of data in the interpolating function . Extrapolation will be used. ≫



```
Print[a]
```

```
{x → 2.90981}
```

```
a /. Rule → List
```

```
{{x, 2.90981}}
```

```
Flatten[{{x, 2.45305}}]
```

```
{x, 2.45305}
```

```
%[[2]]
```

```
2.45305
```

End of Notebook so far