Beam Physics Note
TRI-BN-20-03
February 28, 2020

Addition of global time tracking to `TRANSOPTR`

Olivier Shelbaya

TRIUMF

Abstract: This note details the addition of global time variable tracking in the envelope code `TRANSOPTR`. A brief example is presented, showing verification of time of flight phase changes in the code, using a simulation of the 11MHz HEBT buncher.

Introduction

In this short note I relate the addition of global time tracking in `TRANSOPTR` [1], which I added and verified as part of ongoing work toward the implementation of the ISAC-I linac in the code. Under its previous configuration, reported in [2], time was simply not counted for elements other than `linac` and `rfqlinac`, found in [3] and [4]. As `TRANSOPTR` did not previously track time through intermediate elements between successive RF cavities, the global relationship between RF phases within a single `TRANSOPTR` simulation did not take into account intertank time of flight phase delays.

Within `TRANSOPTR`, time is treated as a dependent coordinate (s is independent), associated with the reference particle's time. We further note that `TRANSOPTR` actually keeps track of ct, in units of length, which is stored in an array: `sx(13,5)`. Ensuring a global time tracking in the code means ensuring the elapsed ct in each element, stored in `dsx(13,5)` at each integration step, is kept and added to the global array `sx` for output.

Tracking time in `SC.f`

The subroutine `SC.f` was modified with the addition of a time-tracking block which follows closely from `SCLINAC.f`:

```
c ******************************************************************
c Time Iteration
c ******************************************************************
gamm1=sx(13,6)
gamma=1.+gamm1
ETA=SQRT(gamm1*(gamm1+2.))
BETA=eta/gamma
```

Previously, elements of the infinitesimal transfer matrix `dsx(i,j)` were set to zero at the end of the subroutine, this is maintained but `dsx(13,5)` is set immediately thereafter:

```
do 56 j=1,6
56 dsx(13,j)=0.
dsx(13,5)=1./beta !  dctime/ds=1./beta
```

Calls to `INTSC`, which numerically solves the sigma matrix, pass subroutine `SC` to the Runge-Kutta integrator, most commonly using the `RKC` [5] routine. As `SC` now keeps track of time, any `TRANSOPTR` element which calls `INTSC` now has global time tracked throughout the simulation. This includes common elements such as quadrupoles and drifts.

Outuputting of ct in `fort.envelope`

I've also added ct output to `fort.envelope`, the newest output format developed for `TRANSOPTR` use with the HLA-envelope application. This first entailed modification of `ENVLP_OUT.f` to adapt the file header:

```
!:  First header line with data label:
WRITE(1,'(A5,13A17,A12,15(A4,2I1,A11),36(A4,2I1,A11))')
+"s","x","x' ","y","y' ","z","z' ","fx","fy","E","ct",
+"x","y","z","",
+((" r",j-1,k,"",j=2,k),k=1,6),
+((" m",l ,m,"",m=1,6),l=1,6)


!:  Second header line with units
WRITE(1,'(A8,64A17)')
+"[cm]",("["//uniti(i)//"]",i=1,6),
+"[rad/cm**2]","[rad/cm**2]","[MeV]","[cm]",
+"[cm*rad]","[cm*rad]","[cm*rad]",
+(("[]",j=2,k),k=1,6),
+(("["//uniti(l)//"/"//uniti(m)//"]",m=1,6),l=1,6)
```

The subroutine `ENVLP_OUT` in the same file was also modified to accommodate the new ct entry, placed in column 11, adjacent to energy:

```
OUTENVLP(1,nelm)=VELM(7,nelm)/conx(8)
DO i=1,6
OUTENVLP(i+1,nelm)=velm(i,nelm) !the 6 sizes in internal units
ENDDO
OUTENVLP(8,nelm)=velm(12,nelm) !focal strength in x
OUTENVLP(9,nelm)=velm(13,nelm) !focal strength in y
OUTENVLP(10,nelm)=velm(14,nelm)!beam energy
OUTENVLP(11,nelm)=SX(13,5) !ctime
OUTENVLP(12,nelm)=emit(1)*pratio
OUTENVLP(13,nelm)=emit(2)*pratio
OUTENVLP(14,nelm)=emit(3)*pratio


ivlp=15
```

Finally, a small modification was made in `main.f` for case `iprint` < 1, which calls the new `fort.envelope` output:

```
elseif(iprint.eq.-1) then
!:  special output for the envelope application
```

```
write(1,*)(OUTENVLP(ii,J),ii=1,65)
```

The upper loop bound for `ii` was increased from 64 to 65 to accommodate ct output.

Time of Flight Phase Effect Demonstration

To test and demonstrate the implementation, I've run a small example segment, starting at HEBT:Q10 and terminating at the exit of the HEBT-11MHz buncher. See [6] for more details on that section. Using a ^{22}Ne^{4+} beam at 300 keV/u, I obtain a time of flight ct = 709.4cm for the 18.0cm long `mquad` representing Q10, which implies:

$$\beta = \frac{d}{ct} \tag{1}$$

$$\beta = 2.537 \times 10^{-2} \tag{2}$$

Using the output ct value from `TRANSOPTR` to produce β and computing γ, we can verify the resulting kinetic energy against the known simulation starting value:

$$K = (\gamma - 1)mc^2 \tag{3}$$

$$K = 3.22064 \times 10^{-4} \times 20484.9 \quad \text{[MeV]} \tag{4}$$

$$K = 6.59745 \quad \text{[MeV]} \tag{5}$$

The input kinetic energy in `data.dat` is of 6.59742 MeV, a difference on the order of 1 part in 2×10^5, which means erros are present and accumulating in ct. While this shows that the time of flight through a quadrupole produces a sensible number, what about RF phasing, the intended goal of this implementation?

For this, I've found a near-optimal accelerating phase for the 11MHz HEBT buncher, corresponding to ϕ_1 = 115.0°, V_s = 0.6 MV [1], which produces an output energy of 6.756 MeV for the same beam, using the HEBT:Q10 to buncher drift which consists of the 18 cm quad and a 55.196 cm drift in `TRANSOPTR`.

When the same beam travels through Q10, but a doubled quad to buncher distance, the expected phase delay that I compute is of $\Delta\phi$ = 307.5° due to the additional time-of-flight up to the buncher,

---

[1] Here V_s is the on-axis electric field scaling voltage factor as used in `TRANSOPTR`, and not the effective voltage as per the usual TTF-jargon.

in 11.78 MHz RF degrees. First, when I run TRANSOPTR with the doubled drift but same ϕ = 115.0°, I no-longer get maximum energy output, as expected. Second, when I change the phase to ϕ_2 = -192.5°, the expected TOF phase shift, again using the doubled drift distance, I restore maximum energy output in the buncher.

Future Possibilities for Long Simulations

The accumulation of numerical errors is an issue to consider, especially in the context of long sequences of elements. With global time tracking, the variable ct accumulates $ds/\beta(s)$ at each integration step, which introduces the risk of numerical errors due to mixing numbers spanning different orders of magnitude. Though the present implementation does not attempt to tackle this, two main ideas have been discussed by the beam pyhsics group. The first involves invoking the FORTRAN function `MOD(A,B)`, which computes the remainder of the division of `A` by `B`. Under this idea, upon entering an RF cavity, either `linac` or `rfqlinac` in TRANSOPTR, a local time variable could be declared which would be initialized with a phase shift corresponding to the remainder of the division of ct by the period of the RF frequency. The remainder is then interpreted as a TOF phase shift. It has also been recommended to implement tracking of $s/\beta(s)$ and compare it to ct as a means to keep track of numerical errors. As this is of importance for phasing of sequential RF cavities, as called for by the ongoing implementation of the ISAC DTL in TRANSOPTR, it was decided to start with the simple implementation and use upcoming tests of the TRANSOPTR-DTL to produce a more in-depth analysis at a later time.

Conclusion

This small addition now enables the global tracking of time in TRANSOPTR, which is an essential feature for upcoming planned use with the ISAC accelerators. The output file `fort.envelope` now features ct output when `iprint` is set to -1 in `data.dat`. A brief example, in which basic time of flight effects are verified in TRANSOPTR, was presented.

Acknowledgements

The author wishes to extend thanks to Thomas Planche, who suggested use of the modulo operation as a means to initiate local time in the linac cavity. Rick Baartman is also thanked for many useful discussions and his suggestion for $s/\beta(s)$ tracking as well. Thanks to the members of the TRIUMF Beam Physics group for their assistance during discussions on this ongoing implementation.

References

[1] Heighway EA and Hutcheon RM. *Transoptr - A second order beam transport design code with optimization and constraints*. Technical report, Atomic Energy of Canada Limited, 1981.

[2] Baartman R. *TRANSOPTR Changes Since 1984*. Technical Report TRI-BN-16-06, TRIUMF, 06 2016.

[3] R. Baartman. Linac Envelope Optics. Technical Report TRI-BN-15-03, TRIUMF, 2015.

[4] O Shelbaya, R Baartman, and O Kester. Fast radio frequency quadrupole envelope computation for model based beam tuning. *Physical Review Accelerators and Beams*, 22(11):114602, 2019.

[5] Jan Christiansen. Numerical solution of ordinary simultaneous differential equations of the 1st order using a method for automatic step change. *Numerische Mathematik*, 14(4):317–324, 1970.

[6] Shelbaya O. *TRANSOPTR Implementation of the HEBT Beamlines*. Technical Report TRI-BN-19-06, TRIUMF, 2019.