

Experiment 949
Technical Note K-062

A Correction to a `setup_pass1.F` Coding Mistake.

Benji Lewis

Abstract

While investigating the target reconstruction failures I discovered a bug in the `setup_pass1.F` `pass2` code. This note details the fix and the expected improvements due to the fix. This fix was implemented before the PNN2 1/3 ntuple production that was done in early 2006.

1 Introduction

Within the `setup_pass1` code the `utc` hits are fitted using `utc_track()`. The result of which are all possible track(s) solutions. Then `rdut_match(1,idctrk)` matches which `utc` track best matches the range stack (`idctrk` indicates which track was chosen). With this `utc` track the target can be reconstructed using `swathccd` or `TGrecon`.

After the target has been reconstructed, a refit is performed on the `utc` track employing the target hits as well as the `utc` hits associated with track indexed as `idctrk`. In addition to these hits, `utc` hits that are close to the previously chosen track will also be in the list of possible hits to find the best solution for a track. Limiting the hits in the manner described, there will be only one solution given by `utc_track()`. Hence `idctrk` should be reassigned to the value of 1, however this was not done. Instead the `idctrk` value from the earlier `rdut_match()` call is used.

No problem occurs if `rdut_match()` picks `idctrk = 1`. However, if `idctrk > 1` the track information is unreliable or stale (from the original `utc_track()` call) after the `utc` refitting. The information is stored correctly, but since `idctrk` is indexed incorrectly all other routines that use `idctrk` will be using faulty information. This includes `swathccd` which will lead to a target reconstruction failure.

2 The Fix

The fix is very simple. `setup_pass1` is modified by adding three lines. Shown in the following code segments with the comment `!04mar24 benjil`.

- Code segment 1

```
idctrk = 0
if(ntrack_d.ge.1) call rdut_match(1,idctrk)
orig_idctrk = idctrk !04mar24 benjil
```

- Code segment 2

```

if(ntthits(ic).ge.1) then
  utfalg=4
  call utc_track(.true.,.true.)
  call dcp1
  idctrk = 1    !04mar24 benjil
endif

```

- Code segment 3

```

if(.not.refit_ok) then
  utfalg=3
  call utc_track(.true.,.true.)
  call dcp1
  idctrk = orig_idctrk    !04mar24 benjil
endif

```

The added line in the first code segment is to save the original `idctrk` value as `orig_idctrk`, which is needed in the third code segment. The added line in the second code segment is to force `idctrk` to be 1 which is the only choice possible after a utc refit. The added line in code segment 3 is used during the rare occasion when the utc refit fails. When the refit fails `utc_track()` is called with all utc hits available without including the target hits and since we already know the outcome of `rdut_match()` there is no need to call it again, so just assign `idctrk = orig_idctrk`.

In this case, `utc_track` is called but `rdut_match()` is not needed since the match will be the same as the original `idctrk`.

3 Benefits from Fix

As mentioned earlier, on the occasion that `idctrk` was chosen to be greater than 1 the target reconstruction will fail. After the fix is implemented, these event could pass `swathccd`. Therefore, there should be an overall acceptance increase due to the code fix.

A sample of Km21 and PNN1/PNN2 events were processed before and after the code fix. The pass2 code used here is the most up-to-date version of the PNN2 pass2 code.

itgqualt	km21		%	pnn1/2		%
	before	after	change	before	after	change
0	1968	1998	1.52%	6718	6730	0.15%
1	225	225	0.0%	1871	1871	0.0%
5 & 6	763	769	0.79%	12692	12910	1.7%
with ps_kink cut applied						
0	1327	1346	1.43%	3160	3180	0.63%
1	49	49	0.0%	80	80	0.0%
5 & 6	78	77	-1.29%	60	60	0.0%

The table shows the number of events that were successfully reconstructed in the target. `itgqualt = 0` and `1` are reconstructions by *swathccd* and `itgqualt = 5` and `6` are reconstructions by *TGrecon*. The columns labeled as before refer to before the bug fix and after refers to after the bug fix. The setup cuts used here are `ndclay = 0` and `kinkqual \neq 1` (cut target scatters). Also, additionally applied are the `ps_kink` cuts. `ps_kink` cuts are the same as `pscut02` in the PNN1 analysis except that the cuts that are dependent upon information in the target has been removed.

4 Conclusions

This code fix is now in the PNN2 version of the pass2 code. Be aware that the code mistake was discovered after the completion of the PNN1 analysis. Although, a very minor coding problem has been fixed a possible acceptance gain of about 1% could possibly be realized.