FUTURE
CIRCULAR
COLLIDER

REVIEW OF TAPERING FOR FCC-EE

K.D.J. André

With materials from M. Rakic, A. Doblhammer, B. Härer and G. Roy

Accelerator Design Meeting #197 & FCCIS Meeting #

Doblhammer, Harer, Holzer, IPAC2016

- 100km racetrack lattice,
- 175 GeV with 2 RF sections

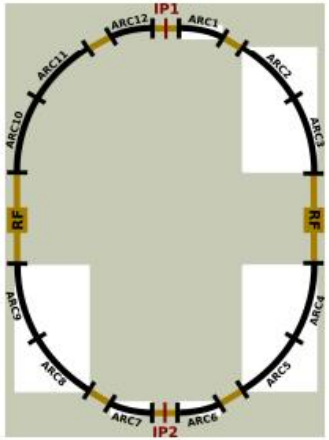


Figure 3: Schematic of the FCC-ee racetrack lattice. Arcs are depicted in black, straight sections in yellow. IPs and RF sections are marked as well.

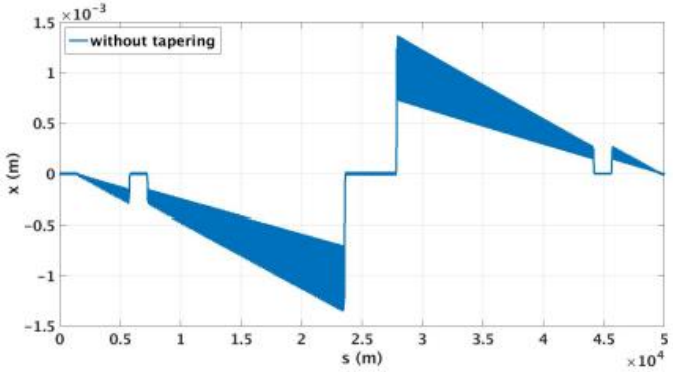
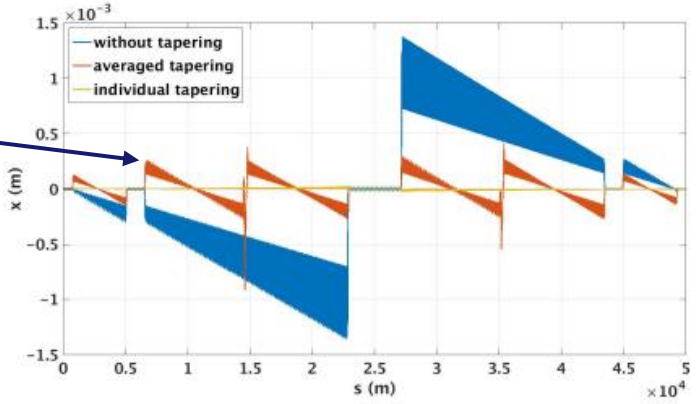


Figure 4: Orbit before tapering for a half ring

300 microns



Doblhammer, Harer, Holzer, IPAC2016

The residual optics and dispersion distortions can be corrected solely by rematching the optics using free quadrupoles in dispersion suppressors and matching sections. With this method, the achieved goal of beta and dispersion beats below 10% could always be achieved.

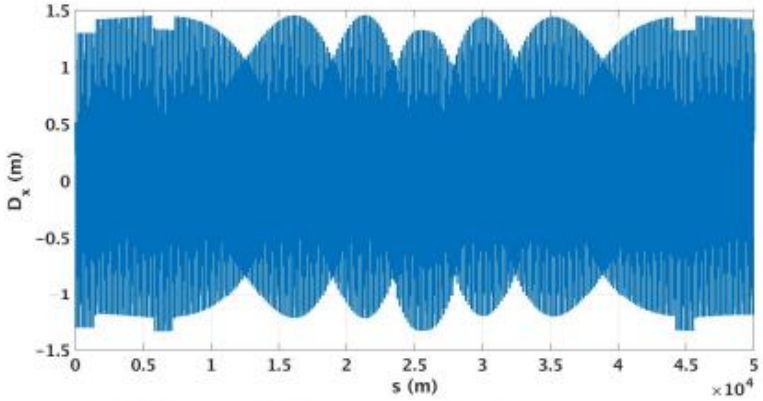


Figure 8: Horizontal Dispersion D_x after dipole tapering

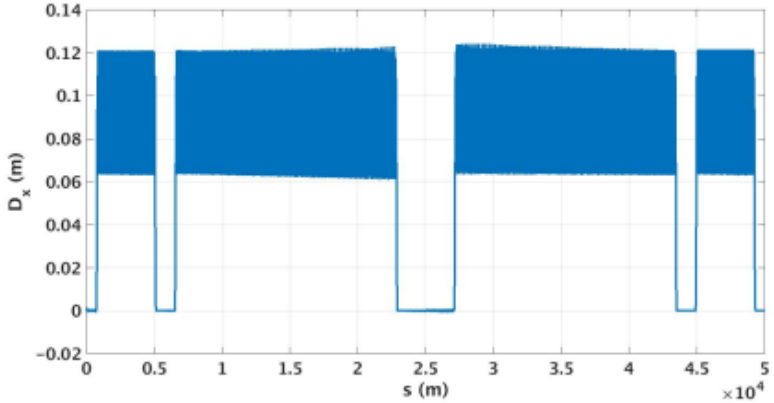
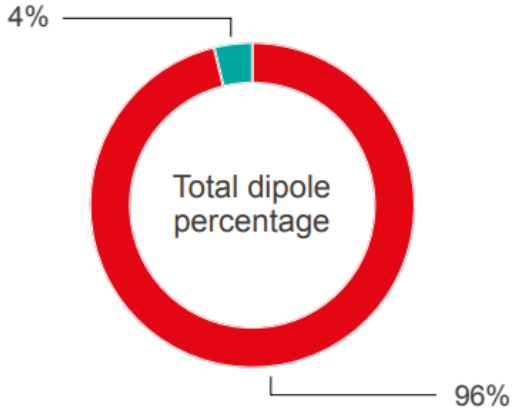


Figure 10: Horizontal Dispersion after matching. The deviations from the ideal dispersion are well below 10%.

M. Rakic's 1-year master's project (2021)| link

First exploration of simplified tapering

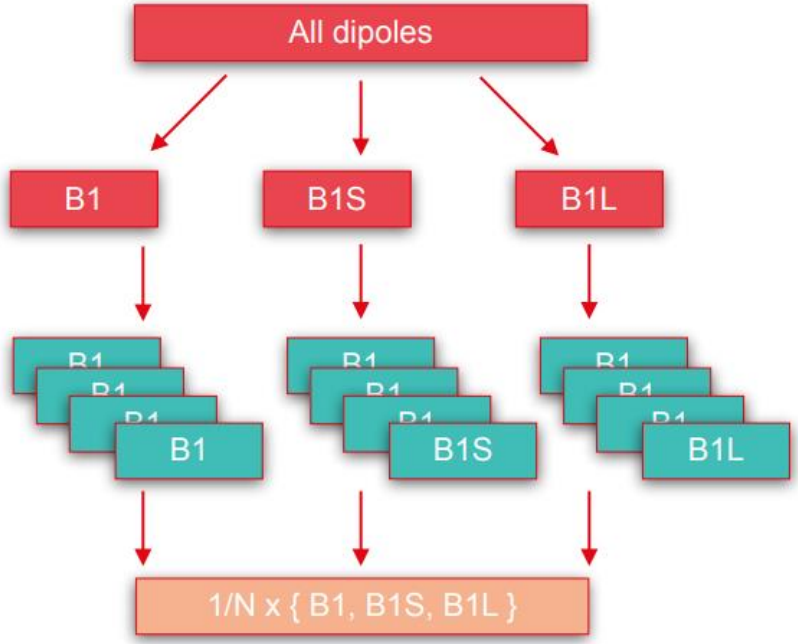
- Identification of 100 microns figure of merit for maximum orbit (K.Oide)
- Identification of most important lattice families (supplied by K.Oide)
  - Main arc dipoles at sextupole-free sections B1
  - Main arc dipoles at long sextupole sections B1L
  - Main arc dipoles at short sextupole sections B1S
- All other dipole families were kept as are (e.g. dispersion suppressors, connecting arc, IP upstream and downstream)
- For B1, B1S and B1L average tapering was employed, all other dipoles received individual tapering



First exploration of simplified tapering

Milica Rakic

- Reasons for looking into simplified tapering:
  - Tapering of individual magnets is expensive
  - Successful simplified tapering for Higgs physics lattice can greatly reduce costs
  - Valuable input for future dipole magnet design phase
- Average tapering entailed:
  - Individual tapering of all dipoles, quadrupoles and sextupoles
  - Segmenting three main dipole families in 4 section based on RF and IP positions
  - Further segmentation into number-adjusted sub-families
  - Taking values from individual tapering and averaging them over sub-families
  - Applying obtained values to dipole strengths of chosen families



Tapering to 50-micron orbit tolerance

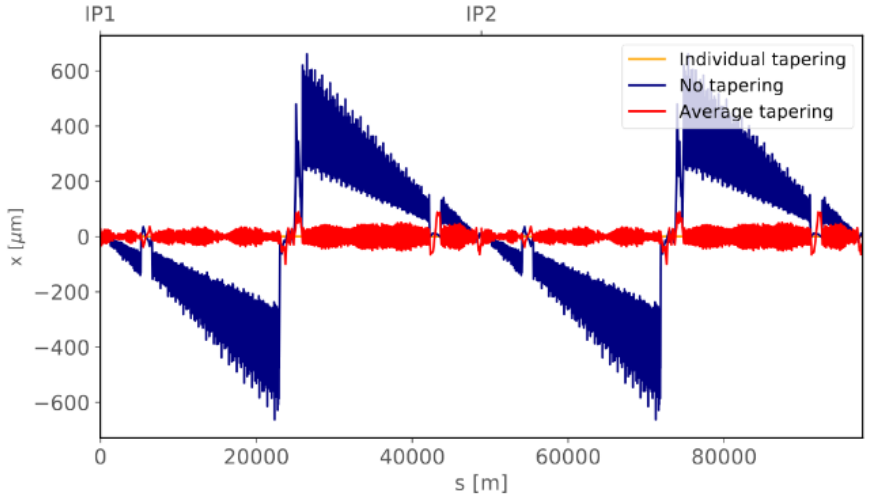


Figure 5: Beam orbit of 50 micron after average tapering is shown in red.

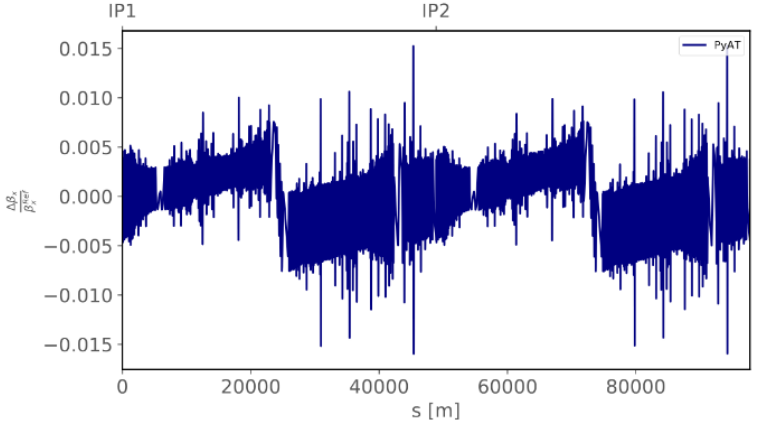
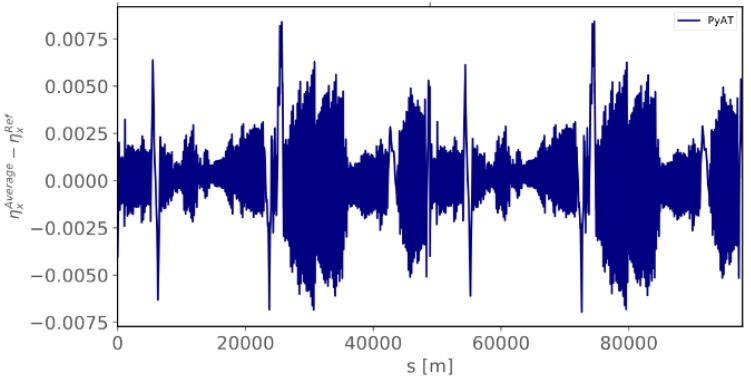


Figure 6: Relative beta function, as compared to no tapering case, for 50 micron orbit.



Tapering to 100-micron orbit tolerance

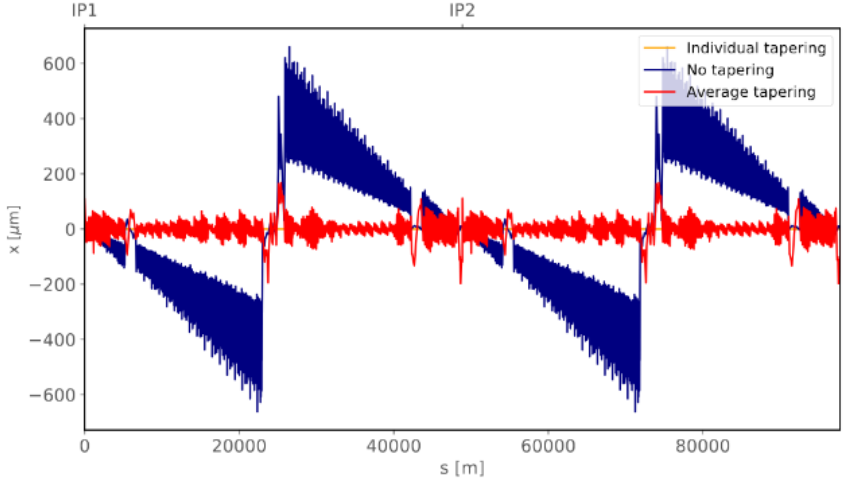


Figure 8: Beam orbit of 100 micron after average tapering is shown in red.

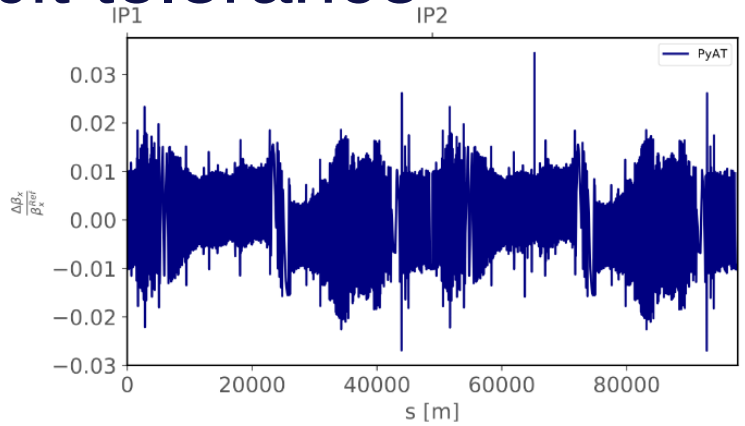
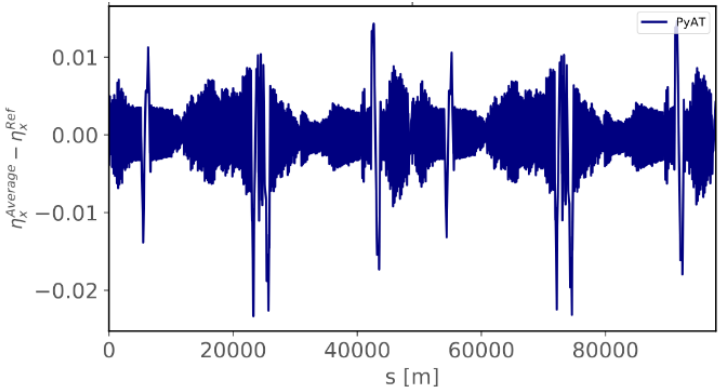


Figure 9: Relative beta function, as compared to no tapering case, for 100 micron orbit.



Tapering to 200-micron orbit tolerance

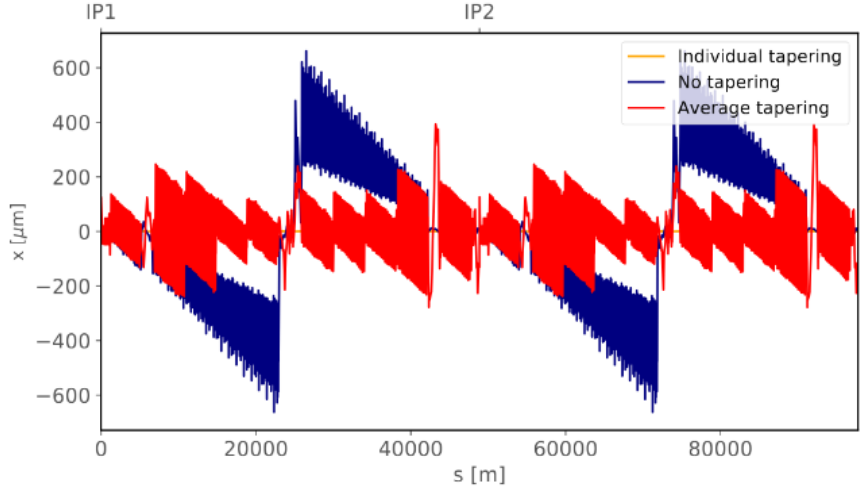


Figure 11: Beam orbit of 200 micron after average tapering is shown in red.

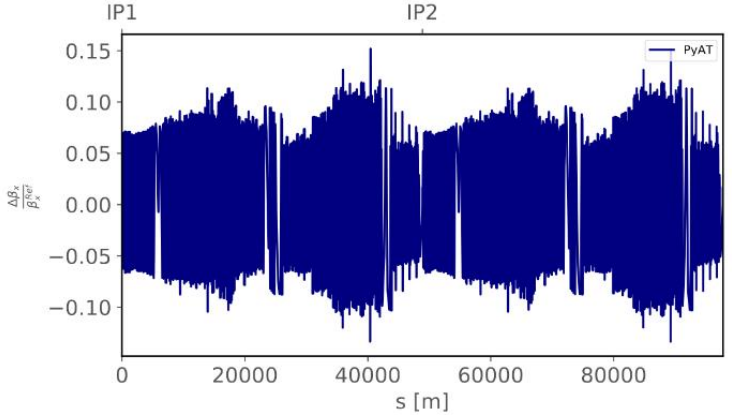
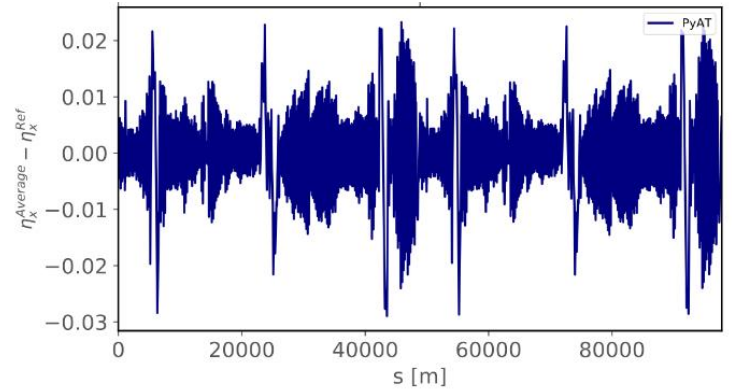


Figure 12: Relative beta function, as compared to no tapering case, for 200 micron orbit.



Conclusion

Github link
Thesis link

- TPIV results:
  - Testing of first PyAT optics calculation with synchrotron radiation
  - Testing of first tapering for PyAT
  - Development of simplified tapering scheme for Higgs lattice in PyAT
- Current PyAT developments:
  - Successful tapering
  - Successful implementation of optics calculations with synchrotron radiation (reference E.Forest)
  - Focusing on further exploring:
    - Optics function degradation below 15%
    - Large distortion of dispersion function
- Moving on:
  - Preforming the matching (optimisation) for the dipoles using averaged state as initial condition
  - Performing orbit optimisation with orbit correctors
  - Rematching optics after alternative tapering scheme

Tapering implementation in MADX

TWISS, TAPERING;

TAPERING	Adjust the strengths of the quadrupoles and sextupoles in order to compensate for the offset in energy. This flag triggers a call to the TAPER command with default parameters and no output file.

TAPER function use:

Setting STEPSIZE to a non-zero value is useful to explore tolerances of a beamline to tapering and determine a likely strategy for stepwise and/or piecewise tapering, *i.e.* applying the same tapering to a set of elements or part of a beamline.

TAPER uses the closed orbit finding algorithm of the TWISS package to calculate the tapering values of all elements if STEPSIZE is zero. If STEPSIZE is non-zero it could be difficult to ensure a closure condition for the tapering values and the first turn algorithm of MAD-X is used instead, also from the TWISS package. Hence for evaluation of tapering of an open line the STEPSIZE should be set to a non-zero value.

Every call to the TAPER command first resets all tapering values to zero in order to be able to adjust the BEAM before recalculating the orbit with tapering. This is also the case when the TAPER command is called internally from TWISS or MATCH command with option TAPERING=true.

TAPER function

TAPER calculates the adjustment to the strengths of elements to account for small momentum variations through RF cavities or synchrotron radiation.

```
TAPER, ITERATE=integer, STEPSIZE=real, FILE=filename, RESET=logical;
```

The attributes for the TAPER command are:

ITERATE	the number of iterations through the element maps. Because the energy loss in dipoles is strongly non linear, MAD-X can cycle ITERATE times through the dipole map to converge to the appropriate tapering value of the element corresponding to the energy loss through the adjusted bending magnet. For quadrupoles, sextupoles, octupoles and multipoles, a tracking through the map is done only once for any positive non-zero value of ITERATE. If ITERATE=0 is specified, the tapering is only calculated from the momentum deviation at the entrance of the element and no account is taken of the energy loss through the current element. A maximum value of ITERATE=10 is enforced. (Default: 3)
STEPSIZE	the tapering step size for stepwise tapering. if STEPSIZE is non-zero and positive, tapering values are calculated as multiples of STEPSIZE. (Default: 0)
FILE	causes MAD-X to write the tapering values applied to each sequence node to the file specified. The file can be LOADed in a subsequent MAD-X run. (Default: "taper.madx")
RESET	causes MAD-X to reset tapering values of all nodes in the current sequence to zero values. All other command parameters are ignored.

Tapering implementation in SAD

Link to the doc
RADTAPER function details:

Scales all magnets except for solenoids according to the local momentum (DDP) of the closed orbit.

It uses the **average of DDPs at the entrance and the exit**.

RADTAPER sets the momentum deviation of the closed orbit to DP0=dp/p0, which is an arbitrary choice of an under-deterministic problem of tapering. The difference in the path length around the ring is adjusted by automatically updating FSHIFT which is the relative shift df/f0 of the revolution (or rf) frequency in a ring.

Tapering implementation in Xsuite

Done by configuring the phase of the RF cavities to compensate for the energy loss and adapting the strength of the magnets to the local momentum of the particle on the closed orbit.

```
line.config.XTRACK_MULTIPOLE_NO_SYNRAD = True
with xt.freeze_longitudinal(line):
    particle_on_co = line.find_closed_orbit()
line.config.XTRACK_MULTIPOLE_NO_SYNRAD = False

beta0 = float(particle_on_co._xobject.beta0[0])

# Check whether compensation is needed
p_test = particle_on_co.copy()
p_test.delta = delta0
line.track(p_test, turn_by_turn_monitor='ONE_TURN_EBE')
mon = line.record_last_track
eloss = -(mon.ptau[0, -1] - mon.ptau[0, 0]) * p_test.p0c[0]
if p_test.state[0] > 0 and abs(eloss) < p_test.energy0[0] * rtol_eneloss:
    if verbose: _print("  - No compensation needed")
    return
```

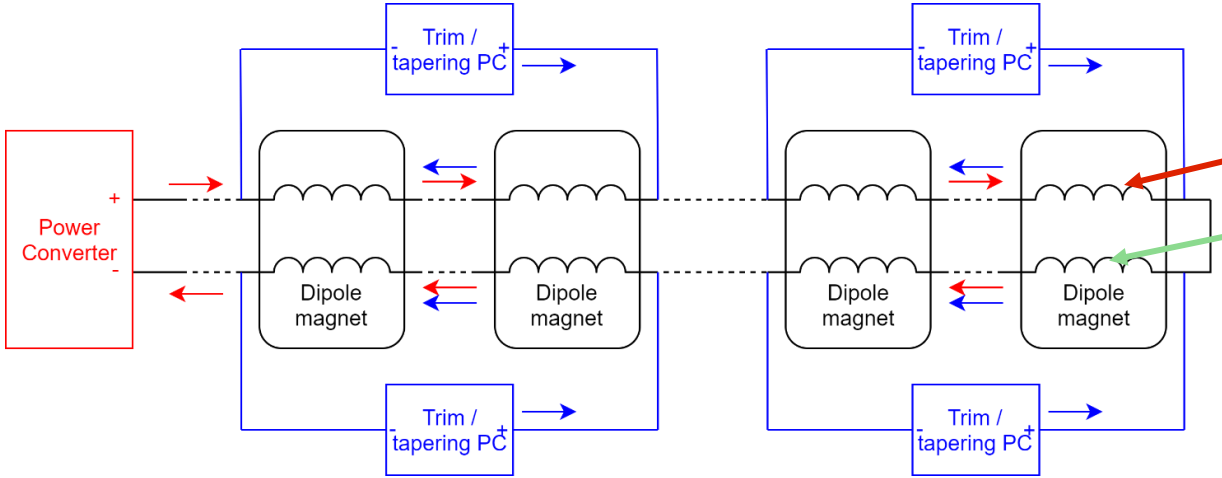
```
# save voltages
v_setter = line.attr._cache['voltage'].multisetter
f_setter = line.attr._cache['frequency'].multisetter
lag_setter = line.attr._cache['lag'].multisetter
lag_taper_setter = line.attr._cache['lag_taper'].multisetter

v0 = v_setter.get_values()
f0 = f_setter.get_values()
lag_zero = lag_setter.get_values()

eneloss_partitioning = v0 / v0.sum()
```

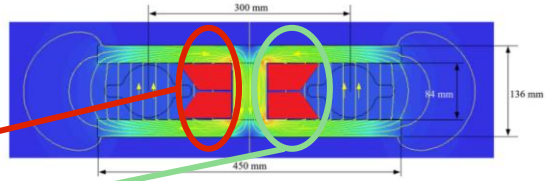
```
# Put all cavities on crest and at zero frequency
lag_taper_setter.set_values(90. - lag_zero)
v_setter.set_values(np.zeros_like(v_setter.get_values()))
f_setter.set_values(np.zeros_like(f_setter.get_values()))

if verbose: _print("Share energy loss among cavities (repeat until energy loss is zero)")
with xt.line._preserve_config(line):
    line.config.XTRACK_MULTIPOLE_TAPER = True
    line.config.XTRACK_DIPOLEEDGE_TAPER = True

i_iter = 0
while True:
    p_test = particle_on_co.copy()
    p_test.delta = delta0
    line.track(p_test, turn_by_turn_monitor='ONE_TURN_EBE')
    mon = line.record_last_track

    if record_iterations:
        line._tapering_iterations.append(mon)

    eloss = -(mon.ptau[0, -1] - mon.ptau[0, 0]) * p_test.p0c[0]
    if verbose: _print(f"Energy loss: {eloss:.3f} eV             ")#, end='\r', flush=True)

    if eloss < p_test.energy0[0] * rtol_eneloss:
        break

    v_setter.set_values(v_setter.get_values()
                        + eloss * eneloss_partitioning)

    i_iter += 1
    if i_iter > max_iter:
        raise RuntimeError("Maximum number of iterations reached")

delta_taper_full = 0.5*(mon.delta[0, :-1] + mon.delta[0, 1:]) # last point is added by the monitor
```

Dipole tapering

Trim solution

- Possible to have trim power converters - 2 x 30 per sector – 720 units total
- Very small power (preliminary 22 A) “ very small voltage → not an issue
- Installation in alcoves
- A compromise should be studied to evaluate losing one trim per beam per sector and global tapering still acceptable





Trims not directly referred to ground → floating

Thank you
for your attention.