TRIUMF Beam Physics Note
TRI-BN-23-11
April 25, 2023

Implementing linear coupling analysis in CYCLOPS

Lige Zhang

TRIUMF

Abstract: We have implemented linearized motion equations in CYCLOPS, taking into account median plane asymmetry fields, to calculate the 4x4 coupled transverse transfer matrix. The equations were derived from a Hamiltonian that preserves the symplecticity of the transfer matrix. We have also developed a post-processing script to handle the coupled matrix. Using the symplectic rotation of the frame, the coupled matrix is converted into normal mode, where there is no coupling, enabling the use of standard twiss parameters for 2D motion analysis of beam dynamics in the cyclotron. We have tested the output results of CYCLOPS against the analytical smooth approximation formula under static tune cases.

4004 Wesbrook Mall, Vancouver, B.C. Canada V6T 2A3 · Tel: 604 222-1047 · Fax: 604 222-1074 · www.triumf.ca

1 Motion equations

The usual expression for the cyclotron Hamiltonian is given in polar coordinates[1, 2], but this only includes the median plane symmetric field, which makes the motion in the transverse directions independent of each other. In this paper, we derive the coupled Hamiltonian by starting with the vector potential without assuming median plane symmetry. The magnetic vector potential A has a degree of freedom from gauge choosing. By adding a gradient of a proper scalar function to the vector potential, we can make the vector potential in the axial direction zero. The magnetic field is then given by $\boldsymbol{\nabla} \times \mathrm{A}$.

$$\begin{aligned} B_r &= -\frac{\partial A_\theta}{\partial z}, \\ B_\theta &= \frac{\partial A_r}{\partial z}, \\ B_z &= \frac{1}{r}\frac{\partial}{\partial r}\left(rA_\theta\right) - \frac{1}{r}\frac{\partial A_r}{\partial \theta}. \end{aligned} \tag{1}$$

Using Gordon's field expansion in powers of z, the vector potential can also be expressed in terms of two independent median plane fields B and C, as shown below:

$$\begin{aligned} rA_r &= \frac{\partial \nabla_2^{-2} B}{\partial \theta} - z\frac{\partial C}{\partial \theta} - \frac{z^2}{2!}\frac{\partial B}{\partial \theta} + \frac{z^3}{3!}\frac{\partial \nabla_2^2 C}{\partial \theta} + \frac{z^4}{4!}\frac{\partial \nabla_2^2 B}{\partial \theta} - \ldots, \\ A_\theta &= -\frac{\partial \nabla_2^{-2} B}{\partial r} + z\frac{\partial C}{\partial r} + \frac{z^2}{2!}\frac{\partial B}{\partial r} - \frac{z^3}{3!}\frac{\partial \nabla_2^2 C}{\partial r} - \frac{z^4}{4!}\frac{\partial \nabla_2^2 B}{\partial r} - \ldots \end{aligned} \tag{2}$$

where $B = B(r, \theta)$ produces a field with median plane symmetry, while $C = C(r, \theta)$ spoils the symmetry. The inverse Laplacian is denoted by ∇_2^{-2}. Since no boundary conditions need to be set for the poison equation defined by the inverse Laplacian, the particular solution that integrates the Green function can be used to calculate the map of ∇_2^{-2} using map B. To avoid the calculation of the inverse Laplacian, a new gauge can be selected that removes the 0th-order terms of z in A_r, which is

$$\begin{aligned} rA_r &= -z\frac{\partial C}{\partial \theta} - \frac{z^2}{2!}\frac{\partial B}{\partial \theta} + \frac{z^3}{3!}\frac{\partial \nabla_2^2 C}{\partial \theta} + \frac{z^4}{4!}\frac{\partial \nabla_2^2 B}{\partial \theta} - \ldots, \\ A_\theta &= -\frac{1}{r}\int rB dr + z\frac{\partial C}{\partial r} + \frac{z^2}{2!}\frac{\partial B}{\partial r} - \frac{z^3}{3!}\frac{\partial \nabla_2^2 C}{\partial r} - \frac{z^4}{4!}\frac{\partial \nabla_2^2 B}{\partial r} - \ldots \end{aligned} \tag{3}$$

The magnetic field given by this vector potential is

$$
\begin{aligned}
B_z &= -B + z\nabla_2^2 C + \frac{z^2}{2!}\nabla_2^2 B - \frac{z^3}{3!}\nabla_2^4 C - \frac{z^4}{4!}\nabla_2^4 B + \ldots, \\
B_r &= -\frac{\partial C}{\partial r} - z\frac{\partial B}{\partial r} + \frac{z^2}{2!}\frac{\partial \nabla_2^2 C}{\partial r} + \frac{z^3}{3!}\frac{\partial \nabla_2^2 B}{\partial r} - \ldots, \\
rB_\theta &= -\frac{\partial C}{\partial \theta} - z\frac{\partial B}{\partial \theta} + \frac{z^2}{2!}\frac{\partial \nabla_2^2 C}{\partial \theta} + \frac{z^3}{3!}\frac{\partial \nabla_2^2 B}{\partial \theta} - \ldots
\end{aligned} \tag{4}
$$

The Hamiltonian in cylindrical coordinates is written as

$$
H = \sqrt{(P_r - qA_r)^2 c^2 + P_z^2 c^2 + c^4 m_0^2 + \frac{c^2 \left(P_\theta - qr\,\mathrm{A}_\theta\left(r, z\right)\right)^2}{r^2}} \tag{5}
$$

Where the canonical momenta are

$$
\begin{aligned}
P_r &= p_r + qA_r \\
P_\theta &= \gamma m_0 \theta' r^2 + qrA_\theta \\
P_z &= p_z
\end{aligned} \tag{6}
$$

Choosing θ as the independent variable, the Hamiltonian H is $-P_\theta$,

$$
\mathrm{H} = -P_\theta = -r\sqrt{P_0^2 - \left(P_r - qA_r\right)^2 - P_z^2} - qrA_\theta \tag{7}
$$

To study the motion around the closed orbit, we introduce the new coordinates and momenta

$$
\begin{aligned}
x &= r - r_0, \\
P_x &= P_r - P_{r0}, \\
y &= y - y_0, \\
P_y &= P_z - P_{z0}.
\end{aligned} \tag{8}
$$

We construct a type 2 generating function as below

$$
G = rP_x - r_0 P_x + P_{r0} r + zP_y - z_0 P_y + P_{z0} z \tag{9}
$$

The new Hamiltonian using the new variables could be written as

$$
K = H + \frac{\partial G}{\partial \theta}. \tag{10}
$$

We expand the Hamiltonian around the close orbit and throw out the constant, the new Hamiltonian with up to quadratic degree terms could be used to linearize motion equations. Due to the length of the equations, we provide the Python script in Appendix 1 to show the derivation process. We use Sympy, a symbolic mathematics package in Python, to derive the linearized equations and generate the Fortran code expressions.

With perfect median plane symmetry, the closed orbit is on the median plane. Substituting $z, p_z, C = 0$ in the linearized equation gives

$$\begin{aligned}
\frac{dx}{d\theta} =& \frac{p_r}{p_\theta} x + \frac{r p^2}{{p_\theta}^3} p_x, \\
\frac{dp_x}{d\theta} =& -\frac{p_r}{p_\theta} p_x - q \left(B + r \frac{\partial B}{\partial r} \right) x, \\
\frac{dz}{d\theta} =& \frac{r}{p_\theta} p_z, \\
\frac{dp_z}{d\theta} =& q \left(r \frac{\partial B}{\partial r} - \frac{p_r}{p_\theta} \frac{\partial B}{\partial \theta} \right) z,
\end{aligned} \tag{11}$$

This is what CYCLOPS has used to calculate the transfer matrix[3], consequently the twiss parameters and tunes for different energies.

2 Implement the linearized equations in CYCLOPS

In the current version of CYCLOPS,[4, 5, 6] the asymmetric median plane field is only used to calculate the static equilibrium orbit for each set of energies. However, the transfer matrix, tunes and twiss parameters are all calculated using the linearized equation without considering the asymmetric field. Thus, the transversal motion is uncoupled and can be described by two 2×2 transfer matrices. The 8 terms of the 2 matrices are calculated by integrating the equations of motion, eq. 11, twice with the two different initial conditions given below, using the linearized equation:

$$V_1 = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}, V_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}, \tag{12}$$

where V is the coordinates vector $[x, p_x, z, p_z]^T$. The uncoupled transfer matrix is also used as an input for COMA [7], a linear simulation code for cyclotron beam dynamic study.

To calculate the transversal 4×4 coupled transfer matrix at each integration step, we incorporated four new linearized equations, derived in Appendix 1, that take into account the asymmetric field. After finding the closed orbit, the coordinates and field along the SEO are used as coefficients in the equations. The linear equations are then integrated four times with four different initial conditions in the Runge-Kutta integrator to calculate all 16 terms of the transfer matrix. The four initial conditions are given below.

$$V_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, V_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, V_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, V_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}. \quad (13)$$

The 1-turn matrix is computed and then stored in a new output file named "fort.40".

3 Coupling matrix

For the coupling resonance study, it is important to preserve the invariant in the simulation by ensuring good symplecticity of the transfer matrix. Figure 1 shows the symplecticity of the 4×4 transfer matrix over the full energy range of TRIUMF cyclotron.

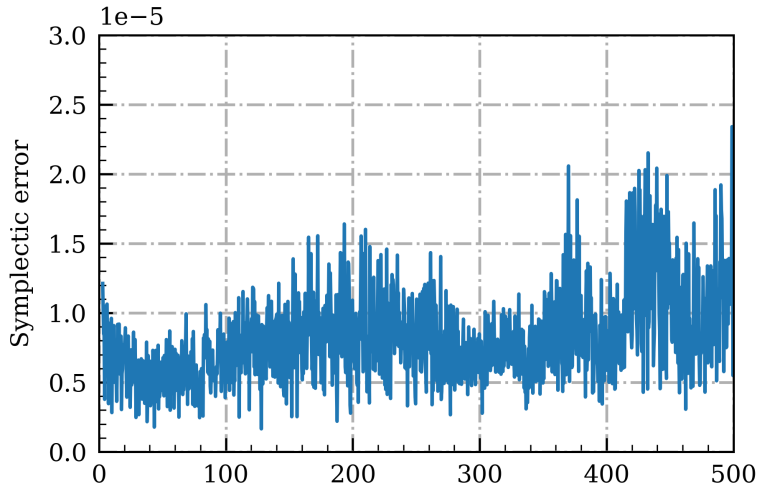

Figure 1: Symplectic error of the transfer matrix computed by CYCLOPS. The symplectic error here is the RMS value of all the terms of the symplectic checking matrix. The symplectic error is less than 2.5×10^{-5} over all the energy ranges, which is small enough for the coupling resonance study in TRIUMF cyclotron as the beam travels only around one thousand turns before extraction.

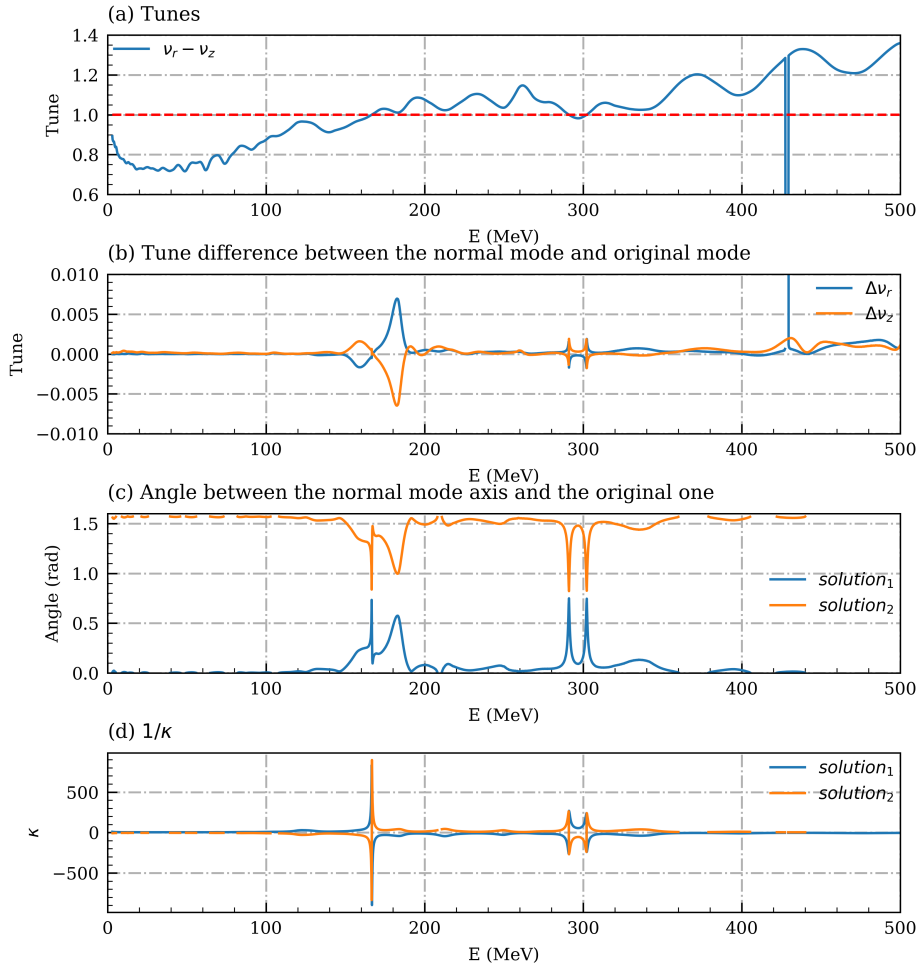

Figure 2: Parameteres extracted from the coupling matrix. (a) $\nu_r - \nu_z$ shows the crossing of the linear coupling resonance, first at around 166 MeV, then twice at around 300 MeV. (b) The tune difference between the original mode and the normal mode is within 0.002, indicating that the tune diagram calculated by the original CYCLOPS without considering the coupling is sufficiently accurate for cyclotron design and optimization. (c) The rotation angle between the normal mode frame and the laboratory frame is calculated using $\arccos(\gamma)$, where γ varies approximately between 0 and 1, indicating that the normal mode axis is rotating from 0 ° to 90 ° during the crossing. When it is on resonance, the angle is 45 °, as shown by the three discontinuous points. (d) κ is calculated by the difference between the normal mode vertical tune and the radial tune, and $1/\kappa$ indicates the coupling strength. This parameter is the same as the one in the smooth approximation[8, 9, 10].

The coupled 1-turn matrix could be decomposed into normal mode using[11, 12, 13]

$$\mathbf{T} = \begin{bmatrix} \mathbf{M} & \mathbf{n} \\ \mathbf{m} & \mathbf{N} \end{bmatrix} = \mathbf{V}\mathbf{U}\mathbf{V}^{-1} \tag{14}$$

where $\mathbf{U} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} \end{bmatrix}, \mathbf{V} = \begin{bmatrix} \gamma\mathbf{I} & \mathbf{C} \\ -\mathbf{C}^{\dagger} & \gamma\mathbf{I} \end{bmatrix}$. The superscript '†'donates the symplectic conjugate. The diagonalized 4×4 matrix $\mathbf{U}$ describes the uncoupled motion in the normal mode. The coordinates in the normal model $\mathbf{w}$ are related to the laboratory coordinates $\mathbf{x}$ by $\mathbf{x} = \mathbf{V}\mathbf{w}$. The solutions of C and γ could be found in Sagan's paper[14], given as

$$\begin{aligned} \gamma &= \sqrt{\frac{1}{2} + \frac{1}{2}\sqrt{\frac{(\mathrm{Tr}[\mathbf{M}-\mathbf{N}])^2}{(\mathrm{Tr}[\mathbf{M}-\mathbf{N}])^2 + 4\det\mathbf{H}}}}, \\ \mathbf{C} &= \frac{-\mathbf{H}\,\mathrm{sgn}(\mathrm{Tr}[\mathbf{M}-\mathbf{N}])}{\gamma\sqrt{(\mathrm{Tr}[\mathbf{M}-\mathbf{N}])^2 + 4\det\mathbf{H}}}. \end{aligned} \tag{15}$$

where $\mathbf{H} = \mathbf{m} + \mathbf{n}^{\dagger}$.

As long as $\det\mathbf{H} > 0$, a second solution of C and γ exists, which is given as

$$\begin{aligned} \gamma &= \sqrt{\frac{1}{2} - \frac{1}{2}\sqrt{\frac{(\mathrm{Tr}[\mathbf{M}-\mathbf{N}])^2}{(\mathrm{Tr}[\mathbf{M}-\mathbf{N}])^2 + 4\det\mathbf{H}}}}, \\ \mathbf{C} &= \frac{\mathbf{H}\,\mathrm{sgn}(\mathrm{Tr}[\mathbf{M}-\mathbf{N}])}{\gamma\sqrt{(\mathrm{Tr}[\mathbf{M}-\mathbf{N}])^2 + 4\det\mathbf{H}}} \end{aligned} \tag{16}$$

4 Static tune cases

To study the static tune case, we tracked one particle for 1000 turns at fixed energies without acceleration using the 1-turn transfer matrix generated by CYCLOPES.

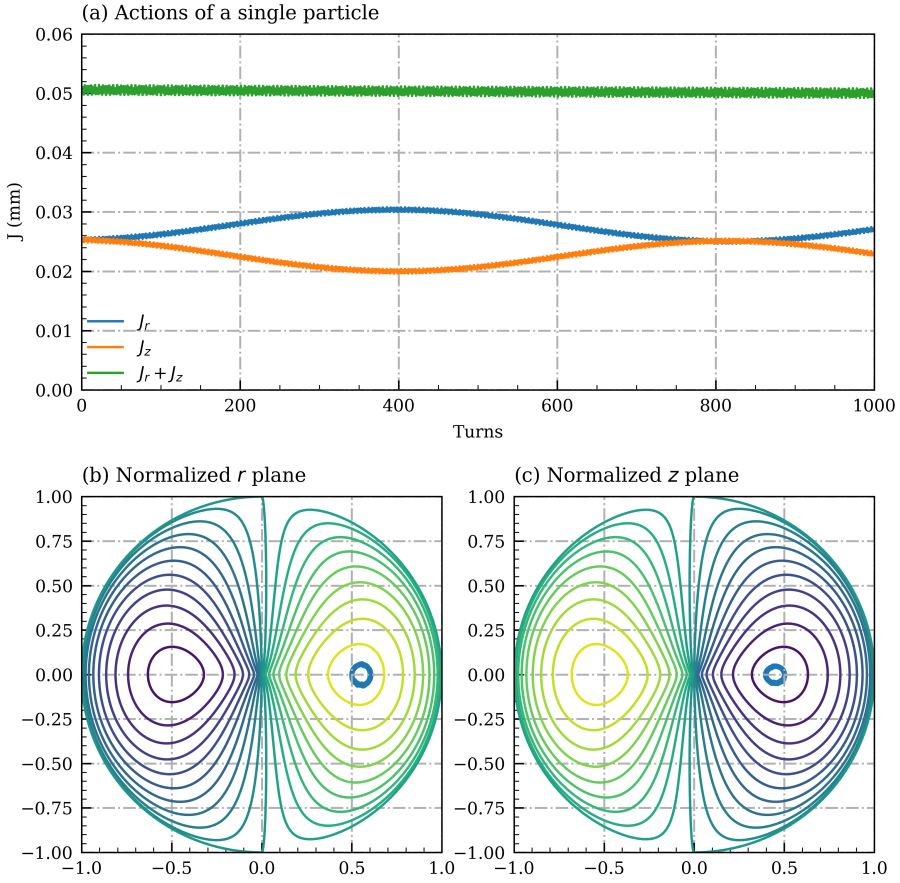

Figure 3: Single particle tracking results using the transfer matrix at 166.59 MeV, which is very close to the on-resonance point. The initial condition is $J_x = J_z$, which is the theoretical fixed point. The small amplitude exchange, not predicted by the analytical formula, is caused by the $\Delta\nu$ error of 6×10^{-5}. The tracking results show good agreement with the expected behavior based on the theoretical analysis, which demonstrates the validity of the model and the accuracy of the transfer matrix.

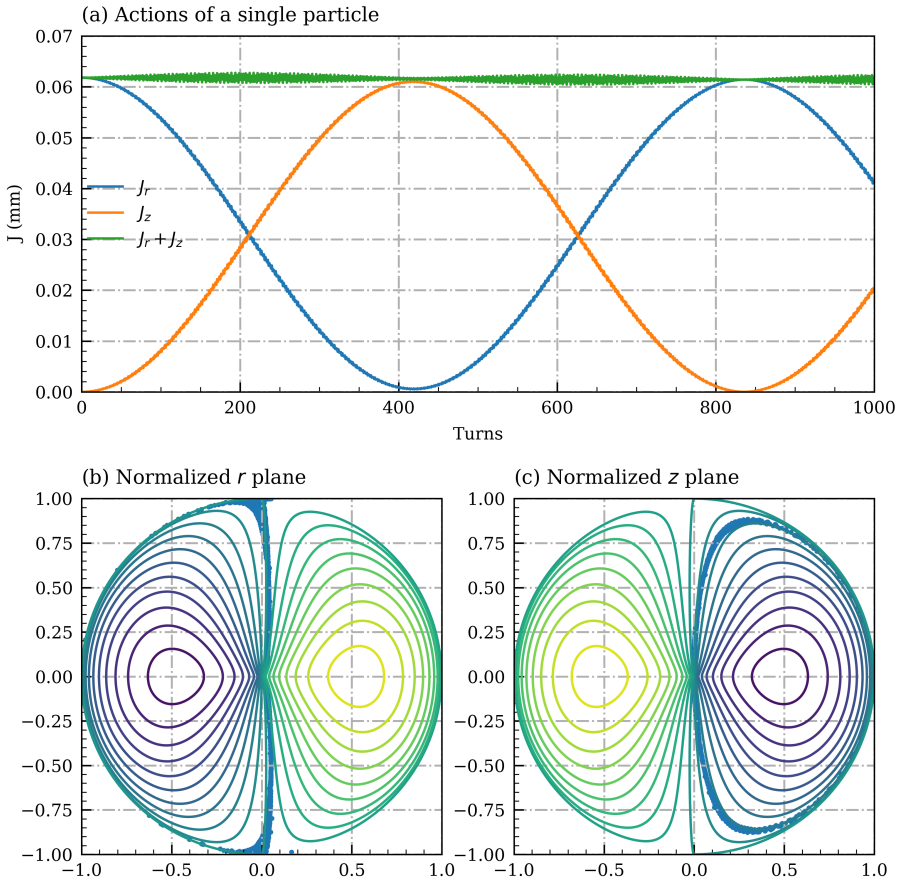

Figure 4: Single particle tracking results using the transfer matrix at 166.59 MeV, which is very close to the on-resonance point. The initial condition is with oscillations in only one direction. The results show a full oscillation amplitude exchange.

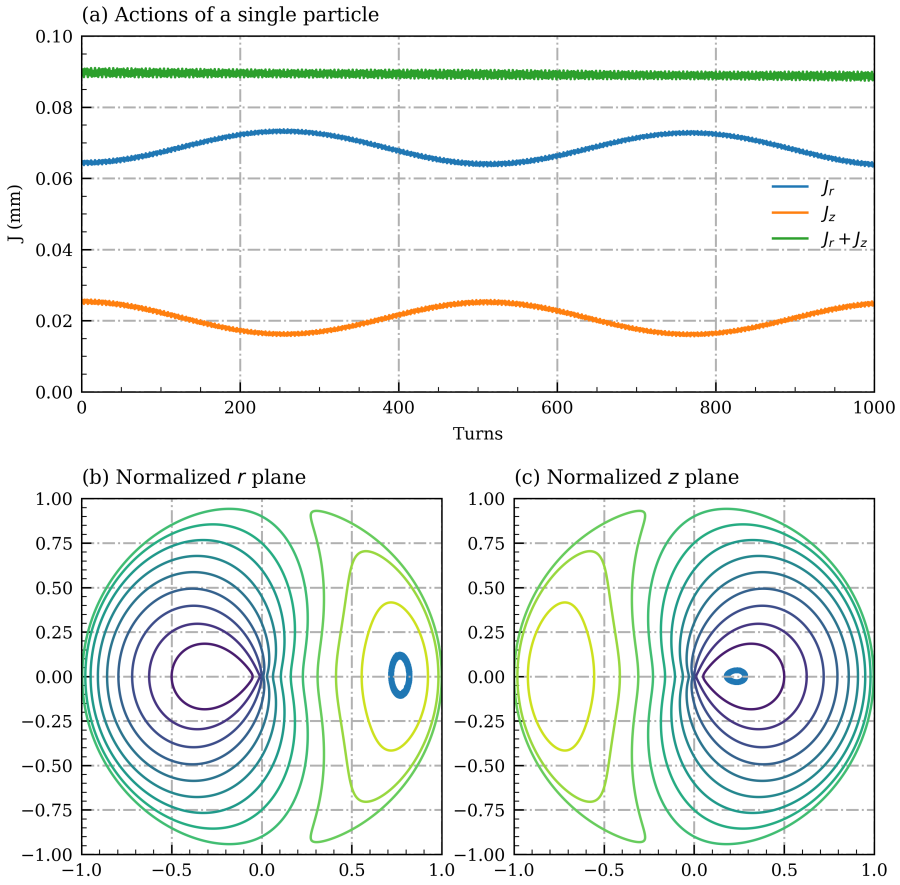

Figure 5: Single particle tracking results using the transfer matrix at 166.4 MeV, where $\Delta\nu = 0.001$ and $\kappa = 0.0016$. The initial condition is around the fixed point.

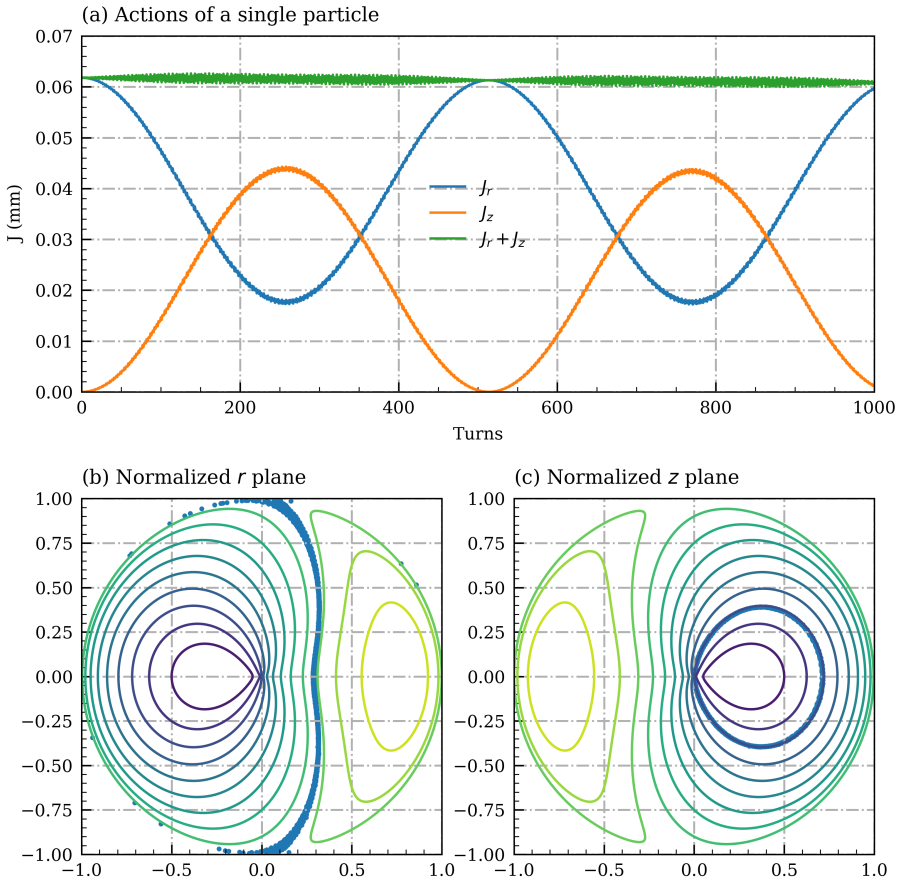

Figure 6: Single particle tracking results using the transfer matrix at 166.4 MeV, where $\Delta\nu = 0.001$ and $\kappa = 0.0016$. The initial condition is with oscillations in only one direction. The results show a maximum oscillation amplitude exchange.

5 Conclusions

We extended CYCLOPS to include the linearized coupling motion equation and compute the 4*4 coupled transversal transfer matrix, enabling the study of coupling resonances in cyclotrons. Single particle tracking was used to validate the output matrix against the smooth approximation formula. In all cases, the invariant $J_x + J_z$ was preserved in the simulation. The trace in the action-angle space was found to agree with the smooth approximation, demonstrating the accuracy of the new implementation.

References

[1] H. Hagedoorn, N. Verster, Orbits in an avf cyclotron, Nuclear Instruments and Methods 18 (1962) 201–228. doi:10.1016/S0029-554X(62)80032-9.

[2] W. Kleeven, On the energy limit of compact isochronous cyclotrons, Journal of Instrumentation 18 (03) (2023) P03025. doi:10.1088/1748-0221/18/03/P03025.
URL `https://dx.doi.org/10.1088/1748-0221/18/03/P03025`

[3] M. Gordon, Canonical treatment of accelerated orbits in sector-focused cyclotrons, Particle Accelerators 14 (1-2) (1983) 119–137.

[4] M. Gordon, Computation of closed orbits and basic focusing properties for sector-focused cyclotrons and the design of `CYCLOPS`, Particle Accelerators 16 (1984) 39–62.

[5] G. Mackenzie, Some Changes Made in the Field Analysis Programs `POLICY` and `CYCLOPS` – Part 1: Interpolation and Smoothing, Tech. Rep. TRI-DN-70-45, TRIUMF (1970).

[6] Y. Rao, R. Baartman, 50 years of cyclotrons designed using cyclops, Proceedings of FFAG 13 (2013).

[7] C. J. Kost, G. H. Mackenzie, Coma: a linear motion code for cyclotrons.
URL `https://www.osti.gov/biblio/4182460`

[8] W. Joho, Extraction of a 590 mev proton beam from the sin ring-cyclotron, Ph.D. thesis, Swiss Federal Institute of Technology, Zurich (1970).

[9] G. Guignard, The general theory of all sum and difference resonances in a three-dimensional magnetic field in a synchrotron, CERN Yellow Reports: Monographs, CERN, Geneva, 1976. doi:10.5170/CERN-1976-006.
URL `https://cds.cern.ch/record/185921`

[10] R. Baartman, Theory of $\nu_x - \nu_z = 1$ resonance in Cyclotron , Tech. Rep. TRI-BN-23-06, TRIUMF (2023).

[11] D. A. Edwards, L. C. Teng, Parametrization of linear coupled motion in periodic systems, IEEE Transactions on Nuclear Science 20 (3) (1973) 885–888. doi:10.1109/TNS.1973.4327279.

[12] M. Billing, Cornell report no (1985).

[13] L. C. Teng, Nal report no. fn-229 (1971).

[14] D. Sagan, D. Rubin, Linear analysis of coupled lattices, Physical Review Special Topics-Accelerators and Beams 2 (7) (1999) 074001.

6 Appendix

6.1 Python script to generate the equations for CYCLOPS

```
import sympy as sym
n = sym.symbols('n')
q = sym.symbols('q')
p,pr,pt,pz = sym.symbols('p,pr,pt,pz')
r,t,z,x,px,y,py = sym.symbols('r,t,z,x,p_x,y,p_y')
Br=-sym.Function('Br')(r,t)
Bt=-sym.Function('Bt')(r,t)
Bz=-sym.Function('Bz')(r,t)
Br0=sym.Function('B_r')(t)
Bt0=sym.Function(r'B_{\theta}')(t)
Bz0=sym.Function('B_z')(t)
DBRDR0=sym.Function('DBRDR')(t)
DBTDR0=sym.Function('DBTDR')(t)
DBZDR0=sym.Function('DBZDR')(t)
pt=(p**2-pr**2-pz**2)**0.5
Ar=sym.Function('A_r')(r,t,z)
At=sym.Function('A_t')(r,t,z)
Ar0=sym.Function('A_r')(t)
At0=sym.Function('A_t')(t)
r0=sym.symbols('R')
t0=sym.Function('t0')(t)
z0=sym.symbols('z_0')
pr0=sym.symbols('p_r')
pt0=sym.symbols(r'p_{\theta}')
pz0=sym.symbols(r'p_{z0}')

B=sym.Function('B')(r,t)
C=sym.Function('C')(r,t)
DBDT=sym.Function(r'\partial B/\partial \theta')(r,t)
DCDT=sym.Function(r'\partial C/\partial \theta')(r,t)
```

```
DBDR=sym.Function(r'\partial B/\partial r')(r,t)
DCDR=sym.Function(r'\partial C/\partial r')(r,t)

DBDTDR=sym.Function(r'\partial^{2} B/\partial r \partial t')(r,t)
DCDTDR=sym.Function(r'\partial^{2} C/\partial r \partial t')(r,t)
DBDRDR=sym.Function(r'\partial^{2} B/ \partial r^2')(r,t)
DCDRDR=sym.Function(r'\partial^{2} C/ \partial r^2')(r,t)

DBDTDRDR=sym.Function(r'\partial^{3} B/\partial r^{2} \partial t')(r,t)
DCDTDRDR=sym.Function(r'\partial^{3} C/\partial r^{2} \partial t')(r,t)
DBDRDRDR=sym.Function(r'\partial^{3} B/\partial r^{3} ')(r,t)
DCDRDRDR=sym.Function(r'\partial^{3} C/\partial r^{3} ')(r,t)


B0        =sym.symbols('BZ')
C0        =sym.symbols('C0')
DBDT0     =sym.symbols('DBZDT')
DCDT0     =sym.symbols('DCDT')
DBDR0     =sym.symbols('DBZDR')
DCDR0     =sym.symbols('DCDR')

DBDTDR0   =sym.symbols('D2BDTDR')
DCDTDR0   =sym.symbols('D2CDTDR')
DBDRDR0   =sym.symbols('D2BDR')
DCDRDR0   =sym.symbols('D2CDR')

DBDTDRDR0 =sym.symbols('DBZ3DTDR2')
DCDTDRDR0 =sym.symbols('DC3DTDR2')
DBDRDRDR0 =sym.symbols('DB3DR')
DCDRDRDR0 =sym.symbols('DC3DR')

####eqt for cyclops
H=-r*sym.sqrt(p**2-(pr-q*Ar)**2-pz**2)-q*r*At
HHI=sym.hessian(H,(r,pr,z,pz))
HH=HHI.subs(sym.sqrt(p**2-pz**2-(pr-q*Ar)**2),pt0)
HH=HH.subs((pr-q*Ar),pr0)
HH=HH.subs(pz,pz0)

J=sym.Matrix([[0,1,0,0],[-1,0,0,0],[0,0,0,1],[0,0,-1,0]])
HH=sym.simplify(J*HH)

HH=HH.subs(At,-1/r*sym.integrate((B*r),r)+DCDR*z+DBDR*z**2/2)
```

```
HH=HH.subs(Ar,1/r*(-DCDT*z-DBDT*z**2/2))

HH=sym.simplify(HH)


HH=HH.subs(sym.diff(B,r),DBDR)
HH=HH.subs(sym.diff(C,r),DCDR)


HH=HH.subs(sym.diff(DBDT,r),DBDTDR)
HH=HH.subs(sym.diff(DCDT,r),DCDTDR)
HH=HH.subs(sym.diff(DBDR,r),DBDRDR)
HH=HH.subs(sym.diff(DCDR,r),DCDRDR)

HH=HH.subs(sym.diff(DBDTDR,r),DBDTDRDR)
HH=HH.subs(sym.diff(DCDTDR,r),DCDTDRDR)
HH=HH.subs(sym.diff(DBDRDR,r),DBDRDRDR)
HH=HH.subs(sym.diff(DCDRDR,r),DCDRDRDR)


HH=HH.subs(DBDTDRDR,DBDTDRDR0)
HH=HH.subs(DCDTDRDR,DCDTDRDR0)
HH=HH.subs(DBDRDRDR,DBDRDRDR0)
HH=HH.subs(DCDRDRDR,DCDRDRDR0)


HH=HH.subs(DBDTDR,DBDTDR0)
HH=HH.subs(DCDTDR,DCDTDR0)
HH=HH.subs(DBDRDR,DBDRDR0)
HH=HH.subs(DCDRDR,DCDRDR0)

HH=HH.subs(DBDR,DBDR0)
HH=HH.subs(DCDR,DCDR0)
HH=HH.subs(DBDT,DBDT0)
HH=HH.subs(DCDT,DCDT0)

HH=HH.subs(B,B0)
HH=HH.subs(C,C0)

HH=HH.subs(z,z0)
HH=HH.subs(r,r0)
```

```
HH=sym.simplify(HH)
HH=HH.subs(q,1)
HH=sym.simplify(HH)
HH

###output for krout.f
R11,R21,R31,R41,R12,R22,R32,R42,R13,R23,R33,R43,R14,R24,R34,R44=sym.symbols('R11,R
R0=sym.Matrix([[R11,R21,R31,R41],[R12,R22,R32,R42],[R13,R23,R33,R43],[R14,R24,R34,
HH=HH*R0
print(HH[0,0])
print(HH[1,0])
print(HH[2,0])
print(HH[3,0])

print(HH[0,1])
print(HH[1,1])
print(HH[2,1])
print(HH[3,1])

print(HH[0,2])
print(HH[1,2])
print(HH[2,2])
print(HH[3,2])

print(HH[0,3])
print(HH[1,3])
print(HH[2,3])
print(HH[3,3])
```

6.2 CYCLOPS repository

https://gitlab.triumf.ca/beamphys/cyclops.git