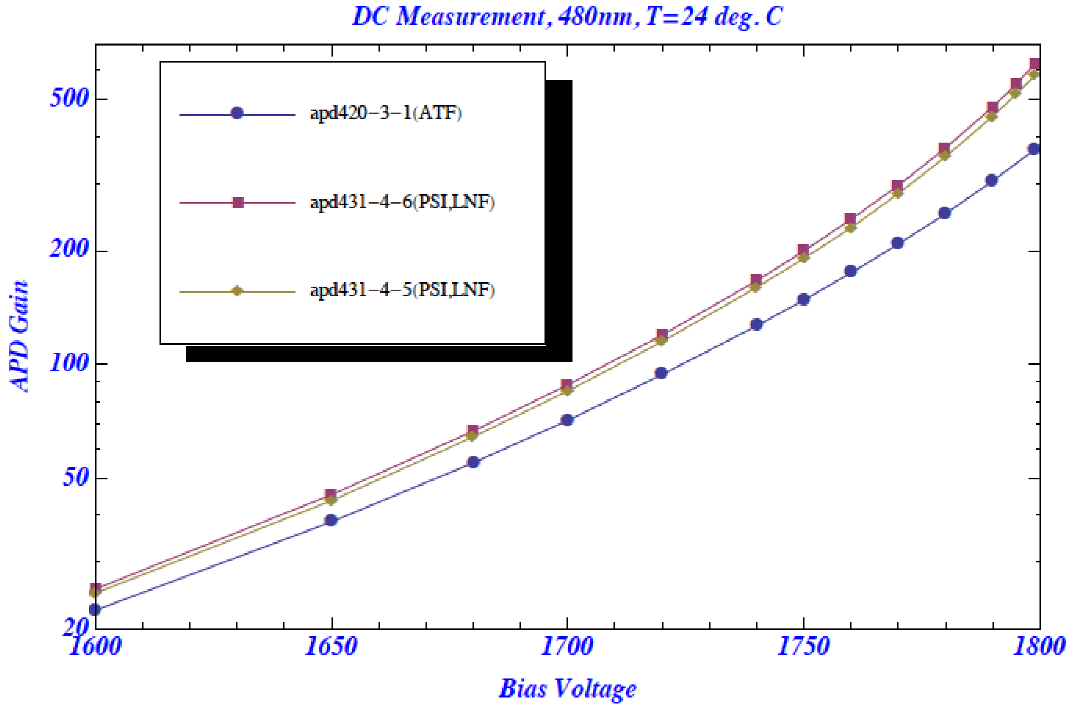
Analysis of CERN testbench data with Vcsel output split to 2 APDs. Scope inputs are:1)Inverse of pulser output into Vcsel 2)front 64mm^2 APD,3)back 64mm^2 APD,4)trigger to pulser and scope.
preamps are Wenteq with a nominal voltage gain of 100. DAQ is a Lecroy 1GHz scope sampling at 5 GSa/s. APD bias voltage was 1800 volts.
SNW-Sept. 14, 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 bit over a factor of 5). Coincidentally, this data set has about the same aplitude as MIPs did in DESY data in the 2*2mm^2 APD.

In[1]:=



In[2]:=
```
ehpairs = 90;
thick = 40;
G_APD = 530.;
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[5]= 0.0061056

Conclusion : We should expect an input signal with peak amplitude of about 6 mV into the preamp and should take into account a factor of about 5 reduction for the capacitance.
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. The 64 mm^2 detector has an amplitude reduction

by about 1 / 4 but the shape is very different.

This means that, with a 1 nsec risetime (note though, that in the present Vcsel data the risetime is ~3nsec) signal the preamp noise at the output, in order to have less than 20 picosecond contribution to the time jitter, should be less than 3 mV- ie :

In[6]:=
```
G_AMP = 100.; SNR = 50.;
```

In[7]:=
```
Noise_Target = v_peak * G_AMP / 4. / SNR
```

Out[7]= 0.0030528

Below we find that we are close to that, (8mV), even before the data with least count reduced by a factor of 20.

In[8]:=
```
SetDirectory["~bastian/Desktop/cernvcsel_wenteq_1800v/"];
Namelist = FileNames[];
Namelist // Length;
nfiles = %;
```

In[12]:=
```
filename = Namelist[[2]]
scopedata = Import[filename, "csv"];
```

Out[12]= LeCroy-RTW-2014-09-10-001.csv

In[14]:=
```
Dimensions[scopedata]
```

Out[14]= {43 540, 505}

In[15]:=
```
(*trace=Take[scopedata[[2]],-502];
baselinenoise=Take[trace,200];
meanbaseline=Mean[baselinenoise];
baseline0=baselinenoise-meanbaseline;
ListPlot[MovingAverage[Take[Abs[Fourier[baseline0]],200],2]]*)
```

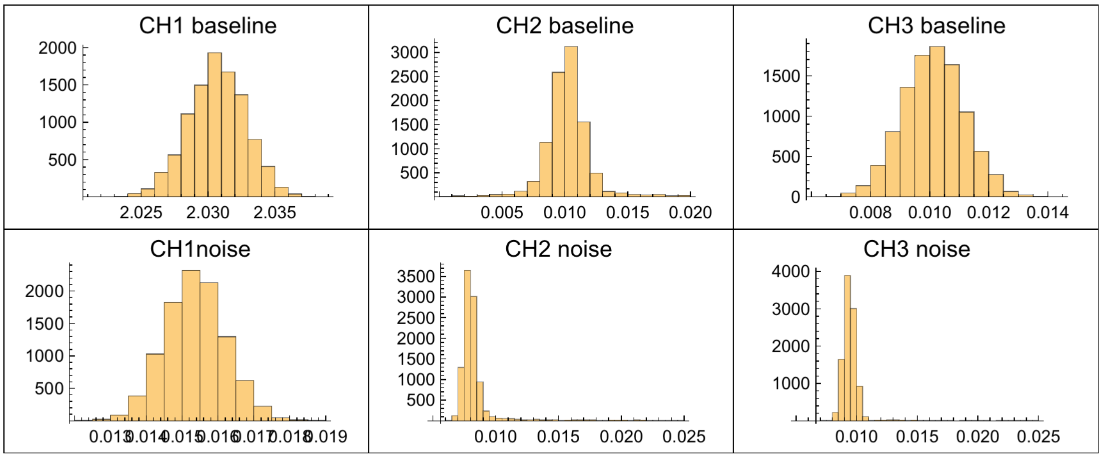
In[16]:=
```
t0 = scopedata[[1, 2]];
dt = scopedata[[1, 3]] * 10^9
time = Range[0, 501] * dt;
```
Out[17]= 0.2

In[19]:=
```
v1 = Table[Take[scopedata[[4 * (i - 1) + 1]], -502], {i, 10000}];
v2 = Table[Take[scopedata[[4 * (i - 1) + 2]], -502], {i, 10000}];
v3 = Table[Take[scopedata[[4 * (i - 1) + 3]], -502], {i, 10000}];
v4 = Table[Take[scopedata[[4 * (i - 1) + 4]], -502], {i, 10000}];
```

"iorder" specifies level of filtering. ie iorder=5 reduces 5 GSa/s to 1 GSa/s average of 5 pts.

In[23]:=
```
inbase1 = Table[Mean[Take[v1[[i]], 200]], {i, 10000}];
innoise1 =
  Table[RootMeanSquare[Take[v1[[i]], 200] - inbase1[[i]]], {i, 10000}];
inbase2 = Table[Mean[Take[v2[[i]], 200]], {i, 10000}];
innoise2 =
  Table[RootMeanSquare[Take[v2[[i]], 200] - inbase2[[i]]], {i, 10000}];
inbase3 = Table[Mean[Take[v3[[i]], 200]], {i, 10000}];
innoise3 = Table[RootMeanSquare[Take[v3[[i]], 200] - inbase3[[i]]], {i, 10000}];
```

In[29]:=
```
GraphicsGrid[{{Histogram[inbase1, PlotLabel → "CH1 baseline"],
   Histogram[inbase2, {0, .02, .001}, PlotLabel → "CH2 baseline"],
   Histogram[inbase3, PlotLabel → "CH3 baseline"]},
  {Histogram[innoise1, PlotLabel → "CH1noise"],
   Histogram[innoise2, {0.005, .025, .0005}, PlotLabel → "CH2 noise"],
   Histogram[innoise3, {0.005, .025, .0005}, PlotLabel → "CH3 noise"]}},
 Frame → All, ImageSize → Large]
```

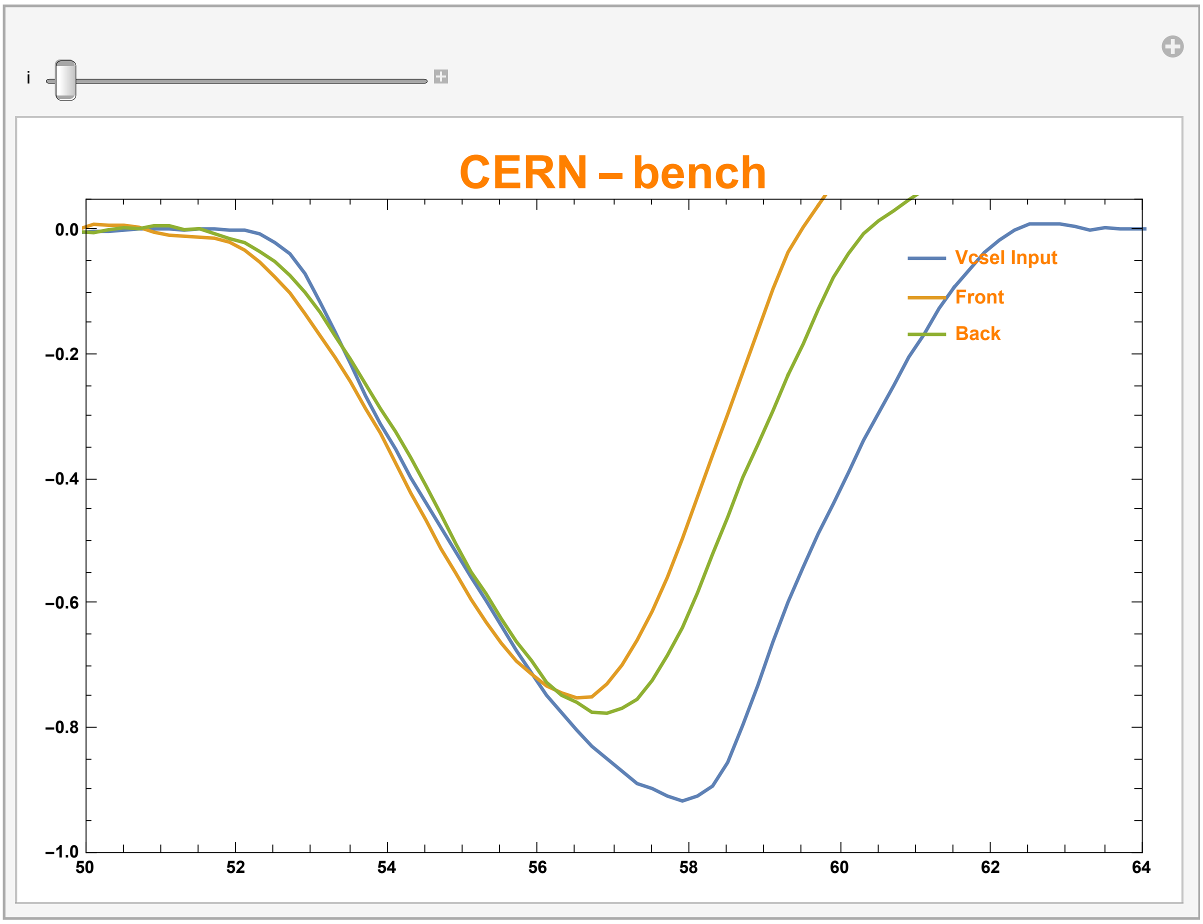
Out[29]=


In[30]:=
```
Print[Style[{" noise1= ", Mean[innoise1], " noise2= ",
   Mean[innoise2], " noise3= ", Mean[innoise3], " volts"}, Large, Red]]
```

{ noise1= , 0.0153645, noise2= , 0.0085075, noise3= , 0.00952091, volts}

In[31]:=
```
time = Range[0, 501] * dt;
iorder = 4;
Manipulate[
 ListPlot[{Transpose[{MovingAverage[time + 1.8, iorder], MovingAverage[
       0.5 * (v1[[i]] - inbase1[[i]]), iorder]}], Transpose[{MovingAverage[
       time, iorder], MovingAverage[(v2[[i]] - inbase2[[i]]), iorder]}],
   Transpose[{MovingAverage[time - .4, iorder],
     MovingAverage[(v3[[i]] - inbase3[[i]]), iorder]}]},
  PlotRange → {{50., 64.}, {-1, .05}}, PlotStyle →
   Directive[AbsoluteThickness[1.8]],
  LabelStyle → Directive[Orange, Bold],
  PlotLabel → Style[CERN - bench, Large],
  AxesLabel → {nanosec, V}, PlotStyle → {Red, Blue, Green},
  PlotLegends → Placed[{"Vcsel Input", "Front", "Back"}, {0.85, 0.85}],
  Joined → True, ImageSize → Large, Frame → True, FrameStyle → Black],
 {i, 1, 10, 1}, SaveDefinitions → True]
```

Out[33]=



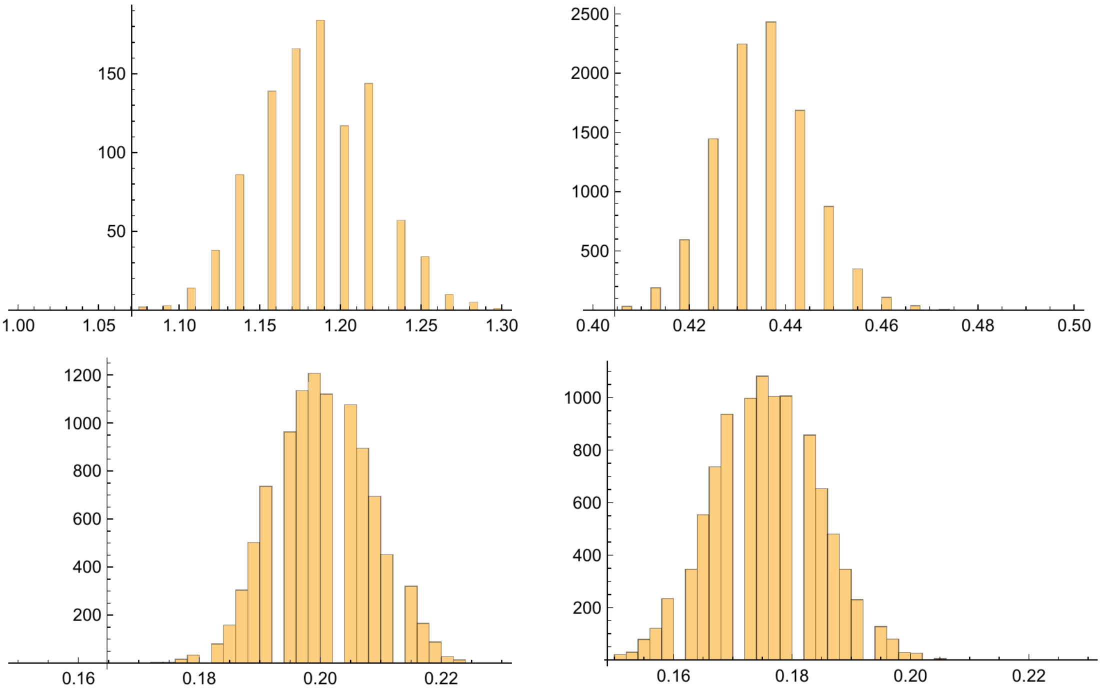
Leading edge slopes

In[34]:=
```
timbin[t_] := IntegerPart[t / dt];
t1 = timbin[53.]; t2 = timbin[55.6];
tdiff = 55.6 - 53.
t11 = timbin[53. - 1.5];
t21 = timbin[55.6 - 1.5];
```

Out[36]= 2.6

In[38]:=
```
ampt1 = Table[v1[[i, t1]], {i, 1, 1000}];
slope1 = Table[- (v1[[i, t21]] - v1[[i, t11]]) / tdiff, {i, 1, 10 000}];
slope2 = Table[- (v2[[i, t2]] - v2[[i, t1]]) / tdiff, {i, 1, 10 000}];
slope3 = Table[- (v3[[i, t2]] - v3[[i, t1]]) / tdiff, {i, 1, 10 000}];
GraphicsGrid[
 {{Histogram[ampt1, {1., 1.3, .005}], Histogram[slope1, {0.4, 0.5, .002}]},
  {Histogram[slope2, {.15, .23, .002}],
   Histogram[slope3, {.15, .23, .002}]}}, ImageSize → Large]
```

Out[42]=



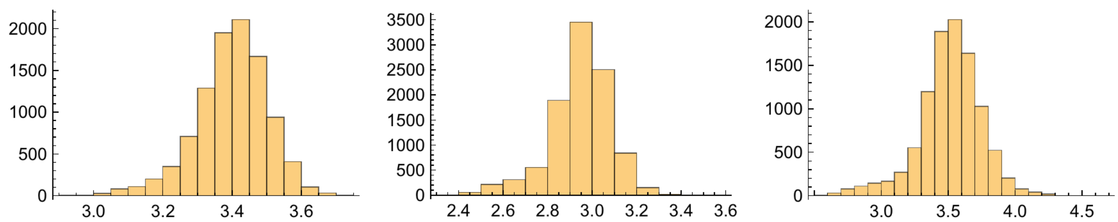
Now look at peak amplitude on all 3 APDs expecting to see a Landau distribution.

In[43]:=
```
Clear[peak1, peak2, peak3];
peak1 = Table[Max[Take[- (v1[[i]] - inbase1[[i]]), {240, 300}]], {i, 10 000}];
peak2 = Table[Max[Take[- (v2[[i]] - inbase2[[i]]), {240, 300}]], {i, 10 000}];
peak3 = Table[Max[Take[- (v3[[i]] - inbase3[[i]]), {240, 300}]], {i, 10 000}];
```

Convert Slopes into 10 - 90 % risetime = 0.8*V_ {peak}/slope

In[47]:=
```
trise1 = Table[(0.8 * peak1[[i]] / slope1[[i]]), {i, 1, 10 000}];
trise2 = Table[(0.8 * peak2[[i]] / slope2[[i]]), {i, 1, 10 000}];
trise3 = Table[(0.8 * peak3[[i]] / slope3[[i]]), {i, 1, 10 000}];
GraphicsRow[
 {Histogram[trise1], Histogram[trise2], Histogram[trise3]}, ImageSize → Large]
```

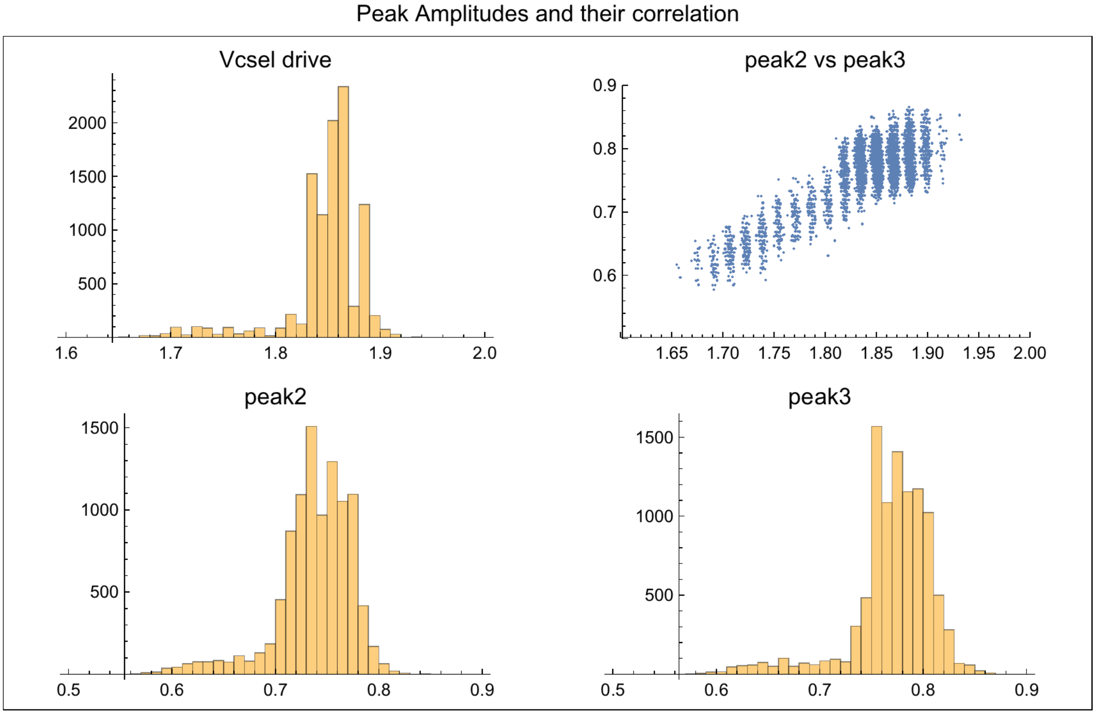
Out[50]=



In[51]:=
```
area3 = Table[Sum[-(v3[[i, j]] - inbase3[[i]]), {j, 240, 300}], {i, 10 000}];
```

In[52]:=
```
GraphicsGrid[{{Histogram[peak1, {1.6, 2, .01}, PlotLabel → "Vcsel drive"],
    ListPlot[Transpose[{peak1, peak3}],
     PlotRange → {{1.6, 2.0}, {0.5, 0.9}}, PlotLabel → "peak2 vs peak3"]},
   {Histogram[peak2, {.5, .9, .01}, PlotLabel → "peak2"],
    Histogram[peak3, {.5, .9, .01}, PlotLabel → "peak3"]}}, Frame → True,
  PlotLabel → "Peak Amplitudes and their correlation", ImageSize → Large]
```

Out[52]=



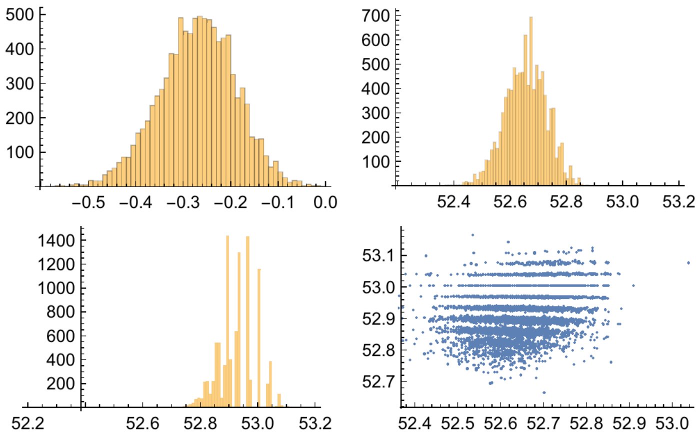
Simple algorithm that gives mediocre results:
(ti-t0)*dV/dt=(V(ti))   ->t0=ti-(V(ti))/dV/dt

In[53]:=
```
t02 = Table[(53. + v2[[i, t1]] / slope2[[i]]), {i, 1, 10 000}];
t03 = Table[(53. + v3[[i, t1]] / slope3[[i]]), {i, 1, 10 000}];
deltat = t02 - t03;
md = Mean[deltat]; m2 = Mean[t02]; m3 = Mean[t03];
Print[md, "    ", m2, "    ", m3]
Print[RootMeanSquare[deltat - md], "    ",
 RootMeanSquare[t02 - m2], "   ", RootMeanSquare[t03 - m3]]
GraphicsGrid[
 {{Histogram[deltat, {-.6, 0., .01}], Histogram[t02, {52.2, 53.2, .01}]},
  {Histogram[t03, {52.2, 53.2, .01}], ListPlot[Transpose[{t02, t03}]]}}]
```

```
-0.270611   52.655   52.9256

0.0843113   0.0761965   0.0663742
```

Out[59]=



Now try to improve things by fitting the leading edge over the same time range. Also adjust fixed phase differences
between channels and scale to simplify fitting. Phases of ch1, 2, 3 are - 1.8, 0., +.4 nsec. Vcsel input is ~2 times amplifier outputs. Polarity of signals is negative. Vcsel input is ~1/.411 largerthan amp signals. We highlight the region from 50 to 60 nsec-> 250<i<300:

In[60]:=
```
sig1 = Table[Take[-0.5 (v1[[i]] - inbase1[[i]]), {242, 291}], {i, 1, 10000}];
sig2 = Table[Take[- (v2[[i]] - inbase2[[i]]), {251, 300}], {i, 1, 10000}];
sig3 = Table[Take[- (v3[[i]] - inbase3[[i]]), {253, 302}], {i, 1, 10000}];
Clear[timecut];
timecut = Range[251, 300] * dt;

(*iorder=4;
Manipulate[ListPlot[
  {Transpose[{MovingAverage[timecut,iorder],MovingAverage[sig1[[i]],iorder]}],
   Transpose[{MovingAverage[timecut,iorder],MovingAverage[sig2[[i]],iorder]}],
   Transpose[{MovingAverage[timecut,iorder],MovingAverage[sig3[[i]],iorder]}]},
  PlotRange→{{50.,60.},{-.1,1}},PlotStyle→Directive[AbsoluteThickness[1.8]],
  LabelStyle→Directive[Orange,Bold],PlotLabel→Style[CERN-bench,Large],
  AxesLabel→{nsec,v},PlotStyle→{Red,Blue,Green},
  PlotLegends→Placed[{"Vcsel Input","Front","Back"},{0.85,0.85}],
  Joined→True,ImageSize→Large,Frame→True,FrameStyle→Black],
 {i,1,200,1},SaveDefinitions→True]*)
```

We now fit with a linear fit, the regon of interest.

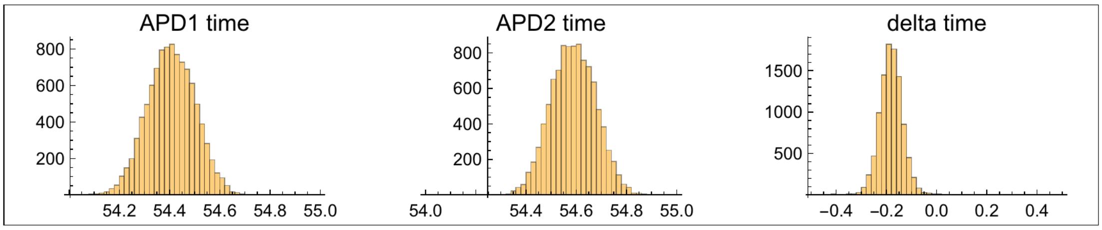
In[67]:=
```
timefit = Range[268, 279] * dt;
points1 =
  Table[Transpose[{timefit, Take[sig1[[i]], {18, 29}]}], {i, 1, 10000}];
points2 = Table[Transpose[{timefit, Take[sig2[[i]], {18, 29}]}], {i, 1, 10000}];
points3 = Table[Transpose[{timefit, Take[sig3[[i]], {18, 29}]}], {i, 1, 10000}];

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

In[193]:=
```
trig1 = ConstantArray[0, 10000];
trig2 = ConstantArray[0, 10000];
trig3 = ConstantArray[0, 10000];
ithr = 2; timethreshold = {.25, .35, .45, .55};
Do[
  Clear[fun]; Clear[g];
  fun[x_] := Evaluate[Fit[points1[[i]], {1, x}, x]];
g = Evaluate[InverseFunction[fun]];
  trig1[[i]] = Evaluate[g[timethreshold[[ithr]]]];
  , {i, 10000}];
Do[
  Clear[fun]; Clear[g];
  fun[x_] := Evaluate[Fit[points2[[i]], {1, x}, x]];
g = Evaluate[InverseFunction[fun]];
  trig2[[i]] = Evaluate[g[timethreshold[[ithr]]]];
  , {i, 10000}];
Do[
  Clear[fun]; Clear[g];
  fun[x_] := Evaluate[Fit[points3[[i]], {1, x}, x]];
g = Evaluate[InverseFunction[fun]];
  trig3[[i]] = Evaluate[g[timethreshold[[ithr]]]];
  , {i, 10000}];
```

In[198]:=
```
GraphicsRow[{Histogram[trig2, {54., 55., .02}, PlotLabel → "APD1 time"],
  Histogram[trig3, {54., 55., .02}, PlotLabel → "APD2 time"],
  Histogram[(trig2 - trig3), {-.5, .5, .02}, PlotLabel → "delta time"]},
 ImageSize → Large, Frame → True]
```

Out[198]=



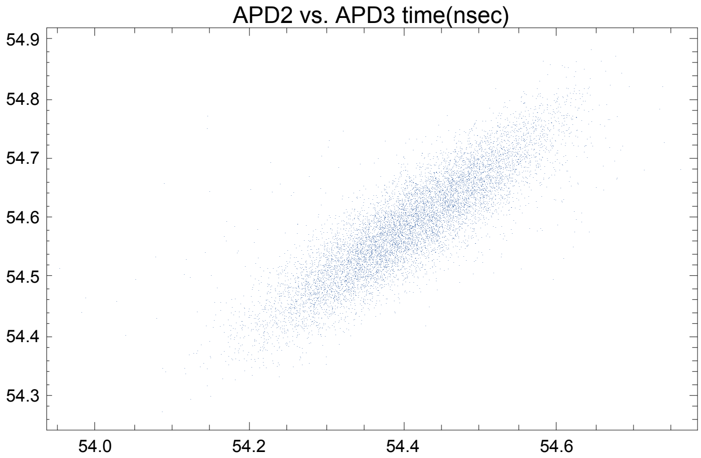
In[199]:=
```
mtrig = Mean[(trig2 - trig3)]
jitter = RootMeanSquare[(trig2 - trig3 - mtrig)]
difftr[[ithr]] = jitter;
```

Out[199]= -0.181454

Out[200]= 0.048518

In[202]:=
```
ListPlot[Transpose[{trig2, trig3}],
 PlotLabel → "APD2 vs. APD3 time(nsec)", Frame → True]
```

Out[202]=

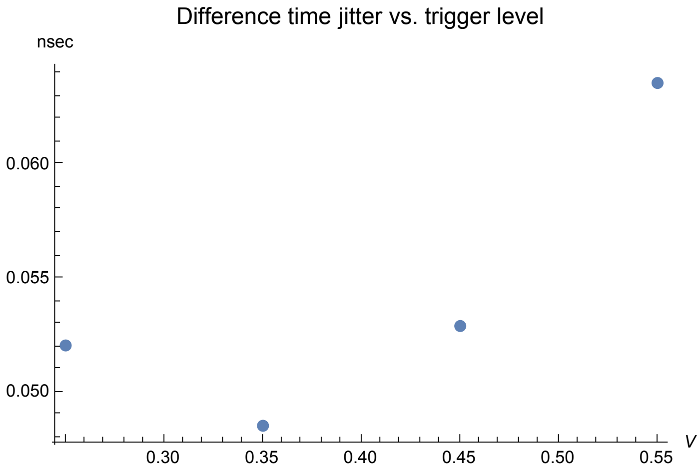


In[188]:=
```
difftr
```

Out[188]=
```
{0.0520504, 0.048518, 0.0529053, 0.0635937}
```

In[203]:=
```
ListPlot[Transpose[{timethreshold, difftr}],
 PlotLabel → "Difference time jitter vs. trigger level", AxesLabel → {V, nsec}]
```

Out[203]=



End of Notebook so far