Subject: interpolation in 5 dimensional space (how and speed)
Posted by pdeyoung on Wed, 04 May 2005 22:59:17 GMT
View Forum Message <> Reply to Message

We have a project where we track a large number of test particles through a magnetic field. Then using the ending variables (five) we want to infer the starting parameters for real particles by comparing them to the nearest test particles. (Please don't laugh at my efforts below - I really don't know the correct way to do this.) In some sense the one dimensional analogy would be to have a y value and find the x values given that you could calculate the y's from a grid of x values ahead of time. Of course in the one-d example there is no assurance that the y's will be equally spaced. Similarly, in the code below the results of the tracks are not equally spaced in output space. (In the code below for simplicity while testing, I just use random arrays.) For background, I found the closest point in each "quadrant" and then found a weighted average based on the distance from the test point and the closest points. (This could be totally bogus.) Anyway, is there a better (and faster) way to do this or is this approach reasonable. If so, is there a way to do it faster. Ultimately we will have to do this 10^6 times for the real data set. I am using IDL6.1

Thanks in advance.

Paul DeYoung
deyoung@hope.edu

```
pro test_array_v2

;final position that needs to be mapped to input variables
x = .612
y = .712
z = .812
v = .312
w = .412

ranarray_1 = RANDOMU(seed,[10,10,10,10,10])
ranarray_2 = RANDOMU(seed,[10,10,10,10,10])
ranarray_3 = RANDOMU(seed,[10,10,10,10,10])
ranarray_4 = RANDOMU(seed,[10,10,10,10,10])
ranarray_5 = RANDOMU(seed,[10,10,10,10,10])



T = SYSTIME(1)
```

```
quad1 = where(ranarray_1 GE x and ranarray_2 GE y and ranarray_3 GE z
and ranarray_4 GE v and ranarray_5 GE w,num1)
quad2 = where(ranarray_1 GE x and ranarray_2 GE y and ranarray_3 GE z
and ranarray_4 GE v and ranarray_5 LT w,num2)
quad3 = where(ranarray_1 GE x and ranarray_2 GE y and ranarray_3 GE z
and ranarray_4 LT v and ranarray_5 GE w,num3)
quad4 = where(ranarray_1 GE x and ranarray_2 GE y and ranarray_3 GE z
and ranarray_4 LT v and ranarray_5 LT w,num4)
quad5 = where(ranarray_1 GE x and ranarray_2 GE y and ranarray_3 LT z
and ranarray_4 GE v and ranarray_5 GE w,num5)
quad6 = where(ranarray_1 GE x and ranarray_2 GE y and ranarray_3 LT z
and ranarray_4 GE v and ranarray_5 LT w,num6)
quad7 = where(ranarray_1 GE x and ranarray_2 GE y and ranarray_3 LT z
and ranarray_4 LT v and ranarray_5 GE w,num7)
quad8 = where(ranarray_1 GE x and ranarray_2 GE y and ranarray_3 LT z
and ranarray_4 LT v and ranarray_5 LT w,num8)
quad9 = where(ranarray_1 GE x and ranarray_2 LT y and ranarray_3 GE z
and ranarray_4 GE v and ranarray_5 GE w,num9)
quad10 = where(ranarray_1 GE x and ranarray_2 LT y and ranarray_3 GE z
and ranarray_4 GE v and ranarray_5 LT w,num10)
quad11 = where(ranarray_1 GE x and ranarray_2 LT y and ranarray_3 GE z
and ranarray_4 LT v and ranarray_5 GE w,num11)
quad12 = where(ranarray_1 GE x and ranarray_2 LT y and ranarray_3 GE z
and ranarray_4 LT v and ranarray_5 LT w,num12)
quad13 = where(ranarray_1 GE x and ranarray_2 LT y and ranarray_3 LT z
and ranarray_4 GE v and ranarray_5 GE w,num13)
quad14 = where(ranarray_1 GE x and ranarray_2 LT y and ranarray_3 LT z
and ranarray_4 GE v and ranarray_5 LT w,num14)
quad15 = where(ranarray_1 GE x and ranarray_2 LT y and ranarray_3 LT z
and ranarray_4 LT v and ranarray_5 GE w,num15)
quad16 = where(ranarray_1 GE x and ranarray_2 LT y and ranarray_3 LT z
and ranarray_4 LT v and ranarray_5 LT w,num16)
quad17 = where(ranarray_1 LT x and ranarray_2 GE y and ranarray_3 GE z
and ranarray_4 GE v and ranarray_5 GE w,num17)
quad18 = where(ranarray_1 LT x and ranarray_2 GE y and ranarray_3 GE z
and ranarray_4 GE v and ranarray_5 LT w,num18)
quad19 = where(ranarray_1 LT x and ranarray_2 GE y and ranarray_3 GE z
and ranarray_4 LT v and ranarray_5 GE w,num19)
quad20 = where(ranarray_1 LT x and ranarray_2 GE y and ranarray_3 GE z
and ranarray_4 LT v and ranarray_5 LT w,num20)
quad21 = where(ranarray_1 LT x and ranarray_2 GE y and ranarray_3 LT z
and ranarray_4 GE v and ranarray_5 GE w,num21)
quad22 = where(ranarray_1 LT x and ranarray_2 GE y and ranarray_3 LT z
and ranarray_4 GE v and ranarray_5 LT w,num22)
quad23 = where(ranarray_1 LT x and ranarray_2 GE y and ranarray_3 LT z
and ranarray_4 LT v and ranarray_5 GE w,num23)
quad24 = where(ranarray_1 LT x and ranarray_2 GE y and ranarray_3 LT z
and ranarray_4 LT v and ranarray_5 LT w,num24)
```

```
quad25 = where(ranarray_1 LT x and ranarray_2 LT y and ranarray_3 GE z
and ranarray_4 GE v and ranarray_5 GE w,num25)
quad26 = where(ranarray_1 LT x and ranarray_2 LT y and ranarray_3 GE z
and ranarray_4 GE v and ranarray_5 LT w,num26)
quad27 = where(ranarray_1 LT x and ranarray_2 LT y and ranarray_3 GE z
and ranarray_4 LT v and ranarray_5 GE w,num27)
quad28 = where(ranarray_1 LT x and ranarray_2 LT y and ranarray_3 GE z
and ranarray_4 LT v and ranarray_5 LT w,num28)
quad29 = where(ranarray_1 LT x and ranarray_2 LT y and ranarray_3 LT z
and ranarray_4 GE v and ranarray_5 GE w,num29)
quad30 = where(ranarray_1 LT x and ranarray_2 LT y and ranarray_3 LT z
and ranarray_4 GE v and ranarray_5 LT w,num30)
quad31 = where(ranarray_1 LT x and ranarray_2 LT y and ranarray_3 LT z
and ranarray_4 LT v and ranarray_5 GE w,num31)
quad32 = where(ranarray_1 LT x and ranarray_2 LT y and ranarray_3 LT z
and ranarray_4 LT v and ranarray_5 LT w,num32)

;print,num1,num2,num3,num4,num5,num6,num7,num8
;PRINT, SYSTIME(1) - T, 'Seconds'

dist_squared_1 = (ranarray_1(quad1)-x)^2 + (ranarray_2(quad1)-y)^2 +
(ranarray_3(quad1)-z)^2 + (ranarray_4(quad1)-v)^2 +
(ranarray_5(quad1)-w)^2
min_dist_1 = min(dist_squared_1,index_1)
dist_squared_2 = (ranarray_1(quad2)-x)^2 + (ranarray_2(quad2)-y)^2 +
(ranarray_3(quad2)-z)^2 + (ranarray_4(quad2)-v)^2 +
(ranarray_5(quad2)-w)^2
min_dist_2 = min(dist_squared_2,index_2)
dist_squared_3 = (ranarray_1(quad3)-x)^2 + (ranarray_2(quad3)-y)^2 +
(ranarray_3(quad3)-z)^2 + (ranarray_4(quad3)-v)^2 +
(ranarray_5(quad3)-w)^2
min_dist_3 = min(dist_squared_3,index_3)
dist_squared_4 = (ranarray_1(quad4)-x)^2 + (ranarray_2(quad4)-y)^2 +
(ranarray_3(quad4)-z)^2 + (ranarray_4(quad4)-v)^2 +
(ranarray_5(quad4)-w)^2
min_dist_4 = min(dist_squared_4,index_4)
dist_squared_5 = (ranarray_1(quad5)-x)^2 + (ranarray_2(quad5)-y)^2 +
(ranarray_3(quad5)-z)^2 + (ranarray_4(quad5)-v)^2 +
(ranarray_5(quad5)-w)^2
min_dist_5 = min(dist_squared_5,index_5)
dist_squared_6 = (ranarray_1(quad6)-x)^2 + (ranarray_2(quad6)-y)^2 +
(ranarray_3(quad6)-z)^2 + (ranarray_4(quad6)-v)^2 +
(ranarray_5(quad6)-w)^2
min_dist_6 = min(dist_squared_6,index_6)
dist_squared_7 = (ranarray_1(quad7)-x)^2 + (ranarray_2(quad7)-y)^2 +
(ranarray_3(quad7)-z)^2 + (ranarray_4(quad7)-v)^2 +
(ranarray_5(quad7)-w)^2
min_dist_7 = min(dist_squared_7,index_7)
```

```
dist_squared_8 = (ranarray_1(quad8)-x)^2 + (ranarray_2(quad8)-y)^2 +
(ranarray_3(quad8)-z)^2 + (ranarray_4(quad8)-v)^2 +
(ranarray_5(quad8)-w)^2
min_dist_8 = min(dist_squared_8,index_8)
dist_squared_9 = (ranarray_1(quad9)-x)^2 + (ranarray_2(quad9)-y)^2 +
(ranarray_3(quad9)-z)^2 + (ranarray_4(quad9)-v)^2 +
(ranarray_5(quad9)-w)^2
min_dist_9 = min(dist_squared_9,index_9)
dist_squared_10 = (ranarray_1(quad10)-x)^2 + (ranarray_2(quad10)-y)^2 +
(ranarray_3(quad10)-z)^2 + (ranarray_4(quad10)-v)^2 +
(ranarray_5(quad10)-w)^2
min_dist_10 = min(dist_squared_10,index_10);
dist_squared_11 = (ranarray_1(quad11)-x)^2 + (ranarray_2(quad11)-y)^2 +
(ranarray_3(quad11)-z)^2 + (ranarray_4(quad11)-v)^2 +
(ranarray_5(quad11)-w)^2
min_dist_11 = min(dist_squared_11,index_11)
dist_squared_12 = (ranarray_1(quad12)-x)^2 + (ranarray_2(quad12)-y)^2 +
(ranarray_3(quad12)-z)^2 + (ranarray_4(quad12)-v)^2 +
(ranarray_5(quad12)-w)^2
min_dist_12 = min(dist_squared_12,index_12)
dist_squared_13 = (ranarray_1(quad13)-x)^2 + (ranarray_2(quad13)-y)^2 +
(ranarray_3(quad13)-z)^2 + (ranarray_4(quad13)-v)^2 +
(ranarray_5(quad13)-w)^2
min_dist_13 = min(dist_squared_13,index_13)
dist_squared_14 = (ranarray_1(quad14)-x)^2 + (ranarray_2(quad14)-y)^2 +
(ranarray_3(quad14)-z)^2 + (ranarray_4(quad14)-v)^2 +
(ranarray_5(quad14)-w)^2
min_dist_14 = min(dist_squared_14,index_14)
dist_squared_15 = (ranarray_1(quad15)-x)^2 + (ranarray_2(quad15)-y)^2 +
(ranarray_3(quad15)-z)^2 + (ranarray_4(quad15)-v)^2 +
(ranarray_5(quad15)-w)^2
min_dist_15 = min(dist_squared_15,index_15)
dist_squared_16 = (ranarray_1(quad16)-x)^2 + (ranarray_2(quad16)-y)^2 +
(ranarray_3(quad16)-z)^2 + (ranarray_4(quad16)-v)^2 +
(ranarray_5(quad16)-w)^2
min_dist_16 = min(dist_squared_16,index_16)
dist_squared_17 = (ranarray_1(quad17)-x)^2 + (ranarray_2(quad17)-y)^2 +
(ranarray_3(quad17)-z)^2 + (ranarray_4(quad17)-v)^2 +
(ranarray_5(quad17)-w)^2
min_dist_17 = min(dist_squared_17,index_17)
dist_squared_18 = (ranarray_1(quad18)-x)^2 + (ranarray_2(quad18)-y)^2 +
(ranarray_3(quad18)-z)^2 + (ranarray_4(quad18)-v)^2 +
(ranarray_5(quad18)-w)^2
min_dist_18 = min(dist_squared_18,index_18)
dist_squared_19 = (ranarray_1(quad19)-x)^2 + (ranarray_2(quad19)-y)^2 +
(ranarray_3(quad19)-z)^2 + (ranarray_4(quad19)-v)^2 +
(ranarray_5(quad19)-w)^2
min_dist_19 = min(dist_squared_19,index_19)
```

```
dist_squared_20 = (ranarray_1(quad20)-x)^2 + (ranarray_2(quad20)-y)^2 +
(ranarray_3(quad20)-z)^2 + (ranarray_4(quad20)-v)^2 +
(ranarray_5(quad20)-w)^2
min_dist_20 = min(dist_squared_20,index_20)
dist_squared_21 = (ranarray_1(quad21)-x)^2 + (ranarray_2(quad21)-y)^2 +
(ranarray_3(quad21)-z)^2 + (ranarray_4(quad21)-v)^2 +
(ranarray_5(quad21)-w)^2
min_dist_21 = min(dist_squared_21,index_21)
dist_squared_22 = (ranarray_1(quad22)-x)^2 + (ranarray_2(quad22)-y)^2 +
(ranarray_3(quad22)-z)^2 + (ranarray_4(quad22)-v)^2 +
(ranarray_5(quad22)-w)^2
min_dist_22 = min(dist_squared_22,index_22)
dist_squared_23 = (ranarray_1(quad23)-x)^2 + (ranarray_2(quad23)-y)^2 +
(ranarray_3(quad23)-z)^2 + (ranarray_4(quad23)-v)^2 +
(ranarray_5(quad23)-w)^2
min_dist_23 = min(dist_squared_23,index_23)
dist_squared_24 = (ranarray_1(quad24)-x)^2 + (ranarray_2(quad24)-y)^2 +
(ranarray_3(quad24)-z)^2 + (ranarray_4(quad24)-v)^2 +
(ranarray_5(quad24)-w)^2
min_dist_24 = min(dist_squared_24,index_24)
dist_squared_25 = (ranarray_1(quad25)-x)^2 + (ranarray_2(quad25)-y)^2 +
(ranarray_3(quad25)-z)^2 + (ranarray_4(quad25)-v)^2 +
(ranarray_5(quad25)-w)^2
min_dist_25 = min(dist_squared_25,index_25)
dist_squared_26 = (ranarray_1(quad26)-x)^2 + (ranarray_2(quad26)-y)^2 +
(ranarray_3(quad26)-z)^2 + (ranarray_4(quad26)-v)^2 +
(ranarray_5(quad26)-w)^2
min_dist_26 = min(dist_squared_26,index_26)
dist_squared_27 = (ranarray_1(quad27)-x)^2 + (ranarray_2(quad27)-y)^2 +
(ranarray_3(quad27)-z)^2 + (ranarray_4(quad27)-v)^2 +
(ranarray_5(quad27)-w)^2
min_dist_27 = min(dist_squared_27,index_27)
dist_squared_28 = (ranarray_1(quad28)-x)^2 + (ranarray_2(quad28)-y)^2 +
(ranarray_3(quad28)-z)^2 + (ranarray_4(quad28)-v)^2 +
(ranarray_5(quad28)-w)^2
min_dist_28 = min(dist_squared_28,index_28)
dist_squared_29 = (ranarray_1(quad29)-x)^2 + (ranarray_2(quad29)-y)^2 +
(ranarray_3(quad29)-z)^2 + (ranarray_4(quad29)-v)^2 +
(ranarray_5(quad29)-w)^2
min_dist_29 = min(dist_squared_29,index_29)
dist_squared_30 = (ranarray_1(quad30)-x)^2 + (ranarray_2(quad30)-y)^2 +
(ranarray_3(quad30)-z)^2 + (ranarray_4(quad30)-v)^2 +
(ranarray_5(quad30)-w)^2
min_dist_30 = min(dist_squared_30,index_30)
dist_squared_31 = (ranarray_1(quad31)-x)^2 + (ranarray_2(quad31)-y)^2 +
(ranarray_3(quad31)-z)^2 + (ranarray_4(quad31)-v)^2 +
(ranarray_5(quad31)-w)^2
min_dist_31 = min(dist_squared_31,index_31)
```

```
dist_squared_32 = (ranarray_1(quad32)-x)^2 + (ranarray_2(quad32)-y)^2 +
(ranarray_3(quad32)-z)^2 + (ranarray_4(quad32)-v)^2 +
(ranarray_5(quad32)-w)^2
min_dist_32 = min(dist_squared_32,index_32)
;print,min_dist_1,index_1,quad1(index_1)
;print,min_dist_2,index_2,quad2(index_2)
;print,min_dist_3,index_3,quad3(index_3)
;print,min_dist_4,index_4,quad4(index_4)
;print,min_dist_5,index_5,quad5(index_5)
;print,min_dist_6,index_6,quad6(index_6)
;print,min_dist_7,index_7,quad7(index_7)
;print,min_dist_8,index_8,quad8(index_8)
;PRINT, SYSTIME(1) - T, 'Seconds'

indices1 = ARRAY_INDICES(ranarray_1, quad1(index_1))
indices2 = ARRAY_INDICES(ranarray_1, quad2(index_2))
indices3 = ARRAY_INDICES(ranarray_1, quad3(index_3))
indices4 = ARRAY_INDICES(ranarray_1, quad4(index_4))
indices5 = ARRAY_INDICES(ranarray_1, quad5(index_5))
indices6 = ARRAY_INDICES(ranarray_1, quad6(index_6))
indices7 = ARRAY_INDICES(ranarray_1, quad7(index_7))
indices8 = ARRAY_INDICES(ranarray_1, quad8(index_8))
indices9 = ARRAY_INDICES(ranarray_1, quad9(index_9))
indices10 = ARRAY_INDICES(ranarray_1, quad10(index_10))
indices11 = ARRAY_INDICES(ranarray_1, quad11(index_11))
indices12 = ARRAY_INDICES(ranarray_1, quad12(index_12))
indices13 = ARRAY_INDICES(ranarray_1, quad13(index_13))
indices14 = ARRAY_INDICES(ranarray_1, quad14(index_14))
indices15 = ARRAY_INDICES(ranarray_1, quad15(index_15))
indices16 = ARRAY_INDICES(ranarray_1, quad16(index_16))
indices17 = ARRAY_INDICES(ranarray_1, quad17(index_17))
indices18 = ARRAY_INDICES(ranarray_1, quad18(index_18))
indices19 = ARRAY_INDICES(ranarray_1, quad19(index_19))
indices20 = ARRAY_INDICES(ranarray_1, quad20(index_20))
indices21 = ARRAY_INDICES(ranarray_1, quad21(index_21))
indices22 = ARRAY_INDICES(ranarray_1, quad22(index_22))
indices23 = ARRAY_INDICES(ranarray_1, quad23(index_23))
indices24 = ARRAY_INDICES(ranarray_1, quad24(index_24))
indices25 = ARRAY_INDICES(ranarray_1, quad25(index_25))
indices26 = ARRAY_INDICES(ranarray_1, quad26(index_26))
indices27 = ARRAY_INDICES(ranarray_1, quad27(index_27))
indices28 = ARRAY_INDICES(ranarray_1, quad28(index_28))
indices29 = ARRAY_INDICES(ranarray_1, quad29(index_29))
indices30 = ARRAY_INDICES(ranarray_1, quad30(index_30))
indices31 = ARRAY_INDICES(ranarray_1, quad31(index_31))
indices32 = ARRAY_INDICES(ranarray_1, quad32(index_32))

;print,indices1
```

```
;print,indices2
;print,indices3
;print,indices4
;print,indices5
;print,indices6
;print,indices7
;print,indices8
;PRINT, SYSTIME(1) - T, 'Seconds'

;get the weighted x index

weight = (1/sqrt(min_dist_1) + 1/sqrt(min_dist_2) + $
          1/sqrt(min_dist_3) + 1/sqrt(min_dist_4) + $
          1/sqrt(min_dist_5) + 1/sqrt(min_dist_6) + $
          1/sqrt(min_dist_7) + 1/sqrt(min_dist_8) + $
          1/sqrt(min_dist_9) + 1/sqrt(min_dist_10) + $
          1/sqrt(min_dist_11) + 1/sqrt(min_dist_12) + $
          1/sqrt(min_dist_13) + 1/sqrt(min_dist_14) + $
          1/sqrt(min_dist_15) + 1/sqrt(min_dist_16) + $
          1/sqrt(min_dist_17) + 1/sqrt(min_dist_18) + $
          1/sqrt(min_dist_19) + 1/sqrt(min_dist_20) + $
          1/sqrt(min_dist_21) + 1/sqrt(min_dist_22) + $
          1/sqrt(min_dist_23) + 1/sqrt(min_dist_24) + $
          1/sqrt(min_dist_25) + 1/sqrt(min_dist_26) + $
          1/sqrt(min_dist_27) + 1/sqrt(min_dist_28) + $
          1/sqrt(min_dist_29) + 1/sqrt(min_dist_30) + $
          1/sqrt(min_dist_31) + 1/sqrt(min_dist_32))
xinterp = (indices1[0]/sqrt(min_dist_1)+indices2[0]/sqrt(min_dist_2) +
$
        indices3[0]/sqrt(min_dist_3)+indices4[0]/sqrt(min_dist_4) +
$
        indices5[0]/sqrt(min_dist_5)+indices6[0]/sqrt(min_dist_6) +
$
        indices7[0]/sqrt(min_dist_7)+indices8[0]/sqrt(min_dist_8) +
$
        indices9[0]/sqrt(min_dist_9)+indices10[0]/sqrt(min_dist_10)
+ $

 indices11[0]/sqrt(min_dist_11)+indices12[0]/sqrt(min_dist_12 ) + $

 indices13[0]/sqrt(min_dist_13)+indices14[0]/sqrt(min_dist_14 ) + $

 indices15[0]/sqrt(min_dist_15)+indices16[0]/sqrt(min_dist_16 ) + $

 indices17[0]/sqrt(min_dist_17)+indices18[0]/sqrt(min_dist_18 ) + $

 indices19[0]/sqrt(min_dist_19)+indices20[0]/sqrt(min_dist_20 ) + $
```

```
 indices21[0]/sqrt(min_dist_21)+indices22[0]/sqrt(min_dist_22 ) + $

 indices23[0]/sqrt(min_dist_23)+indices24[0]/sqrt(min_dist_24 ) + $

 indices25[0]/sqrt(min_dist_25)+indices26[0]/sqrt(min_dist_26 ) + $

 indices27[0]/sqrt(min_dist_27)+indices28[0]/sqrt(min_dist_28 ) + $

 indices29[0]/sqrt(min_dist_29)+indices30[0]/sqrt(min_dist_30 ) + $

 indices31[0]/sqrt(min_dist_31)+indices32[0]/sqrt(min_dist_32 )) / $
        (weight)
yinterp = (indices1[1]/sqrt(min_dist_1)+indices2[1]/sqrt(min_dist_2) +
$
        indices3[1]/sqrt(min_dist_3)+indices4[1]/sqrt(min_dist_4) +
$
        indices5[1]/sqrt(min_dist_5)+indices6[1]/sqrt(min_dist_6) +
$
        indices7[1]/sqrt(min_dist_7)+indices8[1]/sqrt(min_dist_8) +
$
        indices9[1]/sqrt(min_dist_9)+indices10[1]/sqrt(min_dist_10)
+ $

 indices11[1]/sqrt(min_dist_11)+indices12[1]/sqrt(min_dist_12 ) + $

 indices13[1]/sqrt(min_dist_13)+indices14[1]/sqrt(min_dist_14 ) + $

 indices15[1]/sqrt(min_dist_15)+indices16[1]/sqrt(min_dist_16 ) + $

 indices17[1]/sqrt(min_dist_17)+indices18[1]/sqrt(min_dist_18 ) + $

 indices19[1]/sqrt(min_dist_19)+indices20[1]/sqrt(min_dist_20 ) + $

 indices21[1]/sqrt(min_dist_21)+indices22[1]/sqrt(min_dist_22 ) + $

 indices23[1]/sqrt(min_dist_23)+indices24[1]/sqrt(min_dist_24 ) + $

 indices25[1]/sqrt(min_dist_25)+indices26[1]/sqrt(min_dist_26 ) + $

 indices27[1]/sqrt(min_dist_27)+indices28[1]/sqrt(min_dist_28 ) + $

 indices29[1]/sqrt(min_dist_29)+indices30[1]/sqrt(min_dist_30 ) + $

 indices31[1]/sqrt(min_dist_31)+indices32[1]/sqrt(min_dist_32 )) / $
        (weight)
zinterp = (indices1[2]/sqrt(min_dist_1)+indices2[2]/sqrt(min_dist_2) +
$
        indices3[2]/sqrt(min_dist_3)+indices4[2]/sqrt(min_dist_4) +
```

```
$
        indices5[2]/sqrt(min_dist_5)+indices6[2]/sqrt(min_dist_6) +
$
        indices7[2]/sqrt(min_dist_7)+indices8[2]/sqrt(min_dist_8) +
$
        indices9[2]/sqrt(min_dist_9)+indices10[2]/sqrt(min_dist_10)
+ $

 indices11[2]/sqrt(min_dist_11)+indices12[2]/sqrt(min_dist_12 ) + $

 indices13[2]/sqrt(min_dist_13)+indices14[2]/sqrt(min_dist_14 ) + $

 indices15[2]/sqrt(min_dist_15)+indices16[2]/sqrt(min_dist_16 ) + $

 indices17[2]/sqrt(min_dist_17)+indices18[2]/sqrt(min_dist_18 ) + $

 indices19[2]/sqrt(min_dist_19)+indices20[2]/sqrt(min_dist_20 ) + $

 indices21[2]/sqrt(min_dist_21)+indices22[2]/sqrt(min_dist_22 ) + $

 indices23[2]/sqrt(min_dist_23)+indices24[2]/sqrt(min_dist_24 ) + $

 indices25[2]/sqrt(min_dist_25)+indices26[2]/sqrt(min_dist_26 ) + $

 indices27[2]/sqrt(min_dist_27)+indices28[2]/sqrt(min_dist_28 ) + $

 indices29[2]/sqrt(min_dist_29)+indices30[2]/sqrt(min_dist_30 ) + $

 indices31[2]/sqrt(min_dist_31)+indices32[2]/sqrt(min_dist_32 )) / $
        (weight)
vinterp = (indices1[3]/sqrt(min_dist_1)+indices2[3]/sqrt(min_dist_2) +
$
        indices3[3]/sqrt(min_dist_3)+indices4[3]/sqrt(min_dist_4) +
$
        indices5[3]/sqrt(min_dist_5)+indices6[3]/sqrt(min_dist_6) +
$
        indices7[3]/sqrt(min_dist_7)+indices8[3]/sqrt(min_dist_8) +
$
        indices9[3]/sqrt(min_dist_9)+indices10[3]/sqrt(min_dist_10)
+ $

 indices11[3]/sqrt(min_dist_11)+indices12[3]/sqrt(min_dist_12 ) + $

 indices13[3]/sqrt(min_dist_13)+indices14[3]/sqrt(min_dist_14 ) + $

 indices15[3]/sqrt(min_dist_15)+indices16[3]/sqrt(min_dist_16 ) + $

 indices17[3]/sqrt(min_dist_17)+indices18[3]/sqrt(min_dist_18 ) + $
```

```
 indices19[3]/sqrt(min_dist_19)+indices20[3]/sqrt(min_dist_20 ) + $

 indices21[3]/sqrt(min_dist_21)+indices22[3]/sqrt(min_dist_22 ) + $

 indices23[3]/sqrt(min_dist_23)+indices24[3]/sqrt(min_dist_24 ) + $

 indices25[3]/sqrt(min_dist_25)+indices26[3]/sqrt(min_dist_26 ) + $

 indices27[3]/sqrt(min_dist_27)+indices28[3]/sqrt(min_dist_28 ) + $

 indices29[3]/sqrt(min_dist_29)+indices30[3]/sqrt(min_dist_30 ) + $

 indices31[3]/sqrt(min_dist_31)+indices32[3]/sqrt(min_dist_32 )) / $
        (weight)
winterp = (indices1[4]/sqrt(min_dist_1)+indices2[4]/sqrt(min_dist_2) +
$
        indices3[4]/sqrt(min_dist_3)+indices4[4]/sqrt(min_dist_4) +
$
        indices5[4]/sqrt(min_dist_5)+indices6[4]/sqrt(min_dist_6) +
$
        indices7[4]/sqrt(min_dist_7)+indices8[4]/sqrt(min_dist_8) +
$
        indices9[4]/sqrt(min_dist_9)+indices10[4]/sqrt(min_dist_10)
+ $

 indices11[4]/sqrt(min_dist_11)+indices12[4]/sqrt(min_dist_12 ) + $

 indices13[4]/sqrt(min_dist_13)+indices14[4]/sqrt(min_dist_14 ) + $

 indices15[4]/sqrt(min_dist_15)+indices16[4]/sqrt(min_dist_16 ) + $

 indices17[4]/sqrt(min_dist_17)+indices18[4]/sqrt(min_dist_18 ) + $

 indices19[4]/sqrt(min_dist_19)+indices20[4]/sqrt(min_dist_20 ) + $

 indices21[4]/sqrt(min_dist_21)+indices22[4]/sqrt(min_dist_22 ) + $

 indices23[4]/sqrt(min_dist_23)+indices24[4]/sqrt(min_dist_24 ) + $

 indices25[4]/sqrt(min_dist_25)+indices26[4]/sqrt(min_dist_26 ) + $

 indices27[4]/sqrt(min_dist_27)+indices28[4]/sqrt(min_dist_28 ) + $

 indices29[4]/sqrt(min_dist_29)+indices30[4]/sqrt(min_dist_30 ) + $

 indices31[4]/sqrt(min_dist_31)+indices32[4]/sqrt(min_dist_32 )) / $
        (weight)
```

```
PRINT, SYSTIME(1) - T, 'Seconds'
print,xinterp,yinterp,zinterp,vinterp,winterp


end
```