Performance Study of JLC Tracker System As a Whole For Reconcilation of CDC Parameters

Keisuke Fujii
KEK

Plan of Talk

- Review of ACFA99 Presentation
- Momentum Resolution Study Since Then
  - How to Run a ROOT App. RESOL
  - Results
    - CDC Alone
    - CDC + VTX
    - CDC + VTX + IMT
- Summary

How to Run Resol (README)

http://www-jlc.kek.jp/subg/cdc/lib/SOFT/resol-0.60.tar.gz

Tracking System Design Tool

Version: 0.60

Description: A ROOT application to analytically estimate momentum resolutions for various tracking system configurations.

UpdateRecord:

1999/11/10 K.Fujii(0.60) Now creates resolution plots if a list of momenta is given.

1999/11/09 K.Fujii(0.50) Original C++ version derived from its FORTRAN ancestor.

Installation:

```
$ xmkmf -a
$ make
```

Usage:

```
$ cd prod
$ ./resol [p(GeV)|50] [cos(theta)|0.] 2> /dev/null
```

This calculates sigma_kappa for (p,cos(theta)).
If a list of more than 2 numbers is given, they are assumed to be a list of momenta in GeV and

```
$ ./resol 1. 5. 10. 20. 50. 100. 250. 2> /dev/null
```

will create resolution plots.

If you want to change detector configurations, modify CTDResol.cxx which is well commented.

Limitations:

Currently IMT and VTX are infinitely long. Consequently, the cos(theta) dependence in the forward region may well be very different from reality.
The CDC is assumed to be the outermost tracker, meaning that no outer trackers are supported at present.
Analytical formulae are based on parabollic approximation and are exact only in the high momentum limit.

How to Define a Tracker System

CTDResol.cxx defines the tracker system configuration

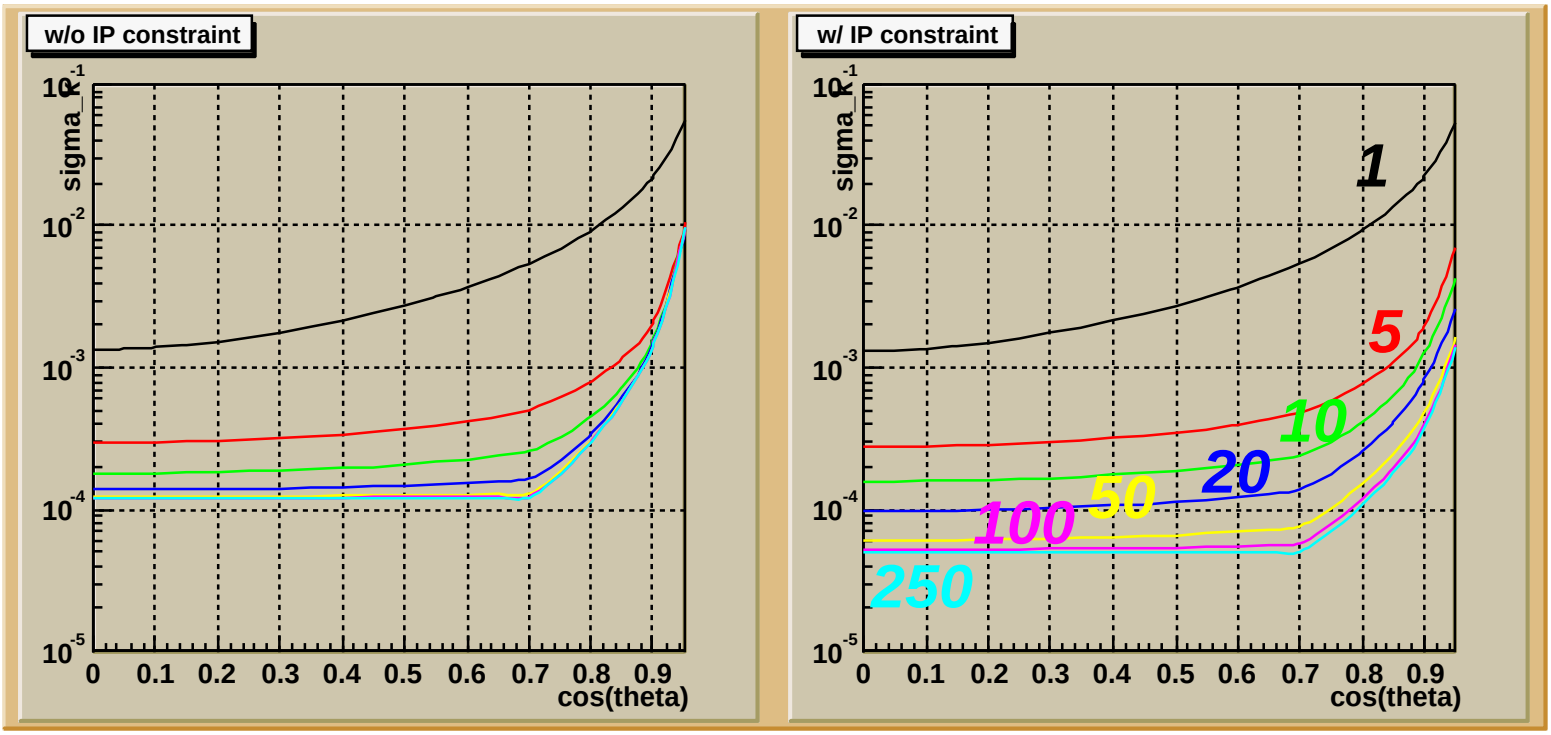
```
.....
//_____________________________________________________________________________
//  ------------------------------
//  void CTDResol(p,tanl,sgk,sgkip)
//  ------------------------------
//  (Inputs)
// Double_t p : momentum (GeV)
// Double_t tanl : tan(lambda)
//  (Outputs)
// Double_t &sgk : sigma_kappa w/o IP constraint (1/GeV)
// Double_t &sgkip : sigma_kappa w/  IP constraint (1/GeV)
//
void CTDResol(Double_t p, Double_t tanl, Double_t &sgk, Double_t &sgkip)
{
   .............
   // ------------------------------
   // Initialize detector properties
   // ------------------------------
   Double_t sgxdum  = 1.e1;    // dummy sigma_x (cm)
   Double_t bfield  = 3.;          // B field (T)
   Int_t       withvtx = 1;          // (0,1) = (w/o,w/) VTX
   Double_t sgxvtx  = 0.0004;  // sigma_x(VTX) (cm)
   Int_t       withimt = 1;          // (0,1) = (w/o,w/) IMT
   Double_t sgximt  = 0.0040;  // sigma_x(IMT) (cm)
   Int_t       nscdc   = 50;       // N_samples of CDC
   Double_t rincdc  = 45.;       // R_in(CDC)    (cm)
   Double_t routcdc = 155.;     // R_out(CDC)   (cm)
   Double_t sgxcdc  = 0.0085; // sigma_x(CDC) (cm)
   .............
```

These variables are later used to define a tracker system.
See the next page.

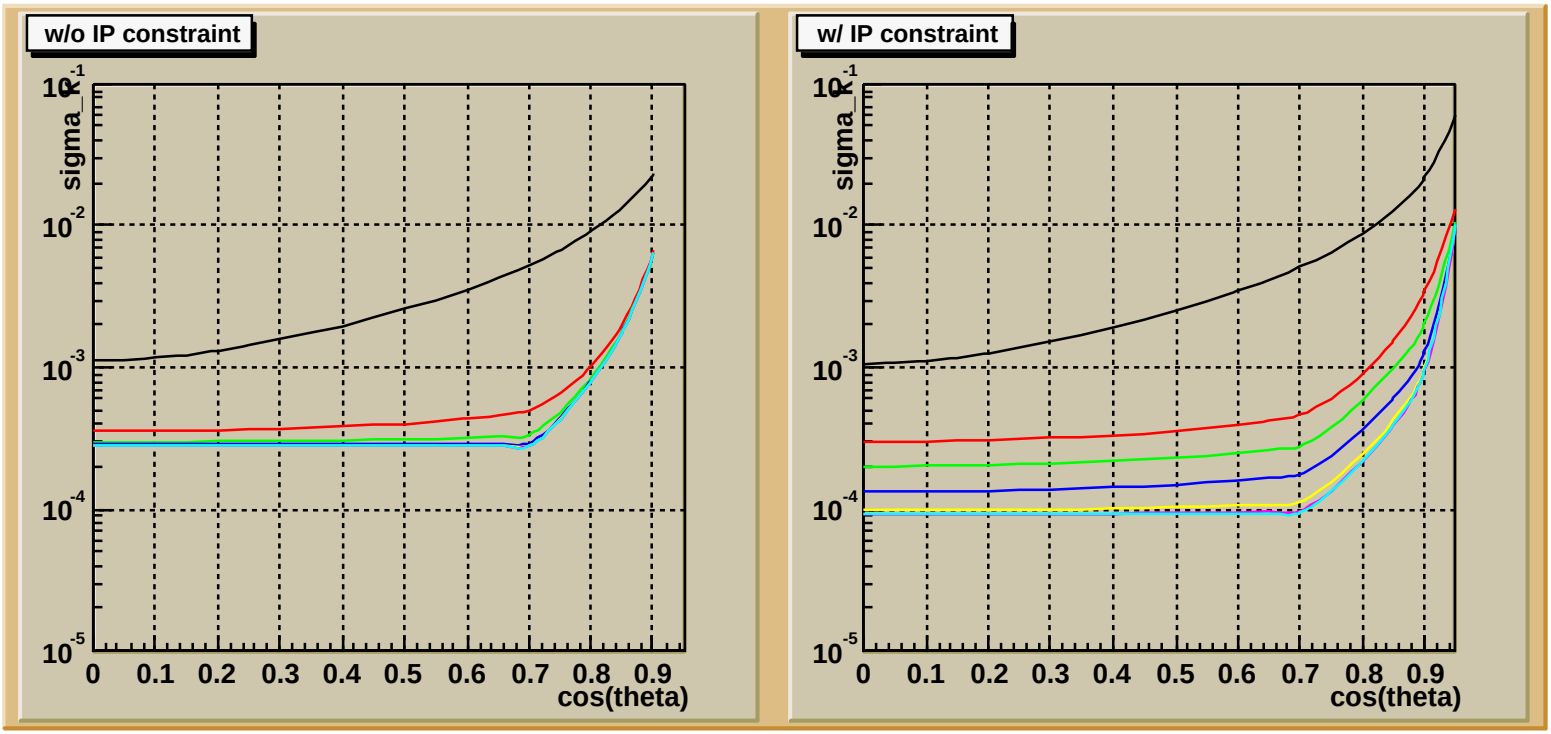
```
  .....
  // ----------------------------------
  // Initialize the tracker configuration
  // ----------------------------------
  // [1] How to define a layer
  //     Create a CTDLayer object with
  //      CTDLayer(name,Nsample,Rin,thickness,Bfield,X0,sgimax)
  //     wehere
  //  Char_t  *name  : name of the layer
  //  Int_t    Nsample : No. samples in this layer
  //  Double_t Rin  : inner radius (cm)
  //  Double_t thickness : thickness of the layer (cm)
  //  Double_t Bfield  : B field (T)
  //  Double_t X0  : radiation length (1/cm)
  //  Double_t sigmax  : resolution per sample (cm)
  //     Alternatively, one can use
  //     CTDLayer(layer,Rin)
  //     where
  //  const CTDLayer &layer : previous layer of the same kind
  //  Rin   : inner radius of the new layer (cm)
  //   if some layer that only differs in Rin has already been defined.
  //   The defined layer should be added to a CTDTracker object.
  //   Once layers other than air are added, one can fill the
  //   inter-layer dead gaps by air by calling
  //     FillGapByAir(x0air)
  //   member function of the CTDTracker class.
  //
  CTDTracker tracker;
  CTDLayer vac("Vacuum",0,0.0,2.00,bfield,x0vac,sgxdum); tracker.Add(&vac);
  CTDLayer bp("BM Pipe",0,2.0,0.05,bfield,x0be ,sgxdum); tracker.Add(&bp);
  CTDLayer vtx1("VTX",withvtx,2.4,0.03,bfield,x0si,sgxvtx); tracker.Add(&vtx1);
  CTDLayer vtx2(vtx1,3.6); tracker.Add(&vtx2);
  CTDLayer vtx3(vtx1,4.8); tracker.Add(&vtx3);
  CTDLayer vtx4(vtx1,6.0); tracker.Add(&vtx4);
  CTDLayer imt1("IMT",withimt,10.,0.03,bfield,x0si,sgximt); tracker.Add(&imt1);
  CTDLayer imt2(imt1,17.); tracker.Add(&imt2);
  CTDLayer imt3(imt1,24.); tracker.Add(&imt3);
  CTDLayer imt4(imt1,31.); tracker.Add(&imt4);
  CTDLayer imt5(imt1,38.); tracker.Add(&imt5);
  CTDLayer st("SUP",0,40.0,0.5,bfield,x0cfrp,sgxdum); tracker.Add(&st);
  CTDLayer cdc1("CDC",0,40.5,4.5,bfield,x0cdc,sgxdum); tracker.Add(&cdc1);
  CTDLayer cdc2("CDC",ns,45.0,elcdc,bfield,x0cdc,sgxcdc); tracker.Add(&cdc2);
  tracker.FillGapByAir(x0air);  // fills gaps by air
  // ----------------------------------
  // Print out the tracker configuration
  // ----------------------------------
  tracker.DebugPrint();
  // ----------------------------------
  // Calculate momentum resolution
  // ----------------------------------
  sgk   = tracker.GetSigmaKappa(p,tanl); // sigma_kappa w/o IP
  sgkip = tracker.GetSigmaKappaIP(p,tanl); // sigma_kappa w/  IP
}
```

2T Design vs 3T Design

B=2T

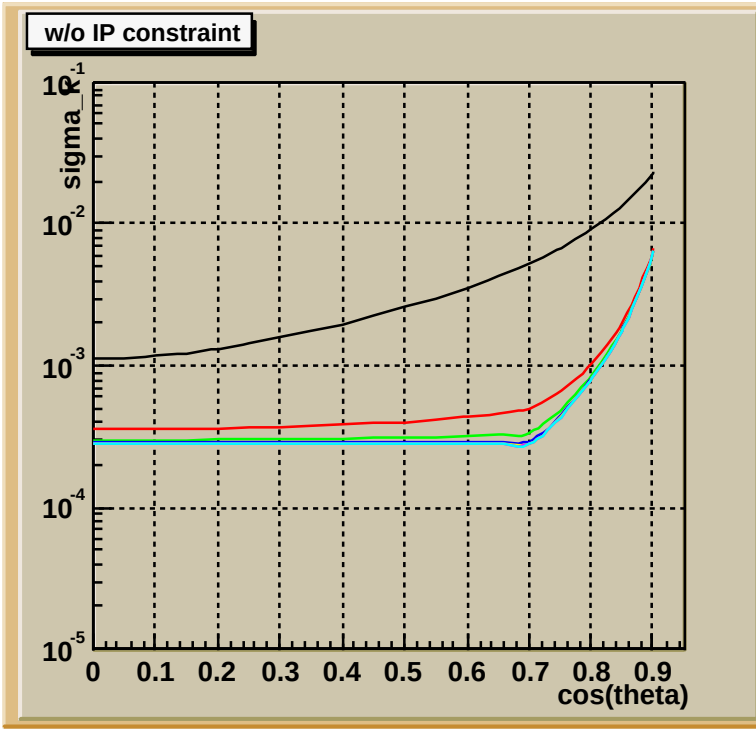


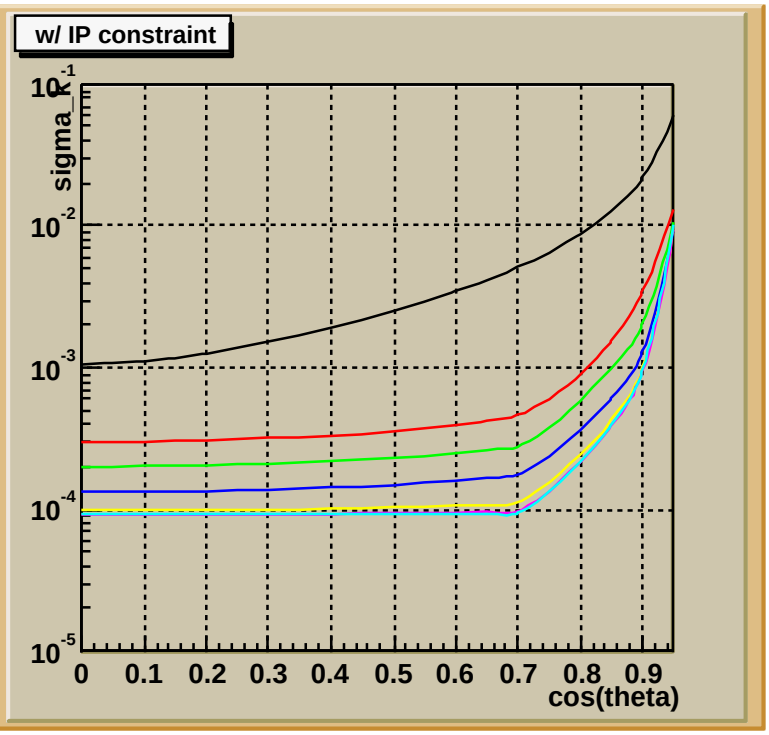
B=3T



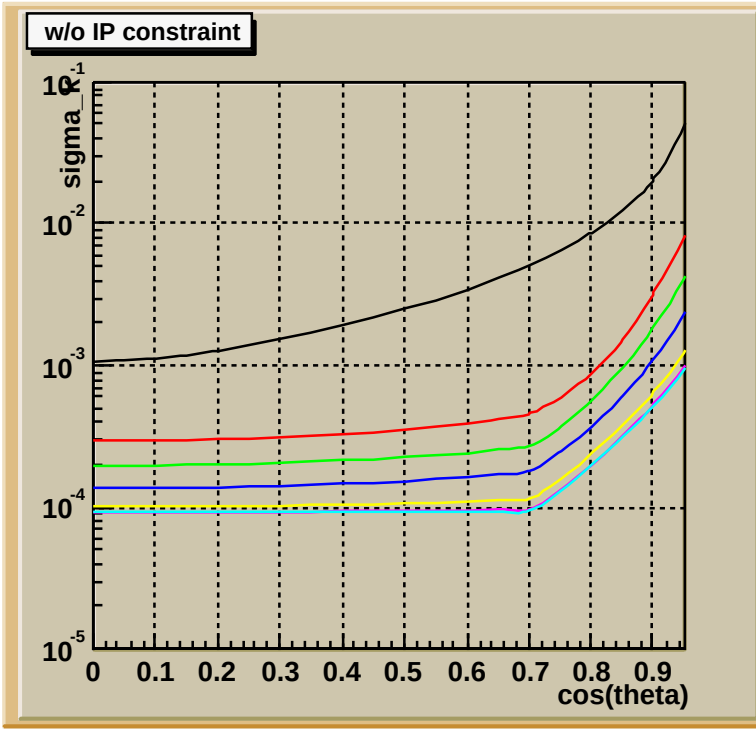
Effect of VTX $\sigma_{xy} = 4\mu m$

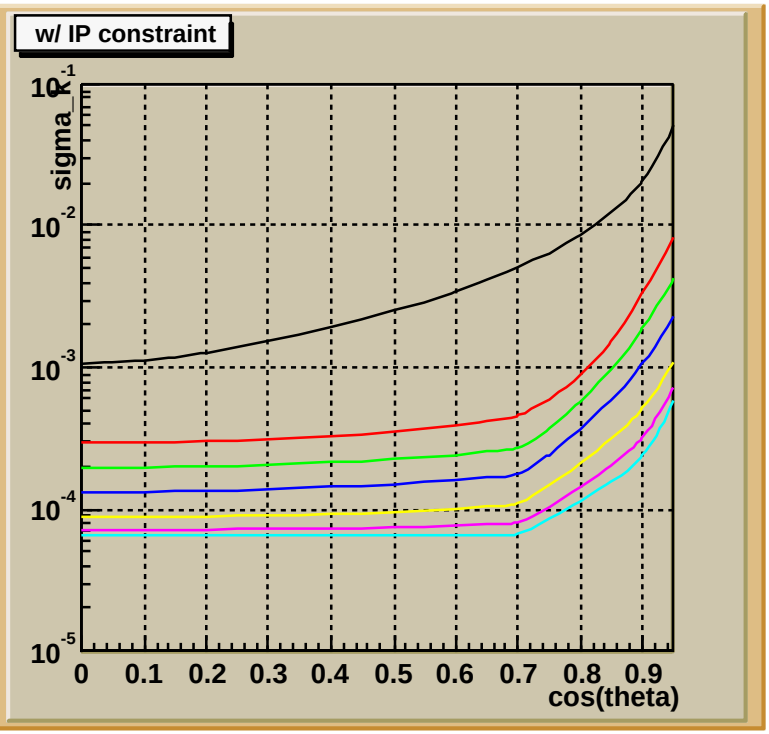
Without VTX





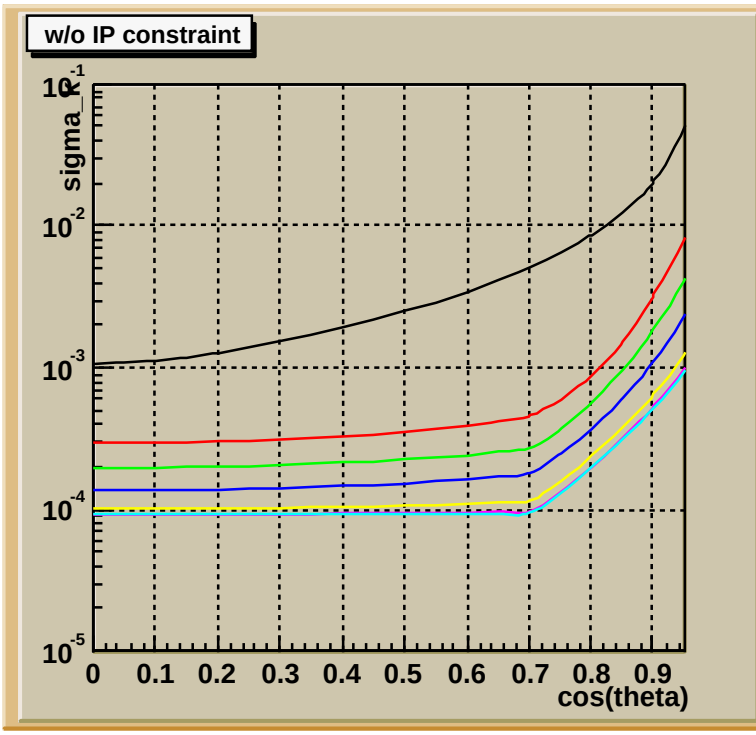
With VTX

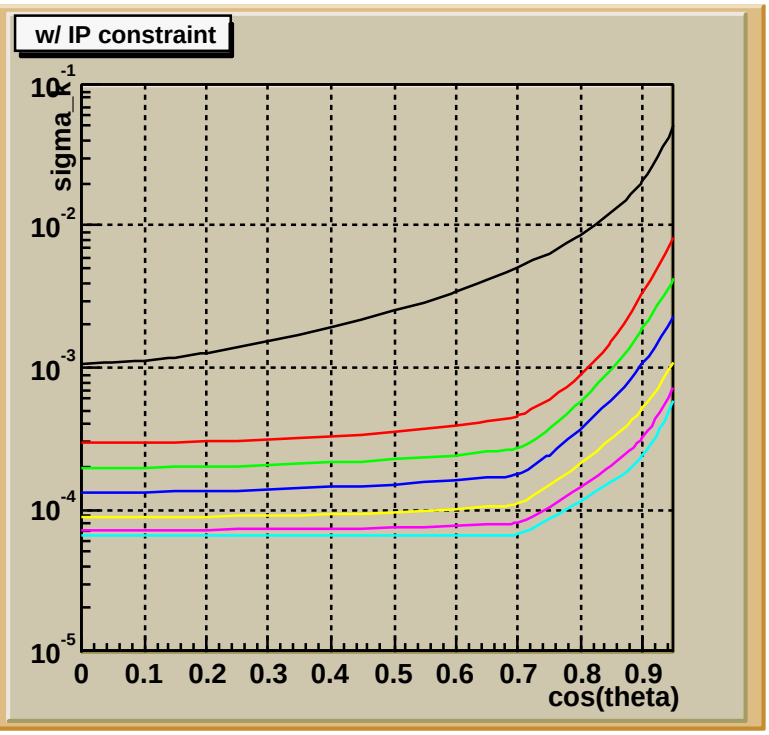




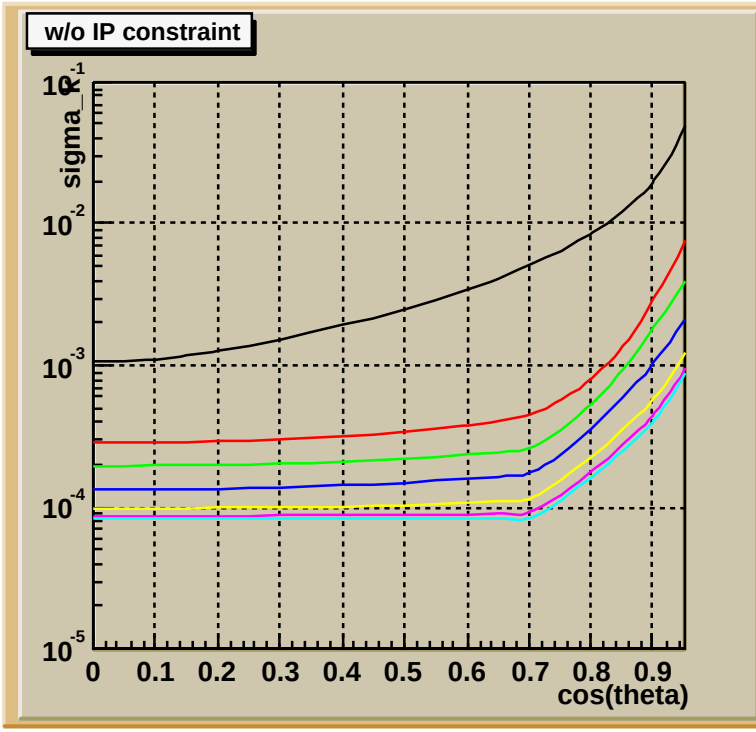
Effect of IMT $\sigma_{xy} = 40\mu m$

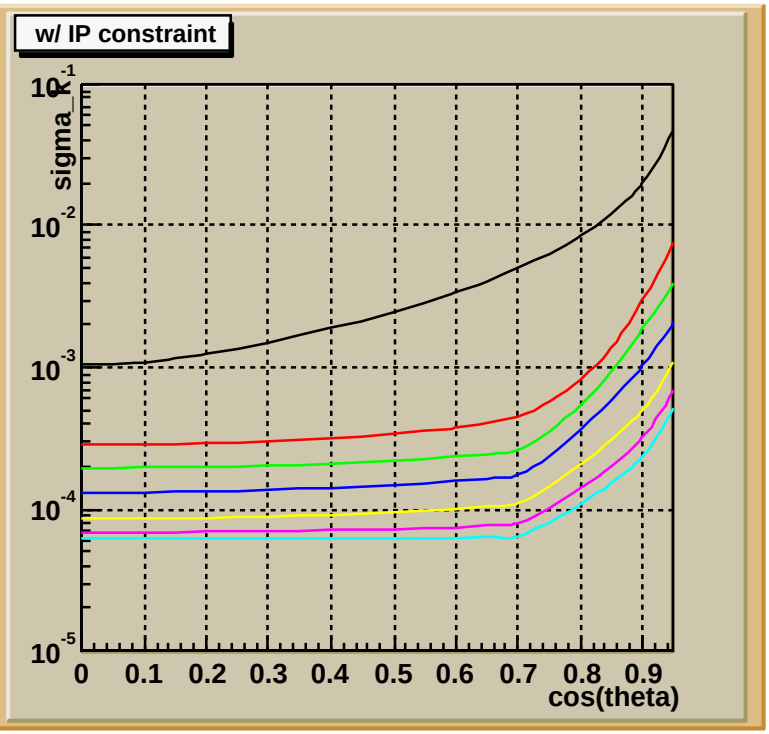
Without IMT





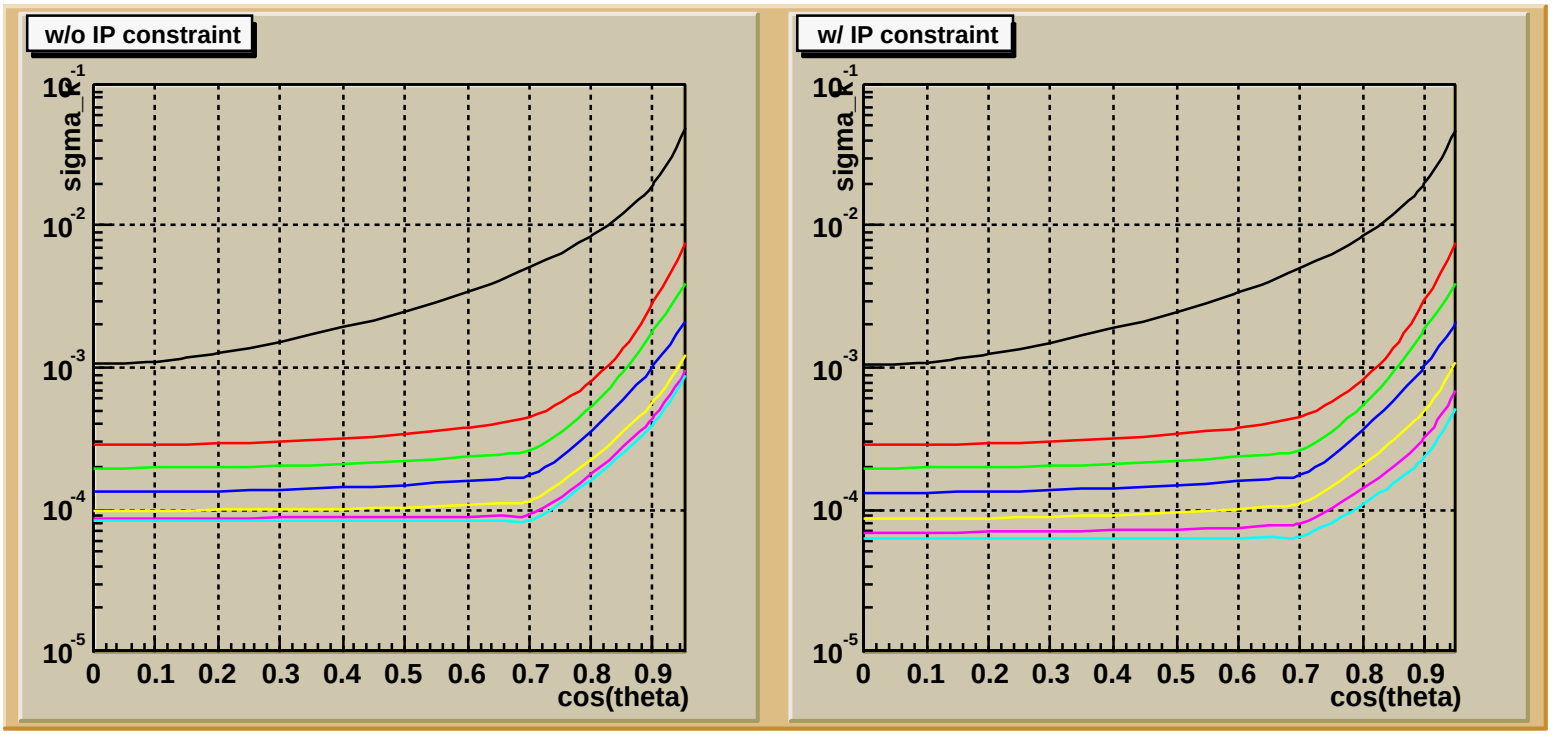
With IMT



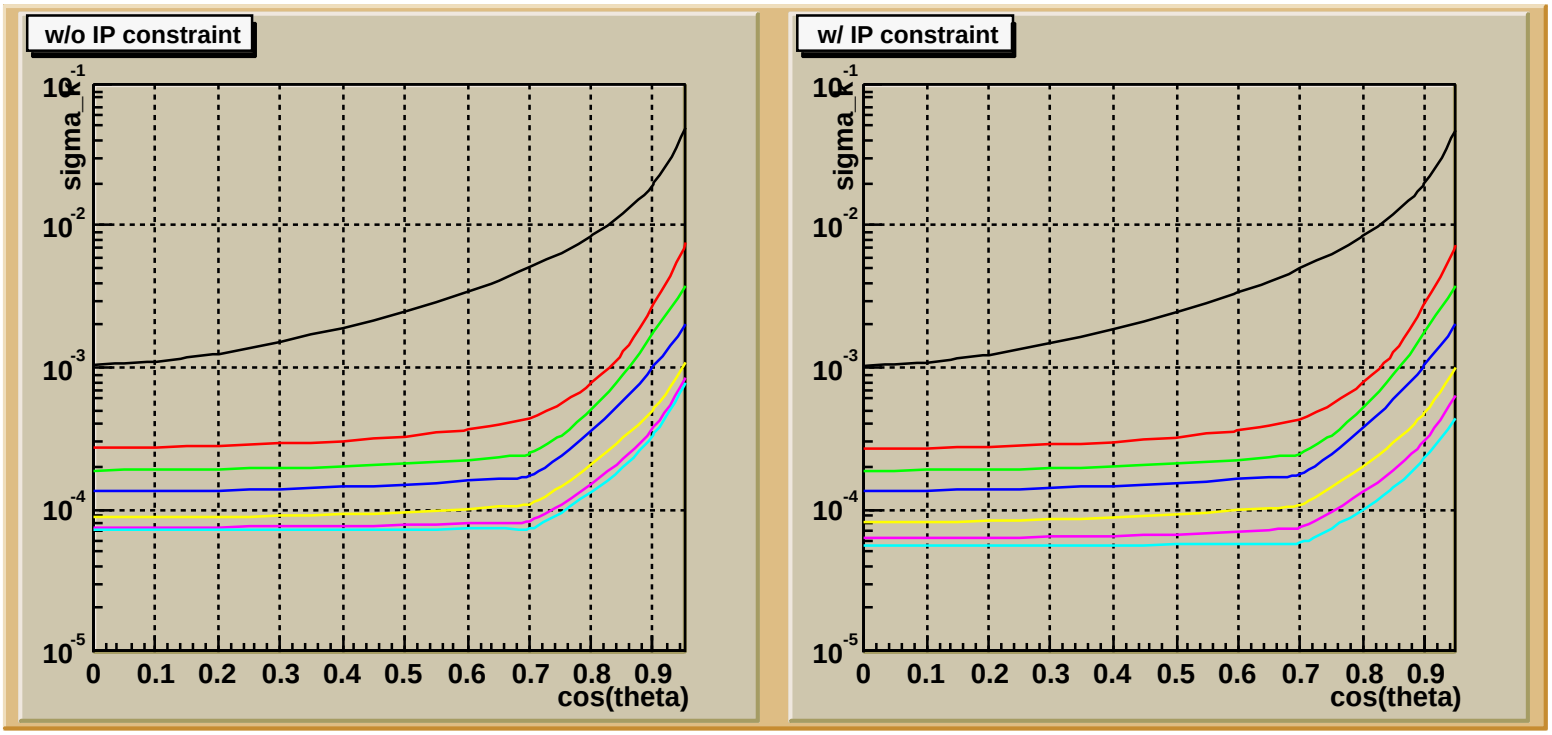


Effect of IMT Resolution

IMT Resolution $\sigma_{xy} = 40\mu m$



IMT Resolution $\sigma_{xy} = 20\mu m$



Summary

```
p = 50 GeV,  tan(lambda) = 0

CO2/C4H10
sgx(CDC) = 85 microns x  n samples : R = (45->45cm+L)
sgx(IST) = 40         x  5 layers  : R = (10,17,24,31,38cm)
sgx(VTX) =  4         x  4 layers  : R = (2.4,3.6,4.8,6.0cm)
```

L(cm)	IT	VT	IP	$\sigma_\kappa(\times 10^{-4})$ Materials w/ IT	$\sigma_\kappa(\times 10^{-4})$ Materials w/o IT
185 (n=80) (B=2T)	×	×	×	1.24	1.24
	×	×	●	0.62	0.60
	×	●	×	0.63	0.61
	×	●	●	0.61	0.58
	●	●	×	0.62	
	●	●	●	0.61	
110 (n=50) (B=3T)	×	×	×	2.86	2.86
	×	×	●	1.01	0.99
	×	●	×	1.01	0.99
	×	●	●	0.86	0.80
	●	●	×	0.96	
	●	●	●	0.84	