Latest analysis as a function of position:

3.10.2006

Definition of "good" event:

```
// Good hit in lead glass:
bool good_electron_hit=false;
    if (adc3_5 < 240. && adc3_5 > 160.) good_electron_hit=true;
// Good hit in the Quartz Start counter 1:
    bool good_start1_hit=false;
    if (adc3_2 > 250. && adc3_3 > 250.) good_start1_hit=true;
// Good hit in the scintillator Start counter 2;
    bool good_start2_hit=false;
    if (adc3_6 > 60. && adc3_7 > 60. && adc3_8 > 60. && adc3_9 > 60.) good_start2_hit=true;
//Good hit:
    bool good_hit=false;
    if (single_hit && good_electron_hit && good_start1_hit && good_start2_hit) good_hit=true;
```

Start 1 counter software:

```
// *************************************************************************************************
// Quartz Start counter #1 - correct for ADC, hodoscope position, temperature drift, etc.
// *************************************************************************************************
    double tdc0_0_corr, tdc0_1_corr, tdc0_2_corr, tdc0_3_corr;
    double tdc_start1_ave, tdc_start1_ave_100, time_walk_mean;
// ADC correction:
    tdc0_0_corr=1861.0 + tdc0_0 - (1894.0 - 0.05346*adc3_0 + 0.00001731*adc3_0*adc3_0 + 2.267*pow(10.,18.)/pow(adc3_0,7));
    tdc0_0_corr=tdc0_0_corr*22.57/1000;
    tdc0_1_corr=1861.0 + tdc0_1 - (1907.0 - 0.05803*adc3_1 + 0.00001355*adc3_1*adc3_1 + 1.39*pow(10.,17.)/pow(adc3_1,7));
    tdc0_1_corr=tdc0_1_corr*22.57/1000;
    tdc0_2_corr=1861.0 + tdc0_2 - (1835.0 - 0.02653*adc3_2 + 0.000006678*adc3_2*adc3_2 + 7.32*pow(10.,17.)/pow(adc3_2,7));
    tdc0_2_corr=tdc0_2_corr*22.57/1000;
    tdc0_3_corr=1861.0 + tdc0_3 - (1867.0 - 0.02838*adc3_3 + 0.000007132*adc3_3*adc3_3 + 3.745*pow(10.,18.)/pow(adc3_3,7));
    tdc0_3_corr=tdc0_3_corr*22.57/1000;
//Hodoscope position correction:
    tdc0_0_corr= 42.0 + tdc0_0_corr - (42.01 + 0.005358*z_coord);
    tdc0_1_corr= 42.0 + tdc0_1_corr - (42.02 + 0.00955*z_coord);
    tdc0_2_corr= 42.0 + tdc0_2_corr - (42.01 + 0.005282*z_coord);
    tdc0_3_corr= 42.0 + tdc0_3_corr - (41.99 + 0.004585*z_coord);
// Temperature drift correction using timing marker_1:
    //tdc0_0_corr= tdc0_0_corr - (t_marker_1 - 36.032);
//Determine Average of two best pads in the start 1 counter
    tdc_start1_ave = (tdc0_2_corr + tdc0_3_corr)/2.0;
//time_walk_mean =  average of 100 good hit "start times" in the start 1 counter
    if(n_trig == 1) tdc_start1_ave_100 = 0.0;
    if (good_hit)
  {
    n_good_hit = n_good_hit + 1;
    n_tick = n_tick + 1;
    if (n_good_hit < 100) time_walk_mean = tdc_start1_ave;
    tdc_start1_ave_100 = tdc_start1_ave_100 + tdc_start1_ave;
    if (n_tick == 100)
      {
        time_walk_mean = tdc_start1_ave_100/100.0;
        tdc_start1_ave_100 = 0.0;
        n_tick = 0;
      }
  }
```

Start 2 counter software:

```
// *************************************************************************************************
// Scint.  Start counter #2 - correct for ADC, hodoscope position, temperature drift, etc.
// *************************************************************************************************
    double tdc0_6_corr, tdc0_7_corr, tdc0_8_corr, tdc0_9_corr, tdc_start2_ave;
// ADC correction:
    tdc0_6_corr=1861.0 + tdc0_6 - (1857.0 - 0.03483*adc3_6 + 0.00000001281*adc3_6*adc3_6*adc3_6 + 1896./adc3_6);
    tdc0_6_corr=tdc0_6_corr*22.57/1000;
```

```
    tdc0_7_corr=1861.0 + tdc0_7 - (1873.0-0.0185*adc3_7 + 0.000000003402*adc3_7*adc3_7*adc3_7 + 2780./adc3_7);
    tdc0_7_corr=tdc0_7_corr*22.57/1000;
    tdc0_8_corr=1861.0 + tdc0_8 - (1877.0-0.01325*adc3_8 + 0.00000000157*adc3_8*adc3_8*adc3_8 + 3137./adc3_8);
    tdc0_8_corr=tdc0_8_corr*22.57/1000;
    tdc0_9_corr=1861.0 + tdc0_9 - (1890.0-0.02194*adc3_9 - 0.0000001112*adc3_9*adc3_9*adc3_9 + 1832./adc3_9);
    tdc0_9_corr=tdc0_9_corr*22.57/1000;
//Hodoscope position correction:
    tdc0_6_corr= 42.0 + tdc0_6_corr - (41.99 + 0.007332*z_coord);
    tdc0_7_corr= 42.0 + tdc0_7_corr - (41.99 + 0.006726*z_coord);
    tdc0_8_corr= 42.0 + tdc0_8_corr - (41.99 + 0.007201*z_coord);
    tdc0_9_corr= 42.0 + tdc0_9_corr - (41.98 + 0.007003*z_coord);
// Temperature drifts using timing marker_1:
    //tdc0_6_corr=tdc0_6_corr-(t_marker_1-36.032);
//Average of four pads
    tdc_start2_ave = (tdc0_6_corr + tdc0_7_corr + tdc0_8_corr + tdc0_9_corr)/4.0;
```

Cross-talk:

```
// *************************************************************************************************************
// Kill the prototype cross-talk hits; apply this cut only for the 2nd peak as the 1st peak photons are too close to each other in time//
//************************************************************************************************************
//
    Float_t time_diff, limit_high, limit_low, tdc_i, tdc_k, cher_i, cher_k, diff_cher;

    limit_low = 0.05;
    limit_high = 0.5;

    // Slot 2
    for (Int_t i=1; i<65;i++)
    for (Int_t k=1; k<65;k++)
    {
  if (good_hit && (tdc_slot2_phillips[i] > low_cut_2nd) && (tdc_slot2_phillips[i] < high_cut_2nd))
   {
    tdc_i = tdc_slot2_phillips[i] - time_offset_slot2_phillips_peak_2[i];
    tdc_k = tdc_slot2_phillips[k] - time_offset_slot2_phillips_peak_2[k];
    if ((good_hit) && (i != k) && ((tdc_i != -1) || (tdc_k != -1)))
    {
      time_diff = fabs(tdc_i-tdc_k);
      if ((tdc_i > tdc_k) && (time_diff < limit_high) && (time_diff > limit_low))
        {
          tdc_k = -1;
          tdc_slot2_phillips[k] = -1;
        }
    }
   }
    }

    // Slot 3
    for (Int_t i=1; i<65;i++)
    for (Int_t k=1; k<65;k++)
    {
  if (good_hit && (tdc_slot3_phillips[i] > low_cut_2nd) && (tdc_slot3_phillips[i] < high_cut_2nd))
   {
    tdc_i = tdc_slot3_phillips[i] - time_offset_slot3_phillips_peak_2[i];
    tdc_k = tdc_slot3_phillips[k] - time_offset_slot3_phillips_peak_2[k];
    if ((good_hit) && (i != k) && ((tdc_i != -1) || (tdc_k != -1)))
    {
      time_diff = fabs(tdc_i-tdc_k);
      if ((tdc_i > tdc_k) && (time_diff < limit_high) && (time_diff > limit_low))
        {
          tdc_k = -1;
          tdc_slot3_phillips[k] = -1;
        }
    }
   }
    }

    // Slot 4
    for (Int_t i=1; i<65;i++)
    for (Int_t k=1; k<65;k++)
```

```
 {
if (good_hit && (tdc_slot4_phillips[i] > low_cut_2nd) && (tdc_slot4_phillips[i] < high_cut_2nd))
 {
  tdc_i = tdc_slot4_phillips[i] - time_offset_slot4_phillips_peak_2[i];
  tdc_k = tdc_slot4_phillips[k] - time_offset_slot4_phillips_peak_2[k];
  if ((good_hit) && (i != k) && ((tdc_i != -1) || (tdc_k != -1)))
  {
   time_diff = fabs(tdc_i-tdc_k);
   if ((tdc_i > tdc_k) && (time_diff < limit_high) && (time_diff > limit_low))
     {
      tdc_k = -1;
      tdc_slot4_phillips[k] = -1;
    }
  }
 }
 }

 // Slot 5
 for (Int_t i=1; i<65;i++)
 for (Int_t k=1; k<65;k++)
 {
if (good_hit && (tdc_slot5_phillips[i] > low_cut_2nd) && (tdc_slot5_phillips[i] < high_cut_2nd))
 {
  tdc_i = tdc_slot5_phillips[i] - time_offset_slot5_phillips_peak_2[i];
  tdc_k = tdc_slot5_phillips[k] - time_offset_slot5_phillips_peak_2[k];
  if ((good_hit) && (i != k) && ((tdc_i != -1) || (tdc_k != -1)))
  {
   time_diff = fabs(tdc_i-tdc_k);
   if ((tdc_i > tdc_k) && (time_diff < limit_high) && (time_diff > limit_low))
     {
      tdc_k = -1;
      tdc_slot5_phillips[k] = -1;
    }
  }
 }
 }

 // Slot 6
 for (Int_t i=1; i<65;i++)
 for (Int_t k=1; k<65;k++)
 {
if (good_hit && (tdc_slot6_phillips[i] > low_cut_2nd) && (tdc_slot6_phillips[i] < high_cut_2nd))
 {
  tdc_i = tdc_slot6_phillips[i] - time_offset_slot6_phillips_peak_2[i];
  tdc_k = tdc_slot6_phillips[k] - time_offset_slot6_phillips_peak_2[k];
  if ((good_hit) && (i != k) && ((tdc_i != -1) || (tdc_k != -1)))
  {
   time_diff = fabs(tdc_i-tdc_k);
   if ((tdc_i > tdc_k) && (time_diff < limit_high) && (time_diff > limit_low))
     {
      tdc_k = -1;
      tdc_slot6_phillips[k] = -1;
    }
  }
 }
 }
```

Charge sharing:

```
// ********************************************************************************************************
 // Take care of charge sharing (a hit is close to boundary between two pads) - 1st peak first:
 // ********************************************************************************************************
 // Does not work at the moment - switch it off
 //

 Float_t limit_epsilon = 0.2;

// Slot 2
 for (Int_t i=1; i<65;i++)
 for (Int_t k=1; k<65;k++)
```

```
 {
if ((good_hit) && (tdc_slot2_phillips[i] > low_cut_1st) && (tdc_slot2_phillips[i] < high_cut_1st))
 {
  tdc_i = tdc_slot2_phillips[i] - time_offset_slot2_phillips_peak_1[i];
  tdc_k = tdc_slot2_phillips[k] - time_offset_slot2_phillips_peak_1[k];
  time_diff = fabs(tdc_i-tdc_k);
  if ((good_hit) && (i != k) && ((tdc_i != -1) || (tdc_k != -1)) && (time_diff < limit_epsilon))
  {
   cher_i = cherenkov_angle_slot2_corr[i];
   cher_k = cherenkov_angle_slot2_corr[k];
   diff_cher = fabs(cher_i - cher_k);
   if (diff_cher < 1.0) tdc_k = -1;
   if (diff_cher < 1.0) tdc_slot2_phillips[k] = -1;
   //if (diff_cher < 1.0) cherenkov_angle_slot2_corr[i] = (cher_i + cher_k)/2.0;
   //if (diff_cher < 1.0) path1_slot2[i] = (path1_slot2[i]+path1_slot2[k])/2.0;
  }
 }
 }

 // Slot 3
 for (Int_t i=1; i<65;i++)
 for (Int_t k=1; k<65;k++)
 {
if ((good_hit) && (tdc_slot3_phillips[i] > low_cut_1st) && (tdc_slot3_phillips[i] < high_cut_1st))
 {
  tdc_i = tdc_slot3_phillips[i] - time_offset_slot3_phillips_peak_1[i];
  tdc_k = tdc_slot3_phillips[k] - time_offset_slot3_phillips_peak_1[k];
  time_diff = fabs(tdc_i-tdc_k);
  if ((good_hit) && (i != k) && ((tdc_i != -1) || (tdc_k != -1)) && (time_diff < limit_epsilon))
  {
   cher_i = cherenkov_angle_slot3_corr[i];
   cher_k = cherenkov_angle_slot3_corr[k];
   diff_cher = fabs(cher_i - cher_k);
   if (diff_cher < 1.0) tdc_k = -1;
   if (diff_cher < 1.0) tdc_slot3_phillips[k] = -1;
   //if (diff_cher < 1.0) cherenkov_angle_slot3_corr[i] = (cher_i + cher_k)/2.0;
   //if (diff_cher < 1.0) path1_slot3[i] = (path1_slot3[i]+path1_slot3[k])/2.0;
  }
 }
 }

 // Slot 4
 for (Int_t i=1; i<65;i++)
 for (Int_t k=1; k<65;k++)
 {
if ((good_hit) && (tdc_slot4_phillips[i] > low_cut_1st) && (tdc_slot4_phillips[i] < high_cut_1st))
 {
  tdc_i = tdc_slot4_phillips[i] - time_offset_slot4_phillips_peak_1[i];
  tdc_k = tdc_slot4_phillips[k] - time_offset_slot4_phillips_peak_1[k];
  time_diff = fabs(tdc_i-tdc_k);
  if ((good_hit) && (i != k) && ((tdc_i != -1) || (tdc_k != -1)) && (time_diff < limit_epsilon))
  {
   cher_i = cherenkov_angle_slot4_corr[i];
   cher_k = cherenkov_angle_slot4_corr[k];
   diff_cher = fabs(cher_i - cher_k);
   if (diff_cher < 1.0) tdc_k = -1;
   if (diff_cher < 1.0) tdc_slot4_phillips[k] = -1;
   //if (diff_cher < 1.0) cherenkov_angle_slot4_corr[i] = (cher_i + cher_k)/2.0;
   //if (diff_cher < 1.0) path1_slot4[i] = (path1_slot4[i]+path1_slot4[k])/2.0;
  }
 }
 }

 // Slot 5
 for (Int_t i=1; i<65;i++)
 for (Int_t k=1; k<65;k++)
 {
if ((good_hit) && (tdc_slot5_phillips[i] > low_cut_1st) && (tdc_slot5_phillips[i] < high_cut_1st))
 {
  tdc_i = tdc_slot5_phillips[i] - time_offset_slot5_phillips_peak_1[i];
```

```
   tdc_k = tdc_slot5_phillips[k] - time_offset_slot5_phillips_peak_1[k];
   time_diff = fabs(tdc_i-tdc_k);
   if ((good_hit) && (i != k) && ((tdc_i != -1) || (tdc_k != -1)) && (time_diff < limit_epsilon))
   {
    cher_i = cherenkov_angle_slot5_corr[i];
    cher_k = cherenkov_angle_slot5_corr[k];
    diff_cher = fabs(cher_i - cher_k);
    if (diff_cher < 1.0) tdc_k = -1;
    if (diff_cher < 1.0) tdc_slot5_phillips[k] = -1;
    //if (diff_cher < 1.0) cherenkov_angle_slot5_corr[i] = (cher_i + cher_k)/2.0;
    //if (diff_cher < 1.0) path1_slot5[i] = (path1_slot5[i]+path1_slot5[k])/2.0;
   }
  }
  }

  // Slot 6
  for (Int_t i=1; i<65;i++)
  for (Int_t k=1; k<65;k++)
  {
if ((good_hit) && (tdc_slot6_phillips[i] > low_cut_1st) && (tdc_slot6_phillips[i] < high_cut_1st))
  {
   tdc_i = tdc_slot6_phillips[i] - time_offset_slot6_phillips_peak_1[i];
   tdc_k = tdc_slot6_phillips[k] - time_offset_slot6_phillips_peak_1[k];
   time_diff = fabs(tdc_i-tdc_k);
   if ((good_hit) && (i != k) && ((tdc_i != -1) || (tdc_k != -1)) && (time_diff < limit_epsilon))
   {
    cher_i = cherenkov_angle_slot6_corr[i];
    cher_k = cherenkov_angle_slot6_corr[k];
    diff_cher = fabs(cher_i - cher_k);
    if (diff_cher < 1.0) tdc_k = -1;
    if (diff_cher < 1.0) tdc_slot6_phillips[k] = -1;
    //if (diff_cher < 1.0) cherenkov_angle_slot6_corr[i] = (cher_i + cher_k)/2.0;
    //if (diff_cher < 1.0) path1_slot6[i] = (path1_slot6[i]+path1_slot6[k])/2.0;
   }
  }
  }

  // ****************************************************************************************************
  // Take care of charge sharing (a hit is close to boundary between two pads) - 2nd peak:
  // ****************************************************************************************************

  // Slot 2
  for (Int_t i=1; i<65;i++)
  for (Int_t k=1; k<65;k++)
  {
if ((good_hit) && (tdc_slot2_phillips[i] > low_cut_2nd) && (tdc_slot2_phillips[i] < high_cut_2nd))
  {
   tdc_i = tdc_slot2_phillips[i] - time_offset_slot2_phillips_peak_2[i];
   tdc_k = tdc_slot2_phillips[k] - time_offset_slot2_phillips_peak_2[k];
   time_diff = fabs(tdc_i-tdc_k);
   if ((good_hit) && (i != k) && ((tdc_i != -1) || (tdc_k != -1)) && (time_diff < limit_epsilon))
   {
    cher_i = cherenkov_angle_slot2_corr[i];
    cher_k = cherenkov_angle_slot2_corr[k];
    diff_cher = fabs(cher_i - cher_k);
    if (diff_cher < 1.0) tdc_k = -1;
    if (diff_cher < 1.0) tdc_slot2_phillips[k] = -1;
    //if (diff_cher < 1.0) cherenkov_angle_slot2_corr[i] = (cher_i + cher_k)/2.0;
    //if (diff_cher < 1.0) path2_slot2[i] = (path2_slot2[i]+path2_slot2[k])/2.0;
   }
  }
  }

  // Slot 3
  for (Int_t i=1; i<65;i++)
  for (Int_t k=1; k<65;k++)
  {
if ((good_hit) && (tdc_slot3_phillips[i] > low_cut_2nd) && (tdc_slot3_phillips[i] < high_cut_2nd))
  {
   tdc_i = tdc_slot3_phillips[i] - time_offset_slot3_phillips_peak_2[i];
```

```
   tdc_k = tdc_slot3_phillips[k] - time_offset_slot3_phillips_peak_2[k];
   time_diff = fabs(tdc_i-tdc_k);
   if ((good_hit) && (i != k) && ((tdc_i != -1) || (tdc_k != -1)) && (time_diff < limit_epsilon))
   {
    cher_i = cherenkov_angle_slot3_corr[i];
    cher_k = cherenkov_angle_slot3_corr[k];
    diff_cher = fabs(cher_i - cher_k);
    if (diff_cher < 1.0) tdc_k = -1;
    if (diff_cher < 1.0) tdc_slot3_phillips[k] = -1;
    //if (diff_cher < 1.0) cherenkov_angle_slot3_corr[i] = (cher_i + cher_k)/2.0;
    //if (diff_cher < 1.0) path2_slot3[i] = (path2_slot3[i]+path2_slot3[k])/2.0;
   }
  }
  }

  // Slot 4
  for (Int_t i=1; i<65;i++)
  for (Int_t k=1; k<65;k++)
  {
if ((good_hit) && (tdc_slot4_phillips[i] > low_cut_2nd) && (tdc_slot4_phillips[i] < high_cut_2nd))
  {
   tdc_i = tdc_slot4_phillips[i] - time_offset_slot4_phillips_peak_2[i];
   tdc_k = tdc_slot4_phillips[k] - time_offset_slot4_phillips_peak_2[k];
   time_diff = fabs(tdc_i-tdc_k);
   if ((good_hit) && (i != k) && ((tdc_i != -1) || (tdc_k != -1)) && (time_diff < limit_epsilon))
   {
    cher_i = cherenkov_angle_slot4_corr[i];
    cher_k = cherenkov_angle_slot4_corr[k];
    diff_cher = fabs(cher_i - cher_k);
    if (diff_cher < 1.0) tdc_k = -1;
    if (diff_cher < 1.0) tdc_slot4_phillips[k] = -1;
    //if (diff_cher < 1.0) cherenkov_angle_slot4_corr[i] = (cher_i + cher_k)/2.0;
    //if (diff_cher < 1.0) path2_slot4[i] = (path2_slot4[i]+path2_slot4[k])/2.0;
   }
  }
  }

  // Slot 5
  for (Int_t i=1; i<65;i++)
  for (Int_t k=1; k<65;k++)
  {
if ((good_hit) && (tdc_slot5_phillips[i] > low_cut_2nd) && (tdc_slot5_phillips[i] < high_cut_2nd))
  {
   tdc_i = tdc_slot5_phillips[i] - time_offset_slot5_phillips_peak_2[i];
   tdc_k = tdc_slot5_phillips[k] - time_offset_slot5_phillips_peak_2[k];
   time_diff = fabs(tdc_i-tdc_k);
   if ((good_hit) && (i != k) && ((tdc_i != -1) || (tdc_k != -1)) && (time_diff < limit_epsilon))
   {
    cher_i = cherenkov_angle_slot5_corr[i];
    cher_k = cherenkov_angle_slot5_corr[k];
    diff_cher = fabs(cher_i - cher_k);
    if (diff_cher < 1.0) tdc_k = -1;
    if (diff_cher < 1.0) tdc_slot5_phillips[k] = -1;
    //if (diff_cher < 1.0) cherenkov_angle_slot5_corr[i] = (cher_i + cher_k)/2.0;
    //if (diff_cher < 1.0) path2_slot5[i] = (path2_slot5[i]+path2_slot5[k])/2.0;
   }
  }
  }

  // Slot 6
  for (Int_t i=1; i<65;i++)
  for (Int_t k=1; k<65;k++)
  {
if ((good_hit) && (tdc_slot6_phillips[i] > low_cut_2nd) && (tdc_slot6_phillips[i] < high_cut_2nd))
  {
   tdc_i = tdc_slot6_phillips[i] - time_offset_slot6_phillips_peak_2[i];
   tdc_k = tdc_slot6_phillips[k] - time_offset_slot6_phillips_peak_2[k];
   time_diff = fabs(tdc_i-tdc_k);
   if ((good_hit) && (i != k) && ((tdc_i != -1) || (tdc_k != -1)) && (time_diff < limit_epsilon))
   {
```

```
      cher_i = cherenkov_angle_slot6_corr[i];
      cher_k = cherenkov_angle_slot6_corr[k];
      diff_cher = fabs(cher_i - cher_k);
      if (diff_cher < 1.0) tdc_k = -1;
      if (diff_cher < 1.0) tdc_slot6_phillips[k] = -1;
      //if (diff_cher < 1.0) cherenkov_angle_slot6_corr[i] = (cher_i + cher_k)/2.0;
      //if (diff_cher < 1.0) path2_slot6[i] = (path2_slot6[i]+path2_slot6[k])/2.0;
     }
    }
   }

   // **********************************************************************************************
   // Correct for the variation between the Linac reference signal and the local time definition,
   // and a time shift due to a change of the z-coord based on the hodoscope info
   // **********************************************************************************************

   for (Int_t i=1; i<65;i++)
  {
   tdc_slot2_phillips[i] = tdc_slot2_phillips[i] + (42.0-time_walk_mean);
   //tdc_slot2_phillips[i] = tdc_slot2_phillips[i] + 0.005153*z_coord;
   tdc_slot3_phillips[i] = tdc_slot3_phillips[i] + (42.0-time_walk_mean);
   //tdc_slot3_phillips[i] = tdc_slot3_phillips[i] + 0.005153*z_coord;
   tdc_slot4_phillips[i] = tdc_slot4_phillips[i] + (42.0-time_walk_mean);
   //tdc_slot4_phillips[i] = tdc_slot4_phillips[i] + 0.003621*z_coord;
   tdc_slot5_phillips[i] = tdc_slot5_phillips[i] + (42.0-time_walk_mean);
   //tdc_slot5_phillips[i] = tdc_slot5_phillips[i] + 0.005885*z_coord;
   tdc_slot6_phillips[i] = tdc_slot6_phillips[i] + (42.0-time_walk_mean);
   //tdc_slot6_phillips[i] = tdc_slot6_phillips[i] + 0.005839*z_coord;
  }
```

TDC correction for Start counter 1 time walk:

```
// **********************************************************************************************
// Correct for the variation between the Linac reference signal and the local time definition,
// and a time shift due to a change of the z-coord based on the hodoscope info
// **********************************************************************************************
for (Int_t i=1; i<65;i++)
  {
   tdc_slot2_phillips[i] = tdc_slot2_phillips[i] + (42.0-time_walk_mean);
   //tdc_slot2_phillips[i] = tdc_slot2_phillips[i] + 0.005153*z_coord;
   tdc_slot3_phillips[i] = tdc_slot3_phillips[i] + (42.0-time_walk_mean);
   //tdc_slot3_phillips[i] = tdc_slot3_phillips[i] + 0.005153*z_coord;
   tdc_slot4_phillips[i] = tdc_slot4_phillips[i] + (42.0-time_walk_mean);
   //tdc_slot4_phillips[i] = tdc_slot4_phillips[i] + 0.003621*z_coord;
   tdc_slot5_phillips[i] = tdc_slot5_phillips[i] + (42.0-time_walk_mean);
   //tdc_slot5_phillips[i] = tdc_slot5_phillips[i] + 0.005885*z_coord;
   tdc_slot6_phillips[i] = tdc_slot6_phillips[i] + (42.0-time_walk_mean);
   //tdc_slot6_phillips[i] = tdc_slot6_phillips[i] + 0.005839*z_coord;
  }
```

Correct, using the Start counter 1, for the time walk:

```
if (good_hit && (n_good_hit > 100)) h_tdc_qtz_start1_ave_corr_rel_qtz_time->Fill(tdc_start1_ave+(42.0-time_walk_mean));
if (good_hit) h_tdc_scint_start2_ave_corr_rel_qtz_time->Fill(tdc_start2_ave+(42.0-time_walk_mean));
```

Code for the prototype, including cuts:

```
// **********************************************************************************************************
// **********************************************************************************************************
//
// The 1-st peak contribution:
//
// **********************************************************************************************************

       if (good_hit && (tdc_slot2_phillips[i] > low_cut_1st) && (tdc_slot2_phillips[i] < high_cut_1st))
      {
       weight = 1.0;

       //TOP_measured-TOP_expected; correct for possible errors in delay cable or TDC calibration:
```

```
diff_top = tdc_slot2_phillips[i]-top1_slot2[i]-time_offset_slot2_phillips_peak_1[i];

//(TOP_measured-TOP_expected)/Lpath:
diff_top_per_photon_path = diff_top/(path1_slot2[i]/100.);

// ****************************************************************************************************
// if (diff_top_per_photon_path < -1.0) cout << "[i_ev] = " << i_ev << ", [i] = " << i << ", tdc_slot2_phillips[i] = " << tdc_slot2_phillips[i] << ", top1_slot2[i] =" << top1_slot2[i] << ", time_offset_slot2_phillips_peak_1[i] =" << time_offset_slot2_phillips_peak_1[i] << ", diff_top = " << diff_top << ", path1_slot2[i]/100 = " << path1_slot2[i]/100 << ", diff_top_per_photon_path = " << diff_top_per_photon_path << endl;
// ****************************************************************************************************

//TOP_measured/Lpath (corrected for possible errors in TDC calibration):
TOP_per_photon_path_meas = (tdc_slot2_phillips[i]-time_offset_slot2_phillips_peak_1[i])/(path1_slot2[i]/100.);

//Calculate the group index from TOP_measured/Lpath:
N_group_meas = (0.299792458*TOP_per_photon_path_meas);
dN_group_meas = N_group_meas - 1.5;

// Position 1 is ~60cm from bar length:
// From spreadsheet for Burle MCP-PMT: weight = -3.438187292678E+06x5 + 2.637846733057E+07x4 - 8.093209893507E+07x3 + 1.241230486441E+08x2 - 9.515763898271E+07x + 2.917317761386E+07, x is a group index
// From spreadsheet for Hamamatsu MaPMT: weight = 1.947751440490E+06x5 - 1.464070908551E+07x4 + 4.400852088239E+07x3 - 6.612554145119E+07x2 + 4.966624644747E+07x - 1.491780661186E+07, x is a group index
//weight = 1.947751440490*1000000*N_group_meas*N_group_meas*N_group_meas*N_group_meas*N_group_meas - 1.464070908551*10000000*N_group_meas*N_group_meas*N_group_meas*N_group_meas + 4.400852088239*10000000*N_group_meas*N_group_meas*N_group_meas - 6.612554145119*10000000*N_group_meas*N_group_meas + 4.966624644747*10000000*N_group_meas - 1.491780661186*10000000;
//if(N_group_meas < 1.476 || N_group_meas > 1.56) weight = 0.0;

h_all_slots_all_pads_diff_1st_peak->Fill(diff_top, weight);
h_dTOP_per_photon_path_1st_peak->Fill(diff_top_per_photon_path, weight);

// n_phase from my spreadsheet:
n_phase_corr_slot2[i] = -0.3972*N_group_meas*N_group_meas + 1.5338*N_group_meas + 0.0588;

//Calculate phase index from the group index using the correlation from the spreadsheet calculation:
beta_ave_n_phase_slot2_corr[i] = 1./(cos(cherenkov_angle_slot2_corr[i]/57.3)*n_phase_corr_slot2[i]);

//Calculate Cherenkov angle using the phase index, which was derived from the group index first:
cher_ang_from_n_phase = 57.3*acos(1/n_phase_corr_slot2[i]);

//1. Correct Cherenkov angle using an empirical correlation with a variable d(TOP/Lpath) - using the 2nd peak photons:
//a - for variable lambda analysis:
//cher_ang_from_n_phase_corr = cher_ang_from_n_phase - (0.09517*diff_top_per_photon_path + 0.1303);
//b - for fixed lamda analysis:
cher_ang_from_n_phase_corr = cher_ang_from_n_phase - (3.808*diff_top_per_photon_path + 0.06068);
//2. Correct Cerenkov angle(TOP) using an empirical correlation: Cher.angle(pixels) vs. Cher.angle (TOP/Lpath):
//a - for variable lambda analysis:
//cher_ang_from_n_phase_corr_1 = cher_ang_from_n_phase - ((35.4 + 0.2497*cherenkov_angle_slot2_corr[i]) - 47.0);
//b - for fixed lambda analysis:
//cher_ang_from_n_phase_corr_1 = cher_ang_from_n_phase - ((36.3 + 0.2342*cherenkov_angle_slot2_corr[i]) - 47.0);
//c - for fixed lambda analysis, and set all timing epsilon offsets to zero:
//cher_ang_from_n_phase_corr_1 = cher_ang_from_n_phase - ((36.14 + 0.2289*cherenkov_angle_slot2_corr[i]) - 47.0);
//d - for fixed lambda analysis, and charge sharing - kill the second peak:
cher_ang_from_n_phase_corr_1 = cher_ang_from_n_phase - ((36.94 + 0.2207*cherenkov_angle_slot2_corr[i]) - 47.0);
//e - for fixed lambda analysis, and charge sharing - average the Cherenkov angle of two hits involved:
//cher_ang_from_n_phase_corr_1 = cher_ang_from_n_phase - ((44.7 + 0.0548*cherenkov_angle_slot2_corr[i]) - 47.0);
//3. Correct Cerenkov angle(pixels) using an empirical correlation: Cher.angle(pixels) vs. Cher.angle (TOP/Lpath):
//cher_ang_from_n_phase_corr_2 = cherenkov_angle_slot2_corr[i] - ((cher_ang_from_n_phase - 35.4)/0.2497 - 47.0);
cher_ang_from_n_phase_corr_2 = cherenkov_angle_slot2_corr[i] - ((cher_ang_from_n_phase - 36.3)/0.2342 - 47.0);
//4. Cherenkov angle correction from my spreadsheet: Cherenkov angle = -1.5975x2 + 3.6987x + 47.032
//cher_ang_time_correction = -1.5975*diff_top_per_photon_path*diff_top_per_photon_path+3.6987*diff_top_per_photon_path;
cher_ang_time_correction = -5.0241*diff_top_per_photon_path*diff_top_per_photon_path+5.401*diff_top_per_photon_path;

if ((fabs(diff_top_per_photon_path) < 2.0)  &&  (fabs(diff_top) < 5.0))
 {
  h_TOP_per_photon_path_1st_peak->Fill(TOP_per_photon_path_meas, weight);
  h_N_group_meas_1st_peak->Fill(N_group_meas, weight);
  h_dN_group_vs_dTOP_per_Lpath_1st_peak->Fill(diff_top_per_photon_path, dN_group_meas,weight);
```

```
            h_cherenkov_angle_chrom_corr_off_1st_peak->Fill(cherenkov_angle_slot2_corr[i],weight);
            h_cherenkov_angle_chrom_corr_off->Fill(cherenkov_angle_slot2_corr[i],weight);
            //h_cherenkov_angle_chrom_corr_on_1st_peak->Fill(cherenkov_angle_slot2_corr[i]-cher_ang_time_correction, weight);
            h_cherenkov_angle_chrom_corr_on_1st_peak->Fill(cherenkov_angle_slot2_corr[i]-2.628*diff_top_per_photon_path, weight);
            //Do not correct the 1-st peak in position 1:
            //h_cherenkov_angle_chrom_corr_on->Fill(cherenkov_angle_slot2_corr[i]-cher_ang_time_correction, weight);
            h_cherenkov_angle_chrom_corr_on->Fill(cherenkov_angle_slot2_corr[i], weight);
            //h_cherenkov_angle_chrom_corr_on->Fill(47.0-cher_ang_time_correction, weight);

            // Plot corrected Cherenkov distribution using various methods - the second peak only:
            //h_all_slot_all_pads_cher_ang_1st_peak_corr->Fill(cher_ang_from_n_phase_corr,weight);
            h_all_slot_all_pads_cher_ang_1st_peak_corr_1->Fill(cher_ang_from_n_phase_corr_1,weight);
            //h_all_slot_all_pads_cher_ang_1st_peak_corr_2->Fill(cher_ang_from_n_phase_corr_2,weight);

            h_all_slot_all_pads_beta_vs_beta_corr_1st_peak->Fill(beta_ave_n_phase_slot2[i], beta_ave_n_phase_slot2_corr[i],weight);
            h_all_slot_all_pads_beta_vs_beta_corr_1st_peak_proj_to_x->Fill(beta_ave_n_phase_slot2[i],weight);
            h_all_slot_all_pads_beta_vs_beta_corr_1st_peak_proj_to_y->Fill(beta_ave_n_phase_slot2_corr[i],weight);

            h_all_slot_all_pads_cher_vs_cher_corr_1st_peak->Fill(cherenkov_angle_slot2_corr[i],cher_ang_from_n_phase,weight);
            h_all_slot_all_pads_cher_vs_cher_corr_1st_peak_proj_to_x->Fill(cherenkov_angle_slot2_corr[i],weight);
            h_all_slot_all_pads_cher_vs_cher_corr_1st_peak_proj_to_y->Fill(cher_ang_from_n_phase,weight);
            h_all_slot_all_pads_cher_vs_dTOP_per_photon_path_1st_peak->Fill(diff_top_per_photon_path,cher_ang_from_n_phase, weight);
            h_all_slot_all_pads_cher_vs_dTOP_per_photon_path_1st_peak_corr-
>Fill(diff_top_per_photon_path,cher_ang_from_n_phase_corr,weight);
            h_all_slot_all_pads_cher_ang_1st_peak_corr->Fill(cher_ang_from_n_phase_corr,weight);

            ave_beta_n_phase_1st_peak_corr[i] = ave_beta_n_phase_1st_peak_corr[i] + beta_ave_n_phase_slot2_corr[i];
            ave_beta_ave_n_phase_1st_peak[i] = ave_beta_ave_n_phase_1st_peak[i] + beta_ave_n_phase_slot2[i];
            ave_n_phase_corr_1st_peak[i] = ave_n_phase_corr_1st_peak[i] + n_phase_corr_slot2[i];
            ave_cherenkov_angle_1st_peak[i] = ave_cherenkov_angle_1st_peak[i] + cherenkov_angle_slot2_corr[i];
            n_accept_1st_peak[i] = n_accept_1st_peak[i] + 1;
          }
      }
      }

// ***********************************************************************************************************
// ***********************************************************************************************************
//
// The 2-nd peak contribution:
//
// ***********************************************************************************************************
// ***********************************************************************************************************
     if (good_hit && (tdc_slot2_phillips[i] > low_cut_2nd) && (tdc_slot2_phillips[i] < high_cut_2nd))
      {
        weight = 1.0;

        //TOP_measured-TOP_expected:
        diff_top = tdc_slot2_phillips[i]-top2_slot2[i]-time_offset_slot2_phillips_peak_2[i];
        h_all_slots_all_pads_diff_2nd_peak->Fill(diff_top);

        //(TOP_measured-TOP_expected)/Lpath:
        diff_top_per_photon_path = diff_top/(path2_slot2[i]/100.);
        h_dTOP_per_photon_path_2nd_peak->Fill(diff_top_per_photon_path);
        h_dTOP_per_photon_path_corr_2nd_peak->Fill(diff_top_per_photon_path);

        //TOP_measured/Lpath (corrected for possible errors in TDC calibration):
        TOP_per_photon_path_meas = (tdc_slot2_phillips[i]-time_offset_slot2_phillips_peak_2[i])/(path2_slot2[i]/100.);

        //Calculate the group index from TOP_measured/Lpath:
        N_group_meas = (0.299792458*TOP_per_photon_path_meas);
        dN_group_meas = N_group_meas - 1.5;

        //Calculate phase index from the group index using the correlation from the spreadsheet calculation:
        n_phase_corr_slot2[i] = -0.3972*N_group_meas*N_group_meas + 1.5338*N_group_meas + 0.0588;

        //Calculate beta of the particle:
        beta_ave_n_phase_slot2_corr[i] = 1./(cos(cherenkov_angle_slot2_corr[i]/57.3)*n_phase_corr_slot2[i]);

        //Calculate Cherenkov angle using the phase index, which was derived from the group index first:
        cher_ang_from_n_phase = 57.3*acos(1/n_phase_corr_slot2[i]);
```

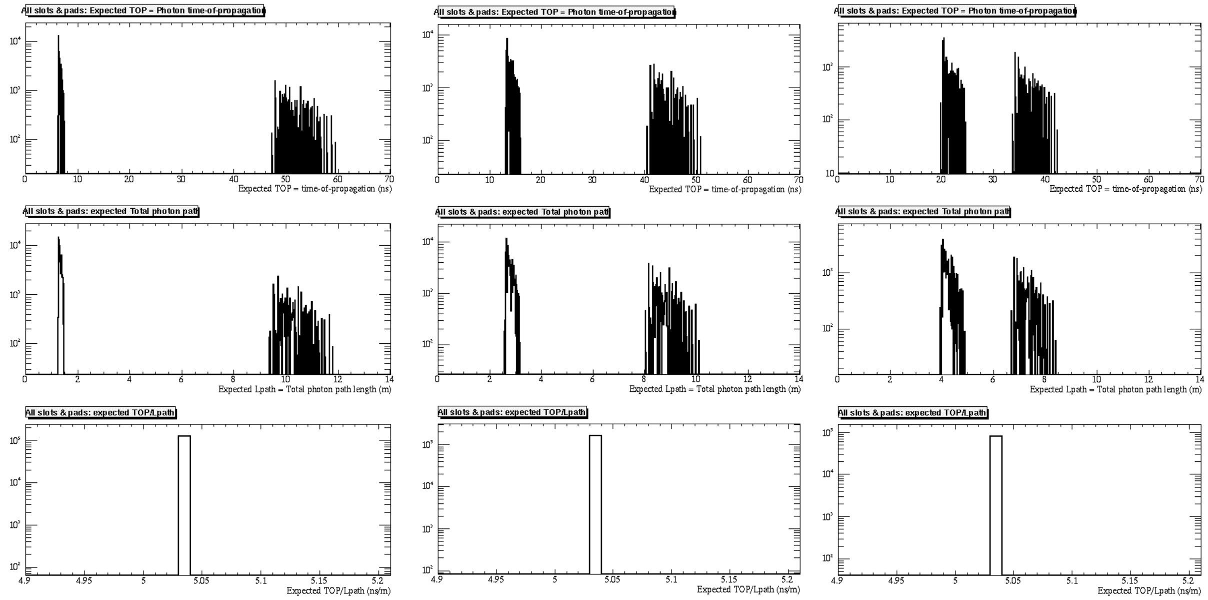
```
diff_cherenkov_angle = cherenkov_angle_slot2_corr[i]-47.032;

//1. Correct Cherenkov angle using an empirical correlation with a variable d(TOP/Lpath) - using the 2nd peak photons:
//a - for variable lambda analysis:
//cher_ang_from_n_phase_corr = cher_ang_from_n_phase - (0.09517*diff_top_per_photon_path + 0.1303);
//b - for fixed lamda analysis:
cher_ang_from_n_phase_corr = cher_ang_from_n_phase - (3.808*diff_top_per_photon_path + 0.06068);
//2. Correct Cerenkov angle(TOP) using an empirical correlation: Cher.angle(pixels) vs. Cher.angle (TOP/Lpath):
//a - for variable lambda analysis:
//cher_ang_from_n_phase_corr_1 = cher_ang_from_n_phase - ((35.4 + 0.2497*cherenkov_angle_slot2_corr[i]) - 47.0);
//b - for fixed lambda analysis:
//cher_ang_from_n_phase_corr_1 = cher_ang_from_n_phase - ((30.79 + 0.3471*cherenkov_angle_slot2_corr[i]) - 47.0);
//c - for fixed lambda analysis, and set all timing epsilon offsets to zero:
//cher_ang_from_n_phase_corr_1 = cher_ang_from_n_phase - ((34.29 + 0.2707*cherenkov_angle_slot2_corr[i]) - 47.0);
//d - for fixed lambda analysis, and charge sharing - kill the second peak:
cher_ang_from_n_phase_corr_1 = cher_ang_from_n_phase - ((30.62 + 0.3507*cherenkov_angle_slot2_corr[i]) - 47.0);
//e - for fixed lambda analysis, and charge sharing - average the Cherenkov angle of two hits involved:
//cher_ang_from_n_phase_corr_1 = cher_ang_from_n_phase - ((34.37 + 0.2708*cherenkov_angle_slot2_corr[i]) - 47.0);
//3. Correct Cerenkov angle(pixels) using an empirical correlation: Cher.angle(pixels) vs. Cher.angle (TOP/Lpath):
//cher_ang_from_n_phase_corr_2 = cherenkov_angle_slot2_corr[i] - ((cher_ang_from_n_phase - 35.4)/0.2497 - 47.0);
cher_ang_from_n_phase_corr_2 = cherenkov_angle_slot2_corr[i] - ((cher_ang_from_n_phase - 30.79)/0.3471 - 47.0);
//4. Cherenkov angle correction from my spreadsheet: Cherenkov angle = -1.5975x2 + 3.6987x + 47.032
//cher_ang_time_correction = -1.5975*diff_top_per_photon_path*diff_top_per_photon_path+3.6987*diff_top_per_photon_path;
cher_ang_time_correction = -5.0241*diff_top_per_photon_path*diff_top_per_photon_path+5.401*diff_top_per_photon_path;

if ((fabs(diff_top_per_photon_path) < 2.0) &&  (fabs(diff_top) < 5.0))
 {
  if (fabs(diff_cherenkov_angle) < 10.0)
   {
            //Find a correlation between Cherenkov angle, d(Cgerenkov angle) and d(TOP/Lpath)
            h_Cherenkov_angle_vs_TOP_per_photon_path_2nd_peak->Fill(diff_top_per_photon_path,
cherenkov_angle_slot2_corr[i],weight);
            h_dCherenkov_angle_vs_dTOP_per_photon_path_2nd_peak->Fill(diff_top_per_photon_path,
diff_cherenkov_angle,weight);
   }
  h_z_coord_vs_tdc_slot2_phillips->Fill(z_coord,tdc_slot2_phillips[i]);

  h_TOP_per_photon_path_2nd_peak->Fill(TOP_per_photon_path_meas);

  h_N_group_meas_2nd_peak->Fill(N_group_meas);
  h_dN_group_vs_dTOP_per_Lpath_2nd_peak->Fill(diff_top_per_photon_path, dN_group_meas,weight);

  // Cherenkov angle distribution for chromatic correction off:
  h_cherenkov_angle_chrom_corr_off_2nd_peak->Fill(cherenkov_angle_slot2_corr[i],weight);
  h_cherenkov_angle_chrom_corr_off->Fill(cherenkov_angle_slot2_corr[i],weight);

  h_all_slot_all_pads_beta_vs_beta_corr_2nd_peak->Fill(beta_ave_n_phase_slot2[i], beta_ave_n_phase_slot2_corr[i],weight);
  h_all_slot_all_pads_beta_vs_beta_corr_2nd_peak_proj_to_x->Fill(beta_ave_n_phase_slot2[i],weight);
  h_all_slot_all_pads_beta_vs_beta_corr_2nd_peak_proj_to_y->Fill(beta_ave_n_phase_slot2_corr[i],weight);

  h_all_slot_all_pads_cher_vs_cher_corr_2nd_peak->Fill(cherenkov_angle_slot2_corr[i],cher_ang_from_n_phase,weight);
  h_all_slot_all_pads_cher_vs_cher_corr_2nd_peak_proj_to_x->Fill(cherenkov_angle_slot2_corr[i],weight);
  h_all_slot_all_pads_cher_vs_cher_corr_2nd_peak_proj_to_y->Fill(cher_ang_from_n_phase, weight);

  h_all_slot_all_pads_cher_vs_dTOP_per_photon_path_2nd_peak->Fill(diff_top_per_photon_path,cher_ang_from_n_phase, weight);
  h_all_slot_all_pads_cher_vs_dTOP_per_photon_path_2nd_peak_corr-
>Fill(diff_top_per_photon_path,cher_ang_from_n_phase_corr,weight);

  // Plot corrected Cherenkov distribution using various methods - the second peak only:
  h_all_slot_all_pads_cher_ang_2nd_peak_corr->Fill(cher_ang_from_n_phase_corr,weight);
  h_all_slot_all_pads_cher_ang_2nd_peak_corr_1->Fill(cher_ang_from_n_phase_corr_1,weight);
  h_all_slot_all_pads_cher_ang_2nd_peak_corr_2->Fill(cher_ang_from_n_phase_corr_2,weight);

  // Chromatic correction from my spreadsheet:
  // Plot size of the chromatic correction:
  h_dCher_by_time_2nd_peak->Fill(cher_ang_time_correction);
  h_dCher_meas_minus_dCher_by_time_2nd_peak->Fill(diff_cherenkov_angle + cher_ang_time_correction);
  h_cherenkov_angle_chrom_corr_on->Fill(cherenkov_angle_slot2_corr[i] - cher_ang_time_correction);
  //Empirical correction from measured correlation:
```

```
            h_cherenkov_angle_chrom_corr_on_2nd_peak->Fill(cherenkov_angle_slot2_corr[i]-2.628*diff_top_per_photon_path);

            ave_beta_n_phase_2nd_peak_corr[i] = ave_beta_n_phase_2nd_peak_corr[i] + beta_ave_n_phase_slot2_corr[i];
            ave_beta_ave_n_phase_2nd_peak[i] = ave_beta_ave_n_phase_2nd_peak[i] + beta_ave_n_phase_slot2[i];
            ave_n_phase_corr_2nd_peak[i] = ave_n_phase_corr_2nd_peak[i] + n_phase_corr_slot2[i];
            ave_cherenkov_angle_2nd_peak[i] = ave_cherenkov_angle_2nd_peak[i] + cherenkov_angle_slot2_corr[i];
            if (weight > 0)
             {
            sigma_beta_n_phase_2nd_peak_corr[i] = sigma_beta_n_phase_2nd_peak_corr[i] + (beta_ave_n_phase_slot2_corr[i]-
0.985)*(beta_ave_n_phase_slot2_corr[i]-0.985);
            sigma_ave_beta_ave_n_phase_2nd_peak[i] = sigma_ave_beta_ave_n_phase_2nd_peak[i] + (beta_ave_n_phase_slot2[i]-
0.985)*(beta_ave_n_phase_slot2[i]-0.985);
            sigma_ave_n_phase_corr_2nd_peak[i] = sigma_ave_n_phase_corr_2nd_peak[i] + (n_phase_corr_slot2[i]-1.47)*(n_phase_corr_slot2[i]-
1.47);
            sigma_ave_cherenkov_angle_2nd_peak[i] = sigma_ave_cherenkov_angle_2nd_peak[i] + (cherenkov_angle_slot2_corr[i]-
47.14)*(cherenkov_angle_slot2_corr[i]-47.14);
             }
            n_accept_2nd_peak[i] = n_accept_2nd_peak[i] + 1;
            h_cherenkov_angle_chrom_corr_on_from_beta->Fill(57.3*acos(1./(beta_ave_n_phase_slot2_corr[i]*n_phase_corr_slot2[i])),weight);
           }
       }

      if (good_hit && (tdc_slot3_phillips[i] > low_cut_2nd) && (tdc_slot3_phillips[i] < high_cut_2nd))
       {
         weight = 1.0;

         //TOP_measured-TOP_expected:
         diff_top = tdc_slot3_phillips[i]-top2_slot3[i]-time_offset_slot3_phillips_peak_2[i];
         h_all_slots_all_pads_diff_2nd_peak->Fill(diff_top);

         //(TOP_measured-TOP_expected)/Lpath:
         diff_top_per_photon_path = diff_top/(path2_slot3[i]/100.);
         h_dTOP_per_photon_path_2nd_peak->Fill(diff_top_per_photon_path);
         h_dTOP_per_photon_path_corr_2nd_peak->Fill(diff_top_per_photon_path);

         //TOP_measured/Lpath (corrected for possible errors in TDC calibration):
         TOP_per_photon_path_meas = (tdc_slot3_phillips[i]-time_offset_slot3_phillips_peak_2[i])/(path2_slot3[i]/100.);

         //Calculate the group index from TOP_measured/Lpath:
         N_group_meas = (0.299792458*TOP_per_photon_path_meas);
         dN_group_meas = N_group_meas - 1.5;

         //Calculate phase index from the group index using the correlation from the spreadsheet calculation:
         n_phase_corr_slot3[i] = -0.3972*N_group_meas*N_group_meas + 1.5338*N_group_meas + 0.0588;

         //Calculate beta of the particle:
         beta_ave_n_phase_slot3_corr[i] = 1./(cos(cherenkov_angle_slot3_corr[i]/57.3)*n_phase_corr_slot3[i]);

         //Calculate Cherenkov angle using the phase index, which was derived from the group index first:
         cher_ang_from_n_phase = 57.3*acos(1/n_phase_corr_slot3[i]);

         diff_cherenkov_angle = cherenkov_angle_slot3_corr[i]-47.032;

         //1. Correct Cherenkov angle using an empirical correlation with a variable d(TOP/Lpath) - using the 2nd peak photons:
         //a - for variable lambda analysis:
         //cher_ang_from_n_phase_corr = cher_ang_from_n_phase - (0.09517*diff_top_per_photon_path + 0.1303);
         //b - for fixed lamda analysis:
         cher_ang_from_n_phase_corr = cher_ang_from_n_phase - (3.808*diff_top_per_photon_path + 0.06068);
         //2. Correct Cerenkov angle(TOP) using an empirical correlation: Cher.angle(pixels) vs. Cher.angle (TOP/Lpath):
         //a - for variable lambda analysis:
         //cher_ang_from_n_phase_corr_1 = cher_ang_from_n_phase - ((35.4 + 0.2497*cherenkov_angle_slot3_corr[i]) - 47.0);
         //b - for fixed lambda analysis:
         //cher_ang_from_n_phase_corr_1 = cher_ang_from_n_phase - ((30.79 + 0.3471*cherenkov_angle_slot3_corr[i]) - 47.0);
         //c - for fixed lambda analysis, and set all timing epsilon offsets to zero:
         //cher_ang_from_n_phase_corr_1 = cher_ang_from_n_phase - ((34.29 + 0.2707*cherenkov_angle_slot3_corr[i]) - 47.0);
         //d - for fixed lambda analysis, and charge sharing - kill the second peak:
         cher_ang_from_n_phase_corr_1 = cher_ang_from_n_phase - ((30.62 + 0.3507*cherenkov_angle_slot3_corr[i]) - 47.0);
         //e - for fixed lambda analysis, and charge sharing - average the Cherenkov angle of two hits involved:
         //cher_ang_from_n_phase_corr_1 = cher_ang_from_n_phase - ((34.37 + 0.2708*cherenkov_angle_slot3_corr[i]) - 47.0);
         //3. Correct Cerenkov angle(pixels) using an empirical correlation: Cher.angle(pixels) vs. Cher.angle (TOP/Lpath):
```

```
//cher_ang_from_n_phase_corr_2 = cherenkov_angle_slot3_corr[i] - ((cher_ang_from_n_phase - 35.4)/0.2497 - 47.0);
cher_ang_from_n_phase_corr_2 = cherenkov_angle_slot3_corr[i] - ((cher_ang_from_n_phase - 30.79)/0.3471 - 47.0);
//4. Cherenkov angle correction from my spreadsheet: Cherenkov angle = -1.5975x2 + 3.6987x + 47.032
//cher_ang_time_correction = -1.5975*diff_top_per_photon_path*diff_top_per_photon_path+3.6987*diff_top_per_photon_path;
cher_ang_time_correction = -5.0241*diff_top_per_photon_path*diff_top_per_photon_path+5.401*diff_top_per_photon_path;

if ((fabs(diff_top_per_photon_path) < 2.0) &&  (fabs(diff_top) < 5.0))
 {
  if (fabs(diff_cherenkov_angle) < 10.0)
   {
          h_Cherenkov_angle_vs_TOP_per_photon_path_2nd_peak->Fill(diff_top_per_photon_path,
cherenkov_angle_slot3_corr[i],weight);
          h_dCherenkov_angle_vs_dTOP_per_photon_path_2nd_peak->Fill(diff_top_per_photon_path,
diff_cherenkov_angle,weight);
   }
  h_TOP_per_photon_path_2nd_peak->Fill(TOP_per_photon_path_meas);

  h_N_group_meas_2nd_peak->Fill(N_group_meas);
  h_dN_group_vs_dTOP_per_Lpath_2nd_peak->Fill(diff_top_per_photon_path, dN_group_meas,weight);

  h_cherenkov_angle_chrom_corr_off_2nd_peak->Fill(cherenkov_angle_slot3_corr[i],weight);
  h_cherenkov_angle_chrom_corr_off->Fill(cherenkov_angle_slot3_corr[i],weight);

  h_all_slot_all_pads_beta_vs_beta_corr_2nd_peak->Fill(beta_ave_n_phase_slot3[i], beta_ave_n_phase_slot3_corr[i],weight);
  h_all_slot_all_pads_beta_vs_beta_corr_2nd_peak_proj_to_x->Fill(beta_ave_n_phase_slot3[i],weight);
  h_all_slot_all_pads_beta_vs_beta_corr_2nd_peak_proj_to_y->Fill(beta_ave_n_phase_slot3_corr[i],weight);

  h_all_slot_all_pads_cher_vs_cher_corr_2nd_peak->Fill(cherenkov_angle_slot3_corr[i],cher_ang_from_n_phase,weight);
  h_all_slot_all_pads_cher_vs_cher_corr_2nd_peak_proj_to_x->Fill(cherenkov_angle_slot3_corr[i],weight);
  h_all_slot_all_pads_cher_vs_cher_corr_2nd_peak_proj_to_y->Fill(cher_ang_from_n_phase, weight);
  h_all_slot_all_pads_cher_vs_dTOP_per_photon_path_2nd_peak->Fill(diff_top_per_photon_path,cher_ang_from_n_phase, weight);
  h_all_slot_all_pads_cher_vs_dTOP_per_photon_path_2nd_peak_corr-
>Fill(diff_top_per_photon_path,cher_ang_from_n_phase_corr,weight);

  // Plot corrected Cherenkov distribution using various methods - the second peak only:
  h_all_slot_all_pads_cher_ang_2nd_peak_corr->Fill(cher_ang_from_n_phase_corr,weight);
  h_all_slot_all_pads_cher_ang_2nd_peak_corr_1->Fill(cher_ang_from_n_phase_corr_1,weight);
  h_all_slot_all_pads_cher_ang_2nd_peak_corr_2->Fill(cher_ang_from_n_phase_corr_2,weight);
  // Chromatic correction from my spreadsheet:
  // Plot size of the chromatic correction:
  h_dCher_by_time_2nd_peak->Fill(cher_ang_time_correction);
  h_dCher_meas_minus_dCher_by_time_2nd_peak->Fill(diff_cherenkov_angle + cher_ang_time_correction);
  h_cherenkov_angle_chrom_corr_on->Fill(cherenkov_angle_slot3_corr[i] - cher_ang_time_correction);
  //Empirical correction from measured correlation:
  h_cherenkov_angle_chrom_corr_on_2nd_peak->Fill(cherenkov_angle_slot3_corr[i]-2.628*diff_top_per_photon_path);

  ave_beta_n_phase_2nd_peak_corr[i] = ave_beta_n_phase_2nd_peak_corr[i] + beta_ave_n_phase_slot3_corr[i];
  ave_beta_ave_n_phase_2nd_peak[i] = ave_beta_ave_n_phase_2nd_peak[i] + beta_ave_n_phase_slot3[i];
  ave_n_phase_corr_2nd_peak[i] = ave_n_phase_corr_2nd_peak[i] + n_phase_corr_slot3[i];
  ave_cherenkov_angle_2nd_peak[i] = ave_cherenkov_angle_2nd_peak[i] + cherenkov_angle_slot3_corr[i];
  if (weight > 0)
   {
  sigma_beta_n_phase_2nd_peak_corr[i] = sigma_beta_n_phase_2nd_peak_corr[i] + (beta_ave_n_phase_slot3_corr[i]-
0.985)*(beta_ave_n_phase_slot3_corr[i]-0.985);
  sigma_ave_beta_ave_n_phase_2nd_peak[i] = sigma_ave_beta_ave_n_phase_2nd_peak[i] + (beta_ave_n_phase_slot3[i]-
0.985)*(beta_ave_n_phase_slot3[i]-0.985);
  sigma_ave_n_phase_corr_2nd_peak[i] = sigma_ave_n_phase_corr_2nd_peak[i] + (n_phase_corr_slot3[i]-1.47)*(n_phase_corr_slot3[i]-
1.47);
  sigma_ave_cherenkov_angle_2nd_peak[i] = sigma_ave_cherenkov_angle_2nd_peak[i] + (cherenkov_angle_slot3_corr[i]-
47.14)*(cherenkov_angle_slot3_corr[i]-47.14);
   }
  n_accept_2nd_peak[i] = n_accept_2nd_peak[i] + 1;
  h_cherenkov_angle_chrom_corr_on_from_beta->Fill(57.3*acos(1./(beta_ave_n_phase_slot3_corr[i]*n_phase_corr_slot3[i])),weight);
 }
}
```

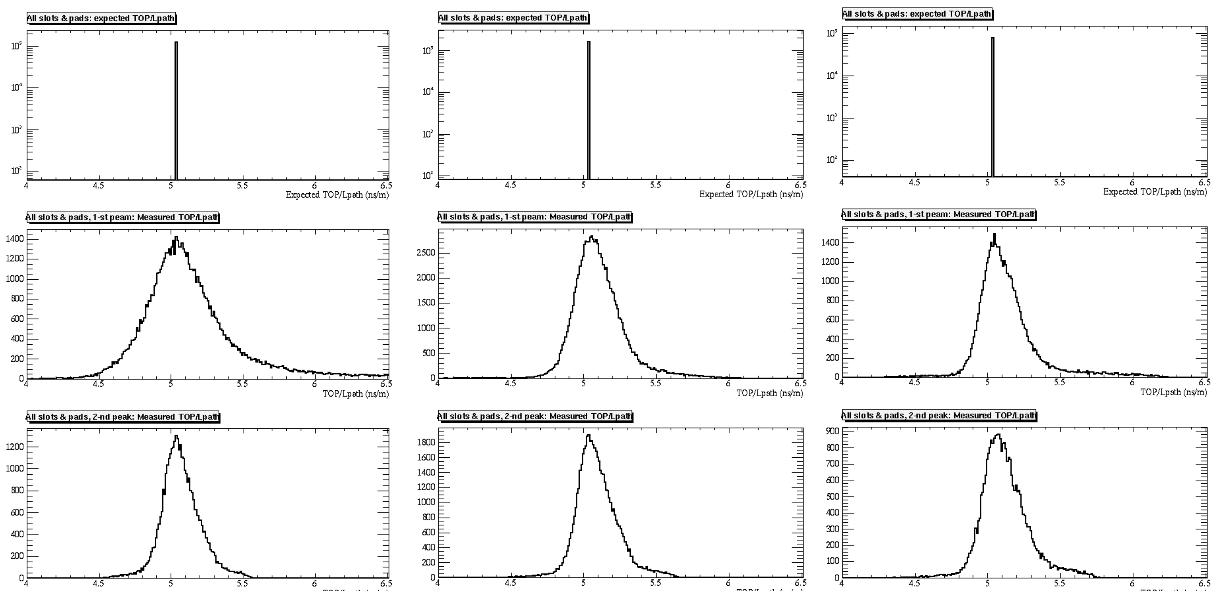
1) Expected total TOP [ns], Lpath [m], TOP/Lpath [ns/m]:

Position 1: **Position 3:** **Position 5:**



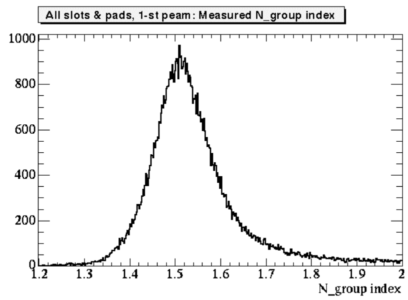
2) Expected and measured TOP/Lpath [ns/m]:

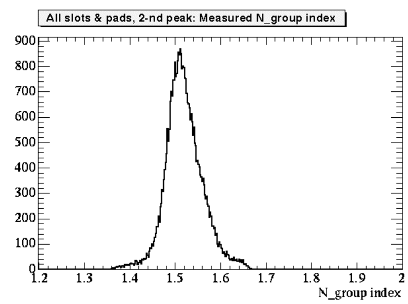
Position 1: **Position 3:** **Position 5:**



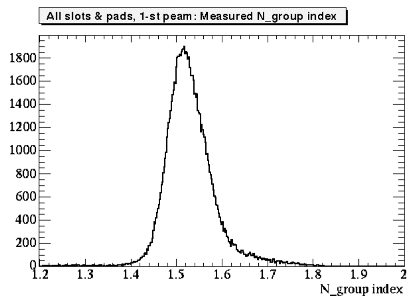
3) Measured group refraction index:

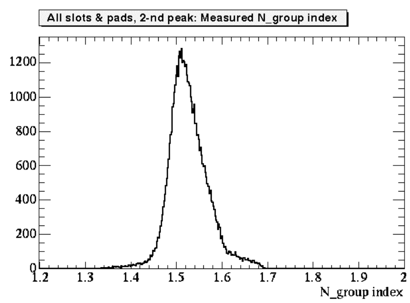
Position 1:



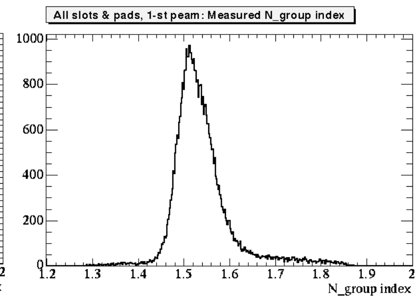


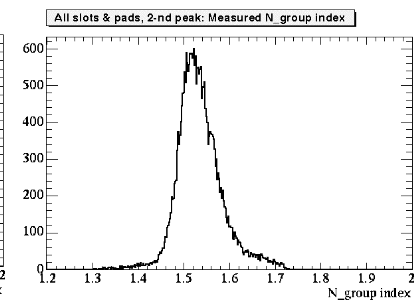
Position 3:





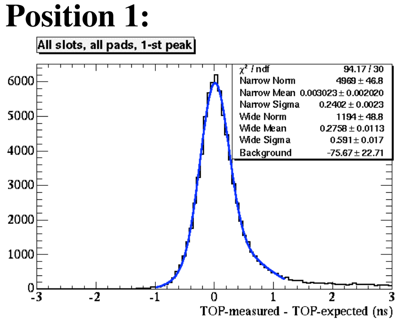
Position 5:

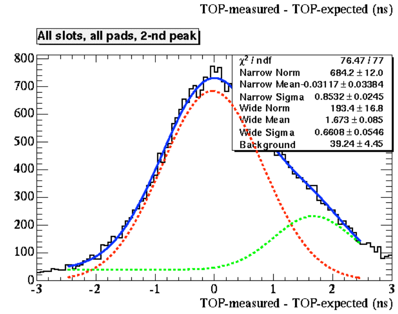




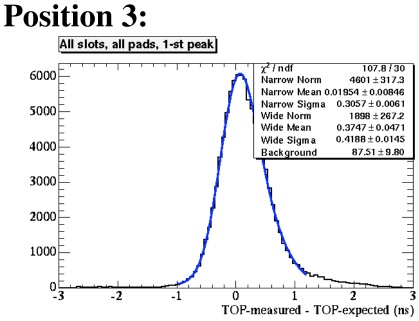
4) dTOP:

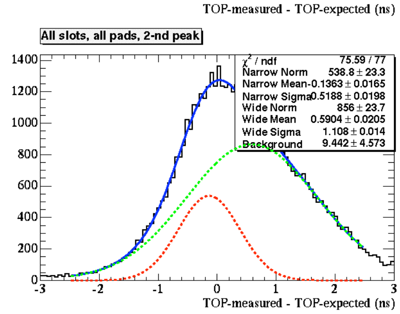
Position 1:



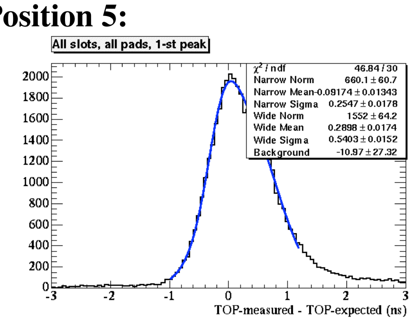


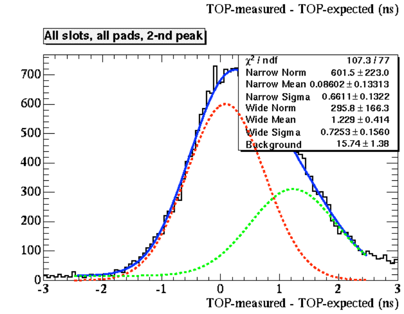
Position 3:





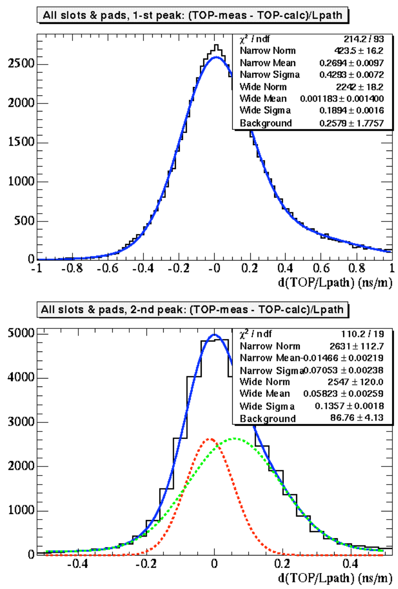
Position 5:



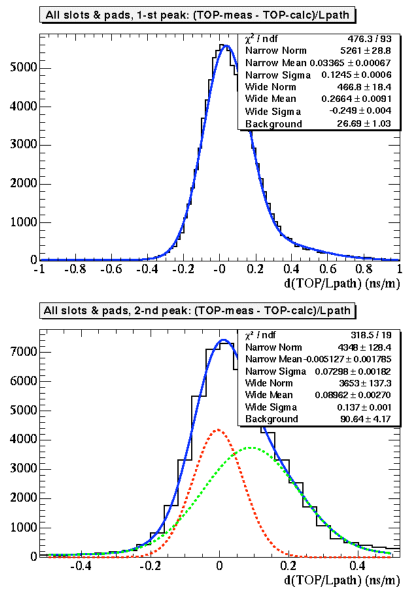


5) dTOP/Lpath:

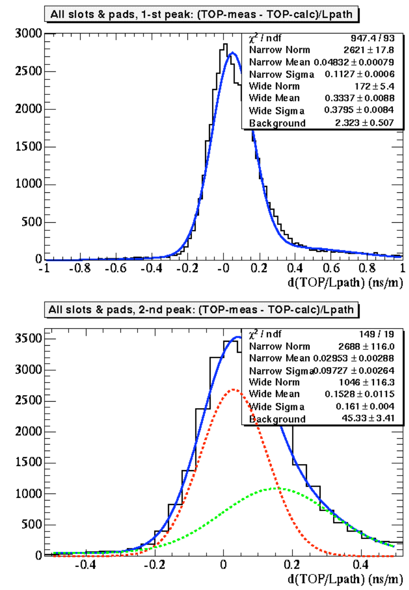
Position 1:



Position 3:

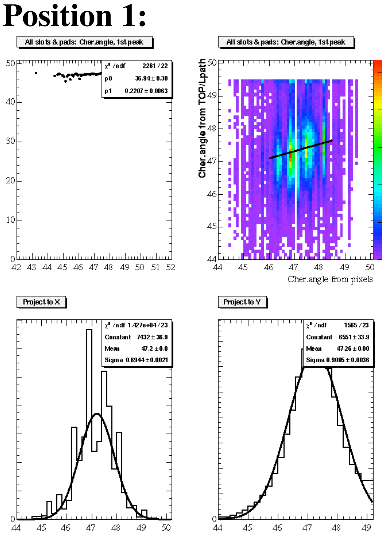


Position 5:

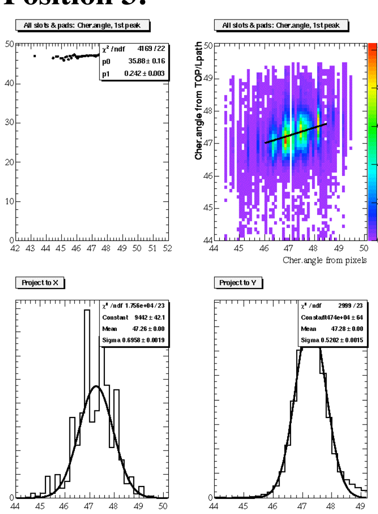


6) Correlation between Cherenkov angle from pixels and from TOP/Lpath – Peak 1:

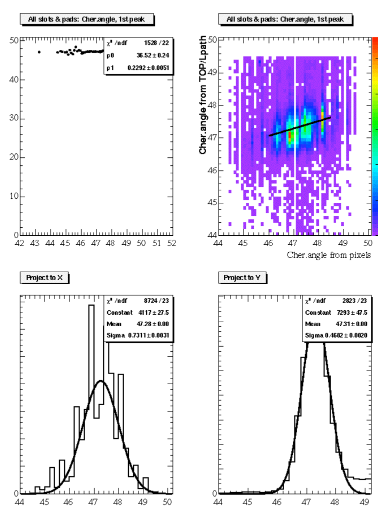
Position 1:



Position 3:

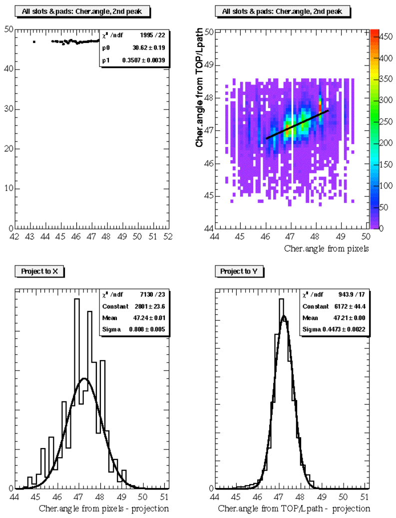


Position 5:

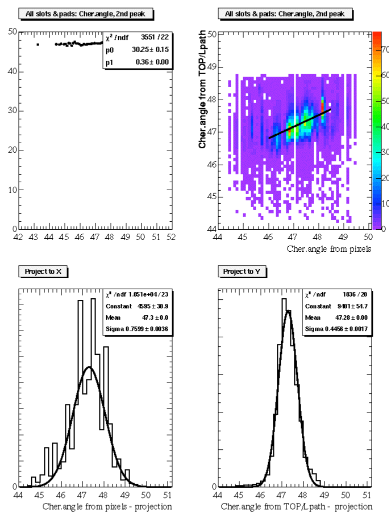


7) Correlation between Cherenkov angle from pixels and from TOP/Lpath – Peak 2:

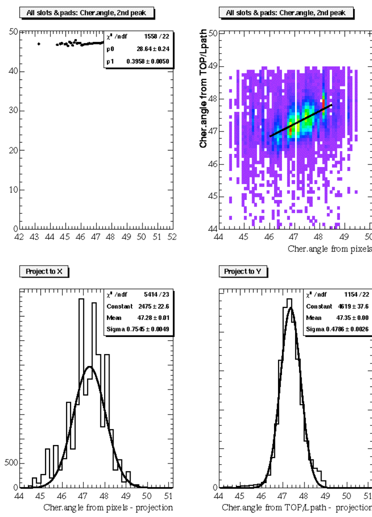
Position 1:



Position 3:

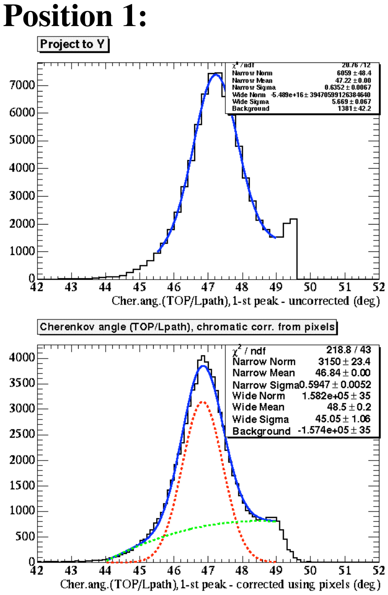


Position 5:

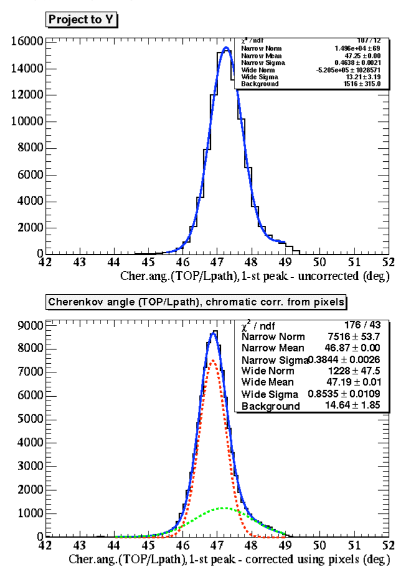


8) Cherenkov angle from TOP/Lpath with correction from pixels – Peak 1:

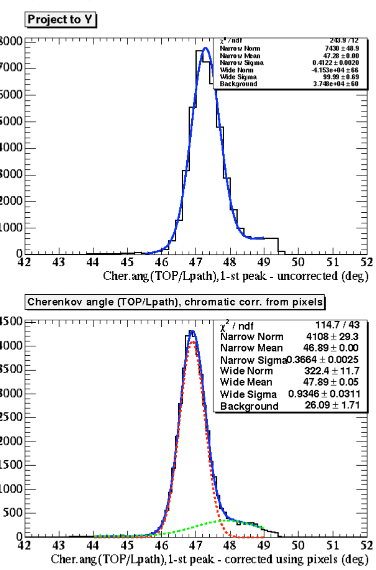
Position 1:



Position 3:

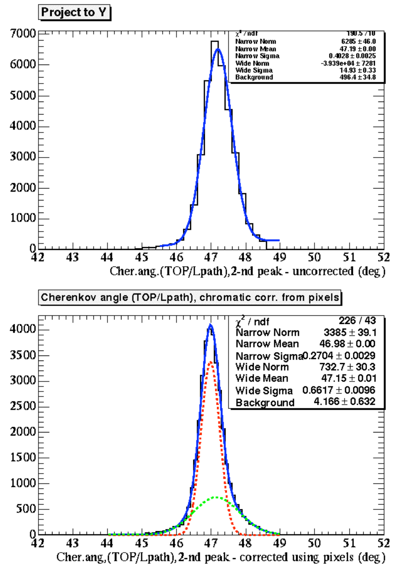


Position 5:

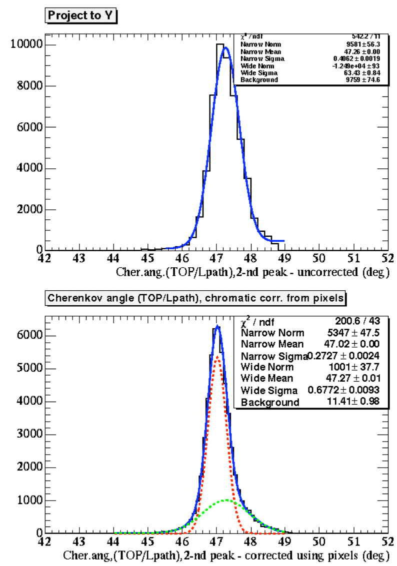


9) Cherenkov angle from TOP/Lpath with correction from pixels – Peak 2:

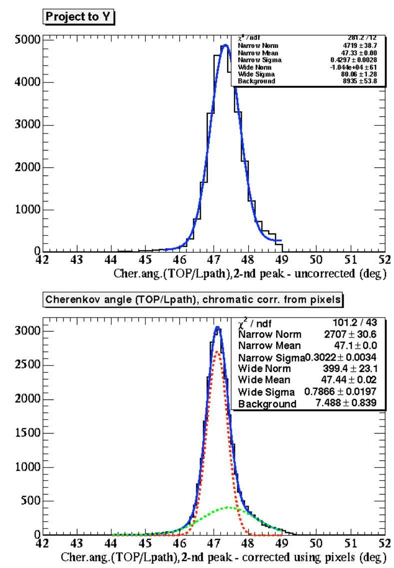
Position 1:



Position 3:

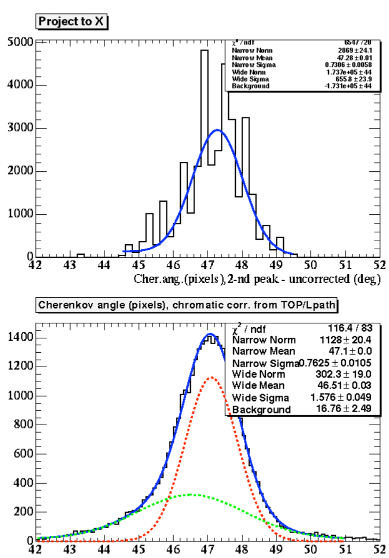


Position 5:

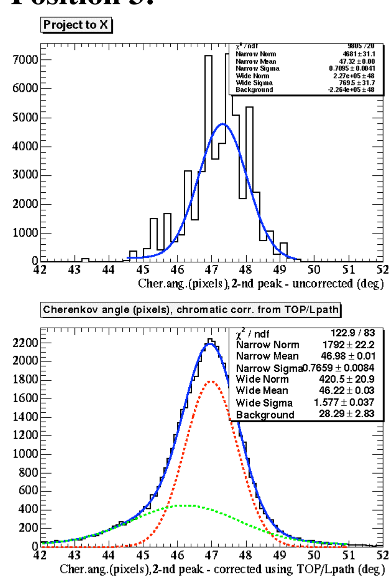


10) Cherenkov angle from pixels with correction from TOP/Lpath – Peak 2:

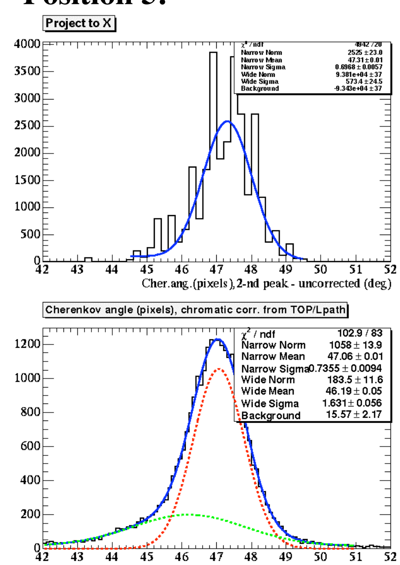
Position 1:



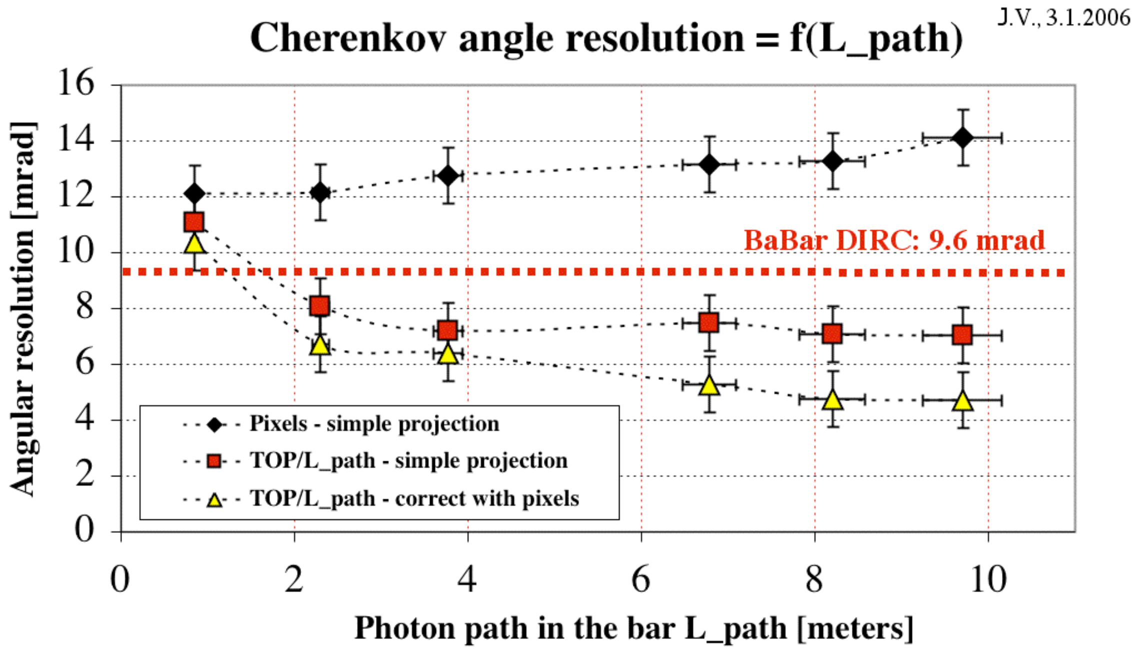
Position 3:



Position 5:



Summary of results with "loose cuts" on dTOP & d(TOP/Lpath):



Direct							
Position along the bar	Run	Ave. photon L_path [m]	Error [m]	Ch. ang. resol. pixels - simply project [mrad]	Ch. ang. resol. TOP/L_path - simply project [mrad]	Ch. ang. resol. pixels - correct with TOP/L_path [mrad]	Ch. ang. resol. TOP/L_path - correct with pixels [mraD]
1	12b	0.85	0.04	12.11867365	11.08551483		10.37870855
2							
3	13	2.3	0.1	12.14310646	8.094240838		6.708551483
4							
5	14	3.77	0.17	12.7591623	7.193717277		6.394415358
6							

Backward							
Position along the bar	Run	Ave. photon L_path in bar [m]	Error [m]	Ch. ang. resol. pixels - simply project [mrad]	Ch. ang. resol. TOP/L_path - simply project [mrad]	Ch. ang. resol. pixels - correct with TOP/L_path [mrad]	Ch. ang. resol. TOP/L_path - correct with pixels [mrad]
1	12b	9.7	0.46	14.10122164	7.029668412		4.719022688
2							
3	13	8.2	0.38	13.2617801	7.089005236		4.759162304
4							
5	14	6.78	0.31	13.16753927	7.4991274		5.27399651
6							

Both					
Ave. photon L_path [m]	Error [m]	Ch. ang. resol. pixels - simply project [mrad]	Ch. ang. resol. TOP/L_path - simply project [mrad]		Ch. ang. resol. TOP/L_path - correct with pixels [mraD]
0.85	0.04	12.11867365	11.08551483		10.37870855
2.3	0.1	12.14310646	8.094240838		6.708551483
3.77	0.17	12.7591623	7.193717277		6.394415358
6.78	0.31	13.16753927	7.4991274		5.27399651
8.2	0.38	13.2617801	7.089005236		4.759162304
9.7	0.46	14.10122164	7.029668412		4.719022688