ACDIV-2018-03
March 2018

Non-paraxial drift pass method

Oscar Jiménez Arranz, Zeus Martí

Abstract

A new pass method named DriftNoParaxPass, which simulates the trajectory of an electron in a straight path given the initial conditions, has been created. That is simulated too in DriftPass but considering paraxial approximation. This approximation consists on:
1. x; y << _
2. x0; y0 << 1
3. _ << 1
The first part of the report consists on the derivation of the non-paraxial formulas used in the pass method. Then a numerical comparison between the two methods is done in three different ways: comparing trajectories, tunes and tracking computation times for the ALBA lattice. Finally, in the annex we will prove that imposing the paraxial approximation we arrive to the same formulas used in DriftPass.

Accelerator Division
Alba Synchrotron Light Source
c/ de la Llum, 2-26
08290 Cerdanyola del Valles, Spain

Non-paraxial drift pass method

Author: Óscar Jiménez Arranz
Supervisor: Zeus Martí Díaz

March 15, 2018

1 Introduction

A new pass method named ***DriftNoParaxPass***, which simulates the trajectory of an electron in a straight path given the initial conditions, has been created. That is simulated too in *DriftPass* but considering paraxial approximation. This approximation consists on:

1. $x, y << \rho$
2. $x', y' << 1$
3. $\delta << 1$

The first part of the report consists on the derivation of the non-paraxial formulas used in the pass method. Then a numerical comparison between the two methods is done in three different ways: comparing **trajectories**, **tunes** and **tracking computation times** for the ALBA lattice. Finally, in the annex we will prove that imposing the paraxial approximation we arrive to the same formulas used in *DriftPass*.

2 Theoretical development

First of all we need to know which are the initial parameters we know. We have as an input the 6 dimensional vector that is commonly used in all pass methods: $[x_0\ x_0'\ y_0\ y_0'\ \delta_0\ s_0]$ where $' = \frac{d}{ds_{id}}$, the ideal energy E_{id}, and l is the drift's length.

Now we can introduce all the formulas we will need:

$$|\vec{p}| = \frac{1}{c}\sqrt{E^2 - m^2c^4} \tag{1}$$

$$E = \gamma mc^2 \tag{2}$$

$$p^\mu = m\gamma(c, v^x, v^y, v^s) \tag{3}$$

$$\gamma = \frac{1}{\sqrt{1 - (\frac{v}{c})^2}} \tag{4}$$

$$x' = \frac{p_x}{|\vec{p_{id}}|} \tag{5}$$

$$\Delta s = -\frac{1}{\beta_{id}}(l - v_{id}t) \tag{6}$$

The first and the second formulas come from the value of the energy for a relativistic particle. The third formula is the relation between velocity and momentum. The fourth formula is the definition of Lorentz coefficient. Finally, the fifth formula is the definition of the canonic variable x', and the sixth formula is the definition of the canonical momentum associated to the δ variable -see Nadolsky thesis [1].

2.1 Mathematical development

First of all we will determine the time that the particle takes to travel along the whole section. This is determined by:

$$t = \frac{l}{v_s} \tag{7}$$

Now we use equation (2) in order to determine Lorentz factor for two particles. The real particle -the one we want to determine its movement- and the ideal particle -that is, the one with no deviation and with the ideal energy:

$$\gamma = \frac{E_{id}(1+\delta)}{mc^2}, \qquad \gamma_{id} = \frac{E_{id}}{mc^2} \tag{8}$$

where we have taken into account that $E = E_{id}(1+\delta)$. If we want to obtain v_x, v_y as function of x', y' we can use equation (1) and (3) respectively in order to archive the following:

$$x' = \frac{p_x}{|\vec{p_{id}}|} = \frac{p_x}{\frac{1}{c}\sqrt{E_{id}^2 - m^2c^4}} = v_x \frac{\gamma cm}{\sqrt{E_{id}^2 - m^2c^4}} \tag{9}$$

So, if we isolate v_x:

$$v_x = x' \frac{\sqrt{E_{id}^2 - m^2c^4}}{\gamma cm} = cx' \frac{\sqrt{E_{id}^2 - m^2c^4}}{E_{id}(1+\delta)} \tag{10}$$

Since we can do the same to y coordinate, we can obtain:

$$v_y = y' \frac{\sqrt{E_{id}^2 - m^2c^4}}{\gamma cm} = cy' \frac{\sqrt{E_{id}^2 - m^2c^4}}{E_{id}(1+\delta)} \tag{11}$$

As we know γ and γ_{id}, we can determine $|\vec{v}|$ and $|\vec{v_{id}}|$ using equation (4):

$$|\vec{v}| = c\sqrt{1 - \frac{1}{\gamma^2}}, \qquad |\vec{v_{id}}| = c\sqrt{1 - \frac{1}{\gamma_{id}^2}} \tag{12}$$

And that is the way we will find the last component of the velocity, v_s:

$$v_s = \sqrt{|\vec{v}|^2 - v_x^2 - v_y^2} \tag{13}$$

Finally, the coordinates at the end of the straight path will be:

$$x = v_x t + x_0 \tag{14}$$

$$x' = x'_0 \tag{15}$$

$$y = v_y t + y_0 \tag{16}$$

$$y' = y'_0 \tag{17}$$

$$\delta = \delta_0 \tag{18}$$

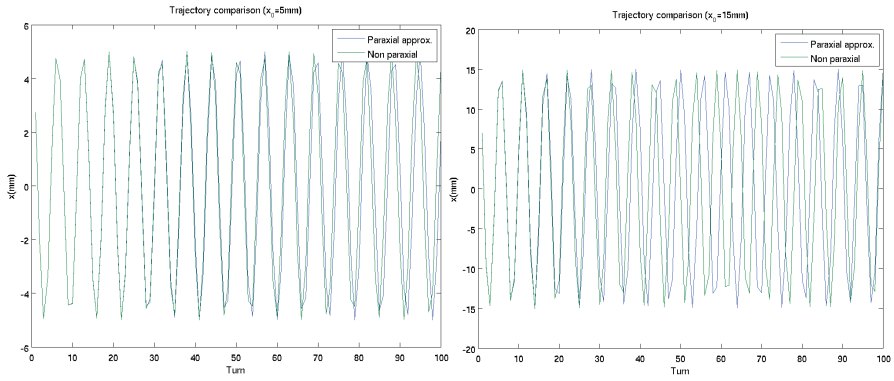
$$s = -cl(\frac{1}{|\vec{v_{id}}|} - \frac{1}{v_s}) + s_0 \tag{19}$$

where we have used equation (6) and other relations in order to obtain the behaviour of s.

3 Comparison with paraxial approximation

3.1 Trajectory

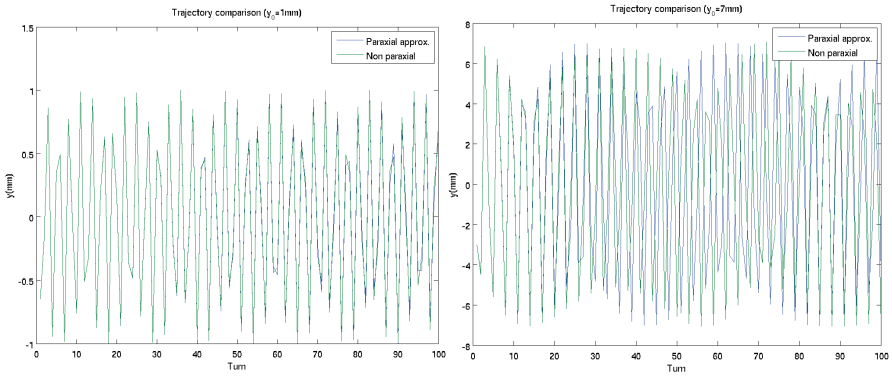
In order to know how this method behaves in comparison with the paraxial approximation, we can plot the two **trajectories** for different initial conditions for the ALBA lattice. If we start with the **x coordinate**:



(a) $x_0 = 5mm$ (b) $x_0 = 15mm$

Figure 1: Graphic representation of the trajectory of an electron in the x coordinate for two different initial displacements. We can see that in the first one ($x_0 = 5mm$) there is no significant difference between the paraxial approximation and the one without approximation. On the other hand, we can see that in the second case ($x_0 = 15mm$) the initial amplitude is big enough to have significant changes in the trajectory.

Now if we do the same for the **y coordinate**:



(a) $y_0 = 1mm$ (b) $y_0 = 7mm$

Figure 2: Graphic representation of the trajectory of an electron in the y coordinate for two different initial conditions. As before, we can see that in the first one ($y_0 = 1mm$) there is no significant difference but in the second one ($y_0 = 7mm$) there is.

Something interesting we can plot too is the **difference between trajectories** for both coordinates:

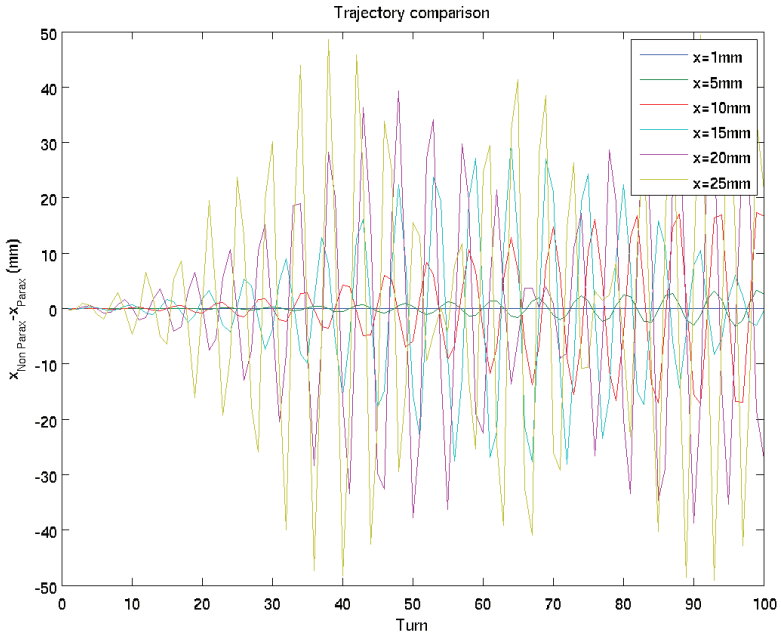


Figure 3: Difference between the trajectory of the paraxial approximation and the non-paraxial for x coordinate.

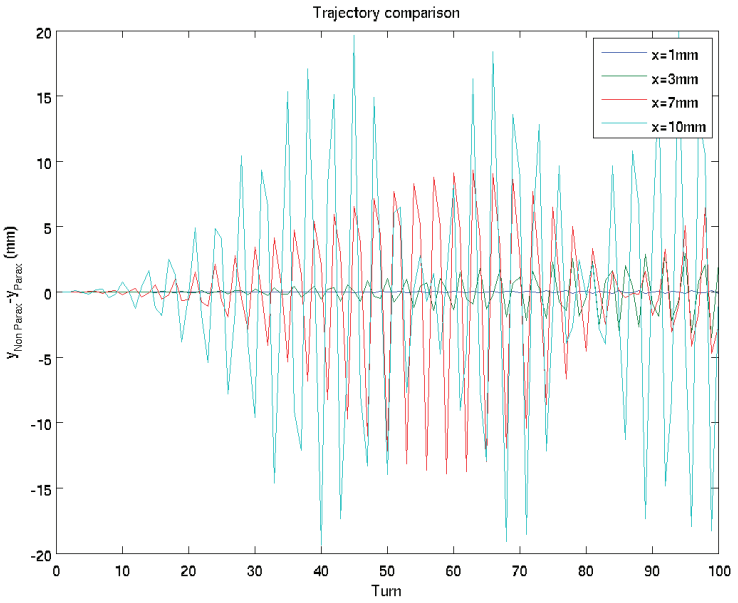


Figure 4: Difference between the trajectory of the paraxial approximation and the non-paraxial for y coordinate.

Finally, we can determine the **standard deviation** for different initial amplitudes and see how it behaves. What we can expect is that it increases with the initial amplitude and that is what we find:

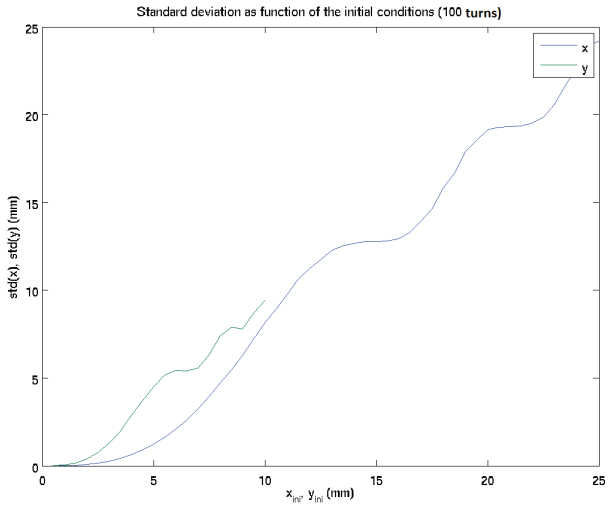
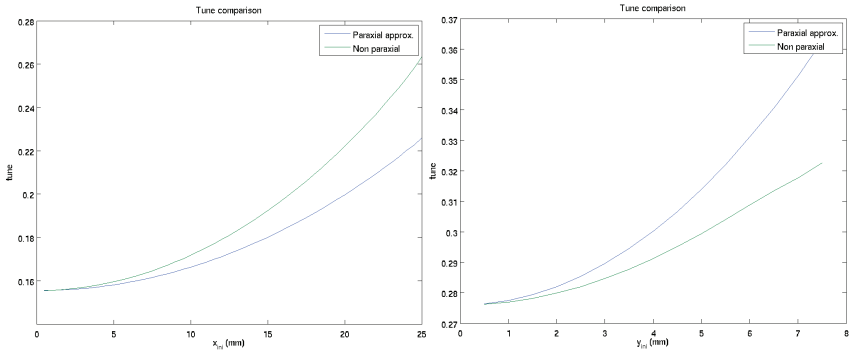


Figure 5: Comparison between the standard deviation of x coordinate and y coordinate as function of the initial deviation for 100 turns.

3.2 Tune

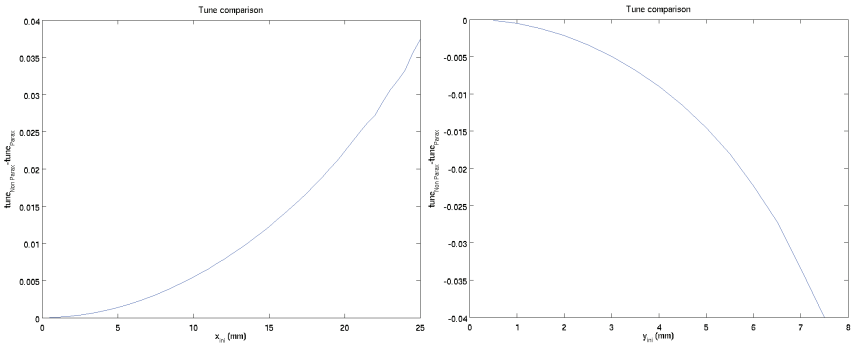
Probably something more important than the change of trajectory is the change of **tune** -or betatronic number- since it can affect the machine lifetime and injection efficiency. We can see its change if we plot -as we have done before- both tunes and the difference between them for 100 turns.

After the analysis of the data we can confirm that **tune and chromaticity are not affected** -the effect in chromaticity is of the order of 10^{-5}.



(a) x coordinate (b) y coordinate

Figure 6: Graphic representation of the tune of an electron for each coordinate and both methods -with and without approximation.



(a) x coordinate (b) y coordinate

Figure 7: Difference between tunes. For x coordinate the non-paraxial tune is smaller than the paraxial one. On the other hand, for y coordinate the non-paraxial is larger than the paraxial one. We can see that there is not a significant difference between both coordinates for the electrons that are near the centre of the tube ($\delta\nu_i \simeq 10^{-4}$) but for the ones that have big oscillations the tune change is significant ($\delta\nu_i \simeq 10^{-2}$).

3.3 Tracking computation time

One last comparison is needed to be done in order to have a quite complete comparison. Something we can ask to ourselves is which is the difference between these two methods but from the point of view of **time optimization**. We can expect this method to be slower than the other one since it needs more complex operations as square roots.

For the ALBA lattice we have run several turns and the difference between both methods is that non-paraxial approximation requires almost **70% more time than the paraxial approximation**. This is probably the most important difference to consider while choosing a method to carry out our long simulations.

4 Annex

4.1 Derivation of the paraxial approximation

In order to prove that the proposed method is correct we derive the paraxial formulas from the ones we obtained assuming the approximations. First of all, let's introduce the paraxial formulas:

$$x = x' \frac{l}{1+\delta} + x_0 \tag{20}$$

$$x' = x'_0 \tag{21}$$

$$y = y' \frac{l}{1+\delta} + y_0 \tag{22}$$

$$y' = y'_0 \tag{23}$$

$$\delta = \delta_0 \tag{24}$$

$$s = \frac{l}{2(1+\delta)^2}(x'^2 + y'^2) + s_0 \tag{25}$$

If we compare them with the equations we obtained we realize that we have to prove the following in the paraxial approximation:

$$v_x t \quad \rightarrow \quad l\frac{x'}{1+\delta} \tag{26}$$

$$v_y t \quad \rightarrow \quad l\frac{y'}{1+\delta} \tag{27}$$

$$-c(\frac{1}{|\vec{v_{id}}|} - \frac{1}{v_s}) \quad \rightarrow \quad \frac{x'^2 + y'^2}{2(1+\delta)^2} \tag{28}$$

Let's start with the first one. First of all we write:

$$v_x = cx' \frac{\sqrt{E_{id}^2 - m^2c^4}}{E_{id}(1+\delta)} \simeq cx' \frac{1}{1+\delta} \tag{29}$$

where we have considered $E_{id} >> mc^2$. Now we can say that $v_s \simeq c$ since we are considering the paraxial approximation, that is, the angle is very small so there is a big loss of velocity in the transversal components. Assuming that:

$$t = \frac{l}{v_s} \simeq \frac{l}{c} \tag{30}$$

Being that said we can arrive to:

$$v_x t \simeq l\frac{x'}{1+\delta} \tag{31}$$

And since we can do the same for y component, we have:

$$v_y t \simeq l\frac{y'}{1+\delta} \tag{32}$$

Finally we only have to prove one last limit. The very first we have to do is:

$$|\vec{v_{id}}| = c\sqrt{1 - \frac{1}{\gamma_{id}^2}} \simeq c(1 - \frac{1}{2}\frac{1}{\gamma_{id}^2}) \simeq c(1 - \frac{1}{2}\frac{m^2c^4}{E_{id}^2}) \tag{33}$$

$$v_s = \sqrt{|\vec{v}|^2 - v_x^2 - v_y^2} = |\vec{v}|\sqrt{1 - \frac{v_x^2 + v_y^2}{|\vec{v}|^2}} \simeq |\vec{v}|(1 - \frac{1}{2}\frac{v_x^2 + v_y^2}{|\vec{v}|^2}) \tag{34}$$

where in both cases we have used the approximation:

$$\sqrt{1 - x} \simeq 1 - \frac{1}{2}x + O(x^2), \quad if\ x \to 0 \tag{35}$$

Now we can do:

$$\frac{1}{|\vec{v_{id}}|} \simeq \frac{1}{c}(1 + \frac{1}{2}\frac{m^2c^4}{E_{id}^2}) \tag{36}$$

$$\frac{1}{v_s} \simeq \frac{1}{|\vec{v}|}(1 + \frac{1}{2}\frac{v_x^2 + v_y^2}{|\vec{v}|^2}) \tag{37}$$

where we have used the approximation:

$$\frac{1}{1 - x} \simeq 1 + x + O(x^2), \quad if\ x \to 0 \tag{38}$$

If we put all together:

$$c(\frac{1}{|\vec{v_{id}}|} - \frac{1}{v_s}) \simeq \frac{1}{2}(\frac{m^2c^4}{E_{id}^2} - \frac{v_x^2 + v_y^2}{|\vec{v}|^2}) \tag{39}$$

And since $E_{id} >> mc^2$, we can ignore the first term:

$$c(\frac{1}{|\vec{v_{id}}|} - \frac{1}{v_s}) \simeq -\frac{1}{2}\frac{v_x^2 + v_y^2}{|\vec{v}|^2} \tag{40}$$

Finally, since $v_i \sim x_i'$ (for i=1,2) we can see that the dependence on this coordinate is of the same order that in the paraxial approximation. Probably the third and higher orders disagree with paraxial approximation but this is enough in order to assert that both methods have the same behaviour in the paraxial approximation.

References

[1] Laurent Nadolsky. Application de l'analyse en fréquence à l'etude de la dynamique des sources de lumière. 2001.

5 Annex 2

5.1 DriftNoParaxPass.c: the code

This is the **C-code** used in order to replace *DriftPass* and then establish the non-paraxial method:

```
#include <math.h>
#include "mex.h"
#include "elempass.h"
#include <stdio.h>


void DriftPass(double *r_in, double le, int num_particles, double Eid)
/* le - physical length
* r_in - 6-by-N matrix of initial conditions reshaped into
* 1-d array of 6*N elements
*/

{   int i, i6;
    double mc2=0.511e6, c=299792458;
    double gamma, E, t, vx, vy, vs, coef, vmod, vmod_id, gamma_id;

    for(i = 0;i<num_particles;i++)
    {   i6 = i*6;
        if(!mxIsNaN(r_in[i6]))
        {
            E=Eid*(1+r_in[i6+4]);
            gamma=E/mc2;
            gamma_id=Eid/mc2;
            vmod=c*sqrt(1-1/(gamma*gamma));
            vmod_id=c*sqrt(1-1/(gamma_id*gamma_id));
            coef=(c*sqrt(Eid*Eid-mc2*mc2))/(mc2*gamma);
            vx=r_in[i6+1]*coef;
            vy=r_in[i6+3]*coef;
            vs=sqrt(vmod*vmod-vx*vx-vy*vy);
            t=le/vs;
            r_in[i6+0]+= t*vx;
            r_in[i6+2]+= t*vy;
            r_in[i6+5]-= c*(le-t*vmod_id)/vmod_id;
        }

    }
}
```