Tuning of the Compact Linear Collider Final-Focus System

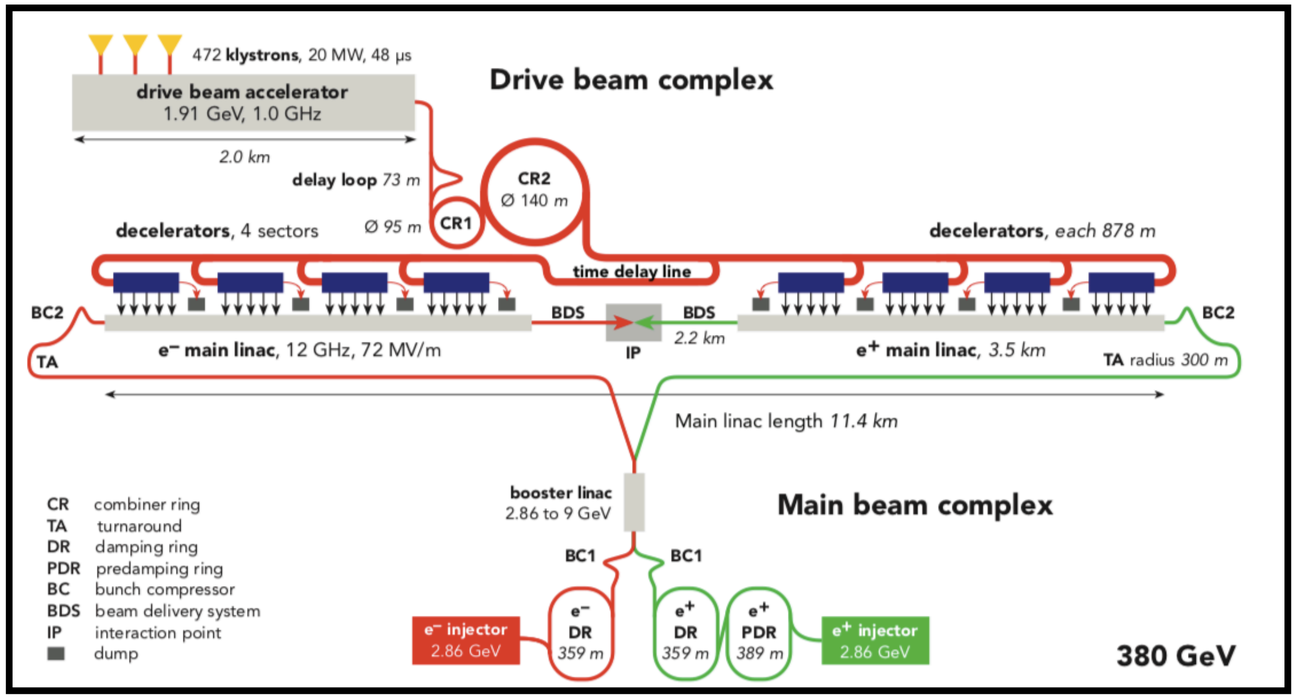
Jim Ögren

FREIA division, Uppsala University
December 2, 2019

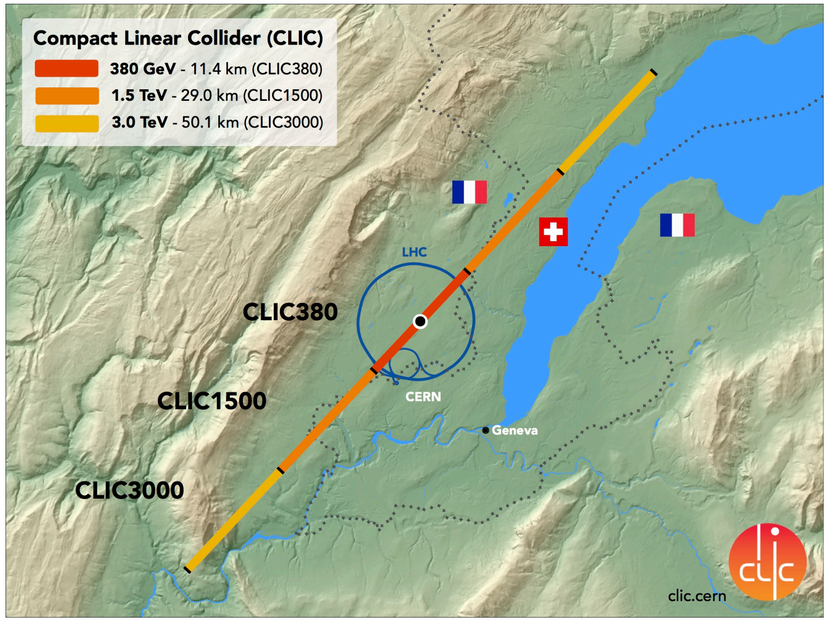
Outline

- Introduction
- The Compact Linear Collider
- The final-focus system
- Beam-beam interaction
- Tuning simulation
- Machine Learning
- Conclusion

The Compact Linear Collider (CLIC)



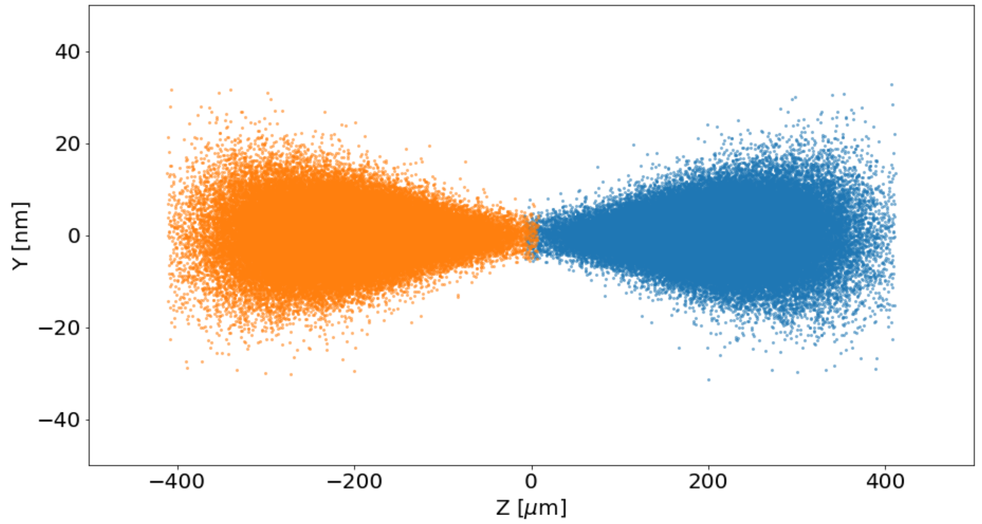
- Proposed linear electron—positron collider at CERN
- Initial center-of-mass energy of 380 GeV for studying Higgs boson and top quark
- Potential upgrade to 1.5 TeV and 3 TeV



- Normal-conducting, high-gradient acceleration
- Unique two-beam accelerating scheme
- Emittance preservation
- Small beam sizes and high luminosity

Luminosity

- The purpose of a collider is to produce as many collisions as possible
- Luminosity [$cm^{-2}s^{-1}$]
- Event rate = Luminosity*cross-section



Luminosity for a linear collider:

$$\mathscr{L} = H_D \frac{N^2}{\sigma_x \sigma_y} n_b f_{\mathrm{rep}}$$

correction factor

Particles per bunch

Bunches per pulse

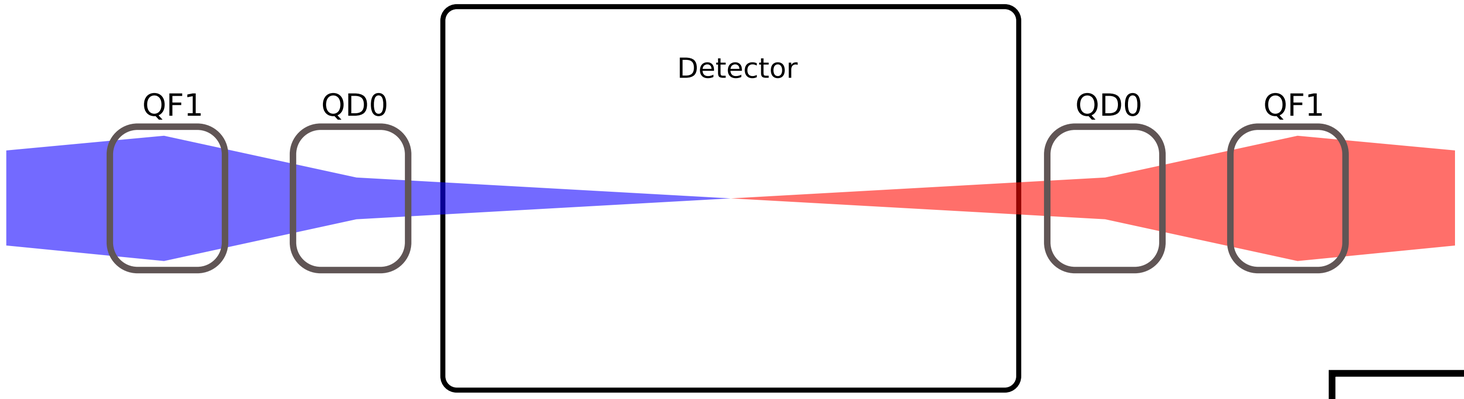
repetition rate

Beam size at interaction point

CLIC 380 GeV Beam parameters

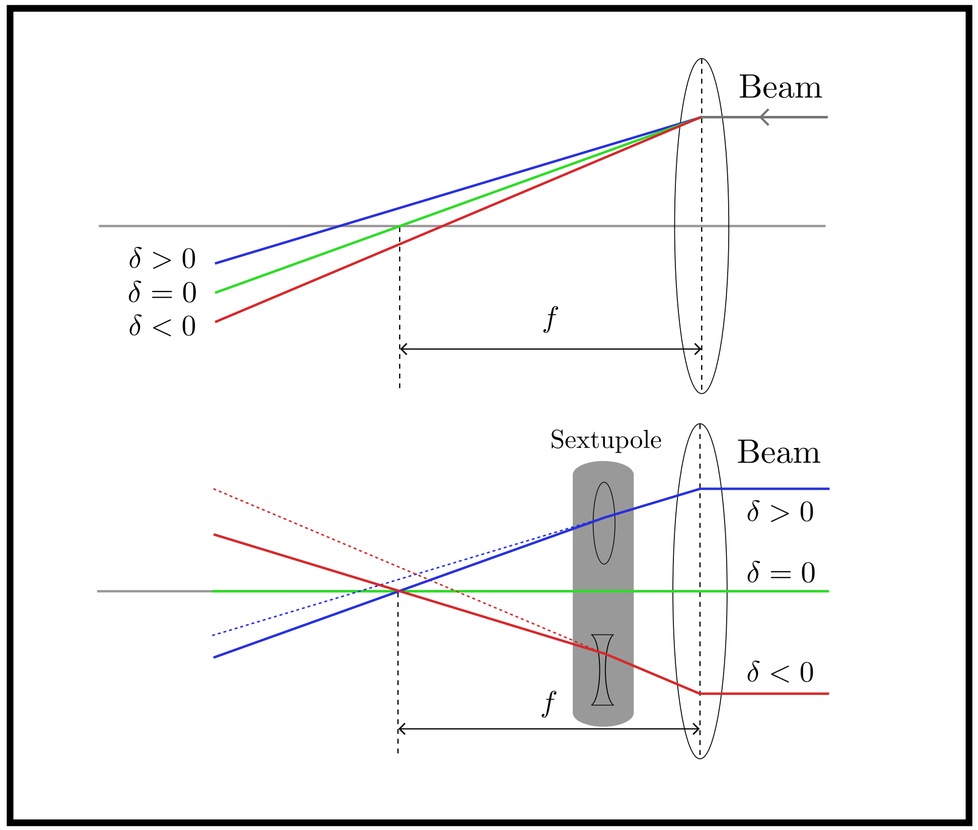
Norm. emittance (end of linac) $\gamma\epsilon_x/\gamma\epsilon_y$	[nm]	900 / 20
Norm. emittance (IP) $\gamma\epsilon_x/\gamma\epsilon_y$	[nm]	950 / 30
Beta function (IP) β_x^*/β_y^*	[mm]	8.2 / 0.1
Target IP beam size σ_x^*/σ_y^*	[nm]	149 / 2.9
Bunch length σ_z	[μm]	70
rms energy spread δ_p	[%]	0.35
Bunch population N_e	[10^9]	5.2
Number of bunches n_b		352
Repetition rate f_{rep}	[Hz]	50
Luminosity $\mathcal{L}_{\mathrm{total}}$	[$10^{34}\mathrm{cm}^{-2}\mathrm{s}^{-1}$]	1.5
Peak luminosity $\mathcal{L}_{1\%}$	[$10^{34}\mathrm{cm}^{-2}\mathrm{s}^{-1}$]	0.9

Making small beams

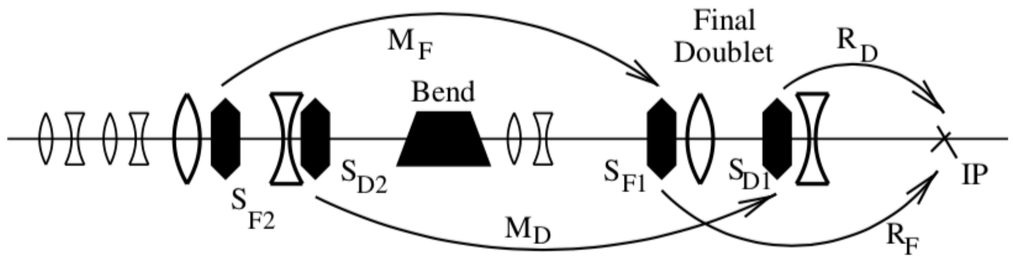


Final doublet
Two strong quadrupoles focus the beam transversely

- Chromaticity: particles with different energies get different focusing
- Add dispersion (energy sorting)
- Sextupoles provide position-dependent focusing



380 GeV final-focus system



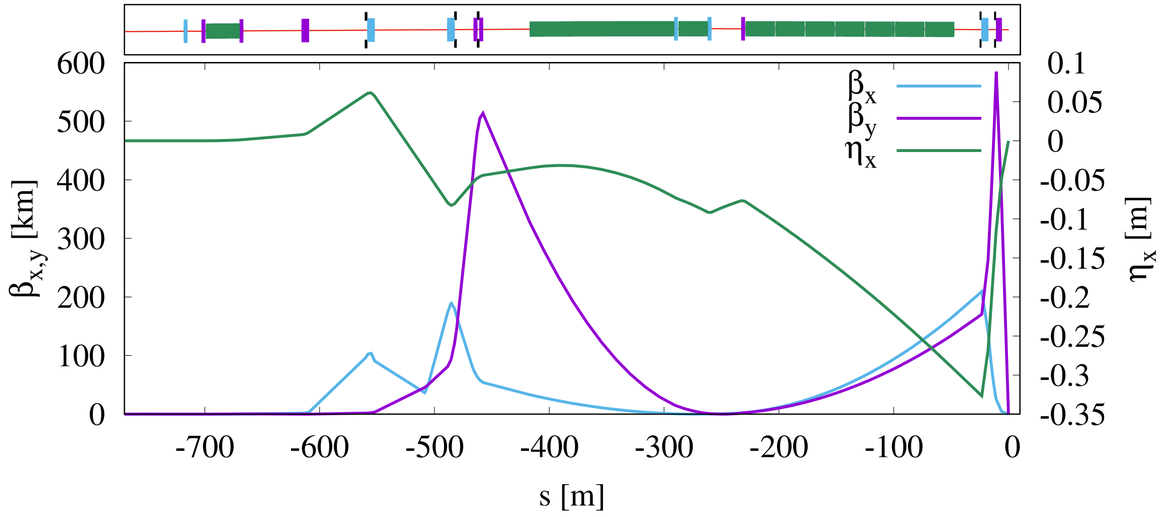
P. Raimondi and A. Seryi, "Novel Final Focus Design for Future Linear Colliders", *Phys. Rev. Lett.* 86, 3779 (2001).

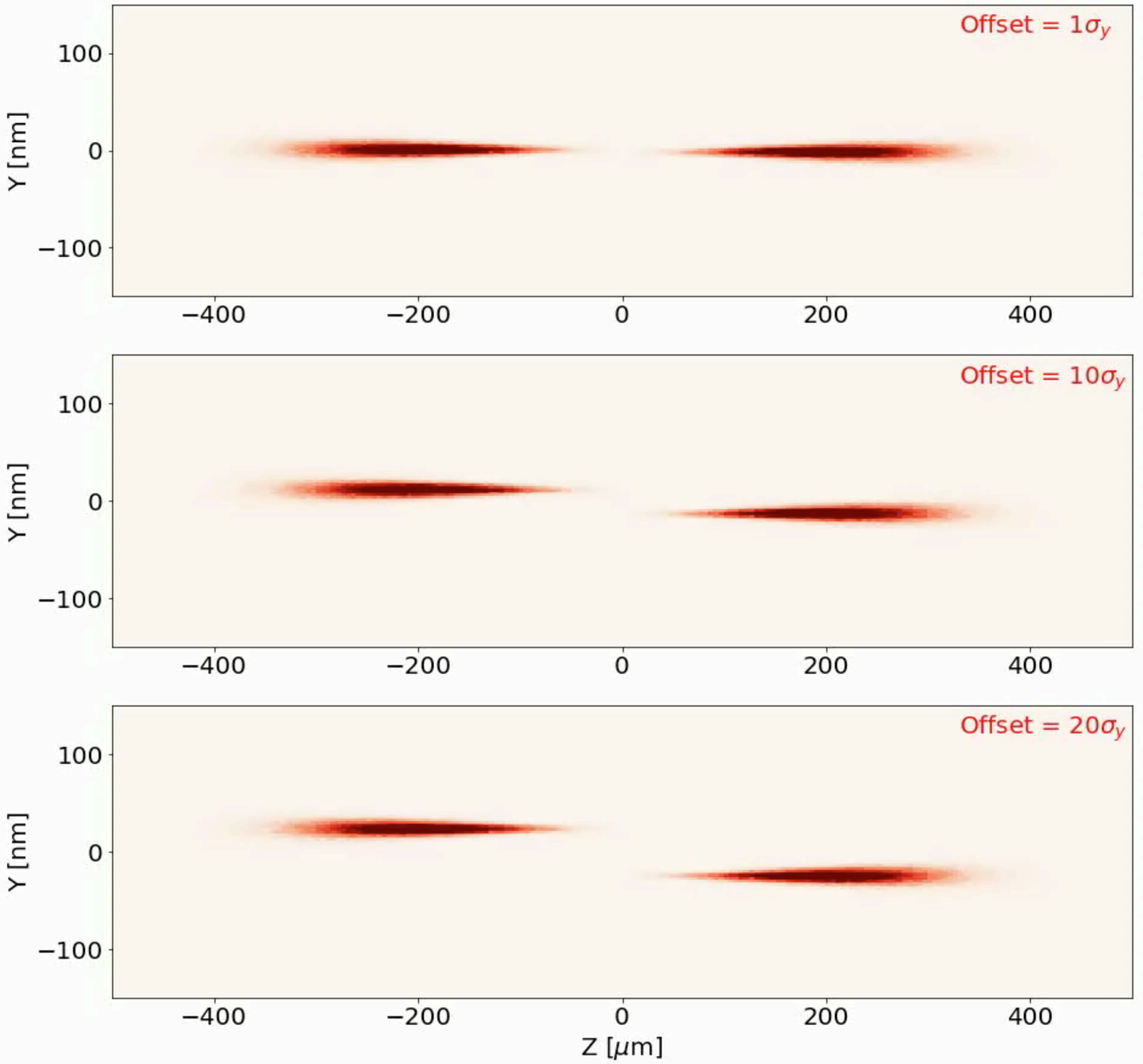
Local chromaticity scheme

- Interleaved sextupoles
- Cancel aberrations
- Trade-off between strong bends (SR) and strong sextupoles (aberrations)
- Highly nonlinear
- Tuning is challenging

CLIC FFS system:

- 780 m long
- 20 quads, 6 sextupoles and 2 octupoles
- 385 m bending magnets
- L* = 6 m
- 100 μm vertical beta function at IP

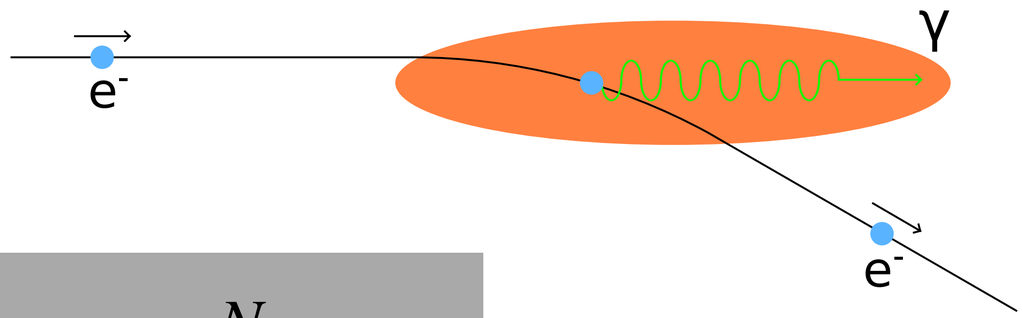



Offset = 1σ_y
Offset = 10σ_y
Offset = 20σ_y
Y [nm]
100
0
−100
−400
−200
0
200
400
Z [μm]

Beam-beam interaction

- Interesting physics
- Disruption is high
- Very different from beam-beam interactions in circular colliders
- Strong electromagnetic interaction at collision
- Beamstrahlung radiation
- Incoherent pair production
- Beam-beam signals can be used in tuning

Beamstrahlung



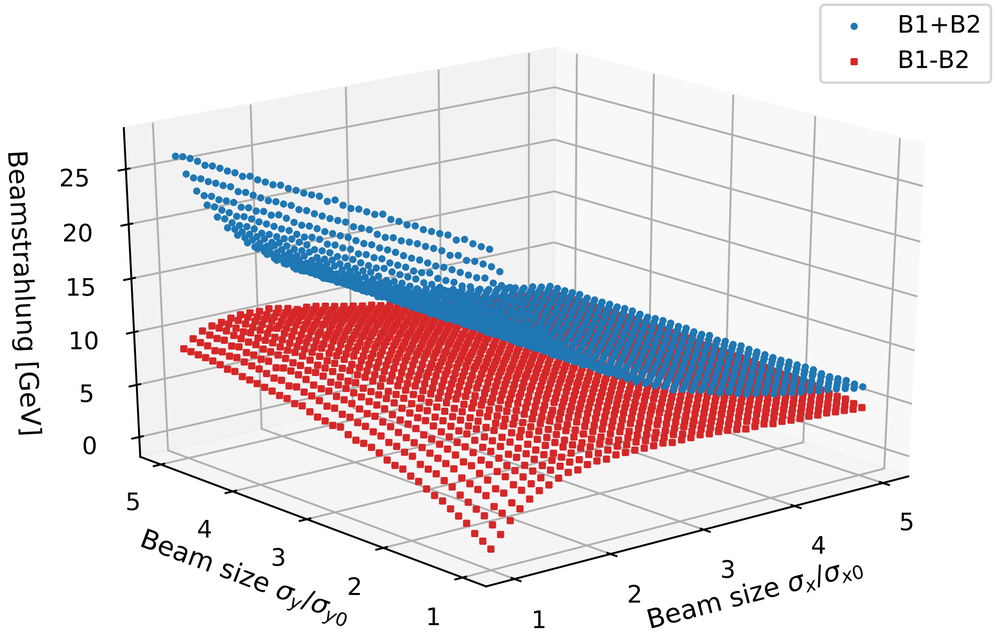
Beamstrahlung:
Particles in the one beam are bent by the EM fields from the other beam during collision and emits synchrotron radiation.

$$n_\gamma \propto \frac{N}{\sigma_x + \sigma_y}$$

$$E_\gamma \propto \frac{N}{(\sigma_x + \sigma_y)\sigma_z}$$

$$\mathscr{L} \propto \frac{N^2}{\sigma_x \sigma_y}$$

- Flat beams are better
- Beamstrahlung can also be used as a beam size indicator
- Measure muons generated from photon interactions in the water dump

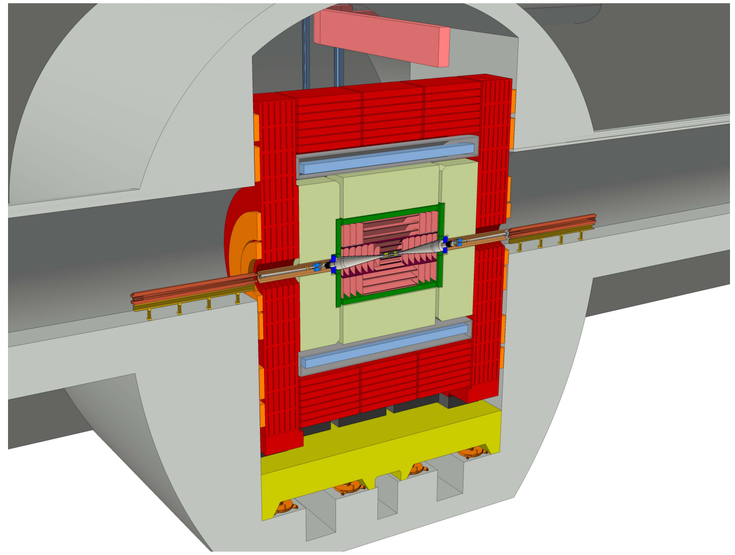


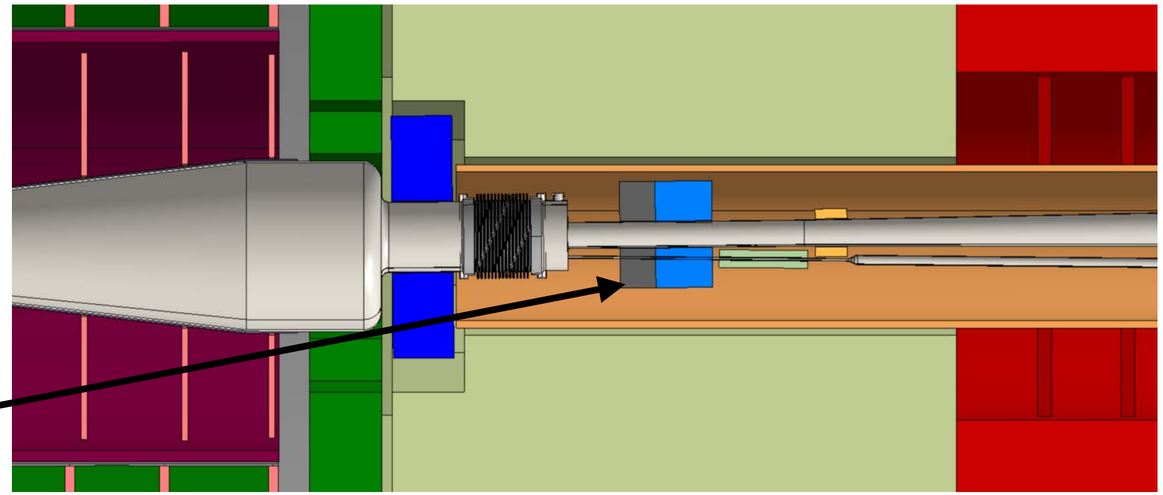
Incoherent pairs

Incoherent e^-e^+ pairs:

- In the beam-beam interaction e^-e^+ pairs are produced via scattering of beam particles and/or beamstrahlung photons
- Signal proportional to luminosity
- Used for tuning signal

The CLIC detector





Spent beam and beamstrahlung photons

BeamCal
Detector in forward region for inc. pairs

Simulation: tuning with static imperfections

Monte Carlo study:

- Seed of 500 machines with randomly distributed imperfections
- Use tolerances as rms
- Assume ideal feedback and head-on collisions

Imperfection	Specified tolerance (rms error)	Elements
Resolution	20 nm	BPMs
Transverse misalignments	10 μm (20 μm)	BPMs, all magnets (multipoles)
Roll errors	100 μrad	BPMs, all magnets
Relative strength error	10^{-4}	All magnets

Tuning goal:

- Luminosity target: 110% of L_0 =1.5e34 $cm^{-2}s^{-1}$
- Tuning goal: 90% of machines to be successfully tuned

Simulation: tuning with realistic signals

1) Beam-based alignment

- Treat electron and positron beamlines independently
- Correct trajectory and dispersion simultaneously

2) Maximize beamstrahlung power

- Maximize total beamstrahlung power (sum of two beamlines)
- Sextupole transverse position, random walk
- Tunes mainly horizontal beam size

3) Maximize total energy in inc. pairs

- Sextupole transverse position, random walk
- Tunes mainly vertical beam size

4) Sextupole knobs

- Scan sextupole knobs (transverse position)
- Maximize energy deposited from inc. pairs in BeamCal
- Use 2e4 particles and then 1e5 particles for fine tuning

5) Quadrupole and Sextupole tuning

- Random walk moving quadrupoles and sextupoles together (if needed)
- Followed by sextupole knob scan

Beam-based alignment (BBA)

- Correct for trajectory and dispersion
- Measure the response matrix (once) on the misaligned machine
- Use movement of quadrupoles instead of steering magnets

$$\begin{bmatrix} \omega_{\text{traj}} \overrightarrow{y}_{\text{traj}} \\ \omega_{\text{disp}} \overrightarrow{y}_{\text{disp}} \end{bmatrix} = \begin{bmatrix} \omega_{\text{traj}} R_{\text{traj}} \\ \omega_{\text{disp}} R_{\text{disp}} \end{bmatrix} \begin{bmatrix} \Delta \overrightarrow{x} \\ \Delta \overrightarrow{y} \end{bmatrix}$$

Check trajectory and dispersion deviation from reference

We use singular-value decomposition (SVD) to invert the system and filter out eigenvalues

$$\lambda < \frac{\lambda_{max}}{10^4}$$

Sextupole knobs - 1st order

- Transverse, collective movement of sextupoles

- The response matrix of 2nd order moments
  - SVD to find orthogonal 'knobs' (vectors from matrix V)

$$U\lambda V^T = \begin{bmatrix} \frac{\partial \sigma_{xx}}{\partial X_1} & \cdots & \frac{\partial \sigma_{xx}}{\partial X_6} & \frac{\partial \sigma_{xx}}{\partial Y_1} & \cdots & \frac{\partial \sigma_{xx}}{\partial Y_6} \\ \frac{\partial \sigma_{xx'}}{\partial X_1} & \cdots & \frac{\partial \sigma_{xx'}}{\partial X_6} & \frac{\partial \sigma_{xx'}}{\partial Y_1} & \cdots & \frac{\partial \sigma_{xx'}}{\partial Y_6} \\ \vdots & \cdots & \vdots & \vdots & \cdots & \vdots \\ \frac{\partial \sigma_{zz}}{\partial X_1} & \cdots & \frac{\partial \sigma_{zz}}{\partial X_6} & \frac{\partial \sigma_{zz}}{\partial Y_1} & \cdots & \frac{\partial \sigma_{zz}}{\partial Y_6} \end{bmatrix}$$

- Each knob is scanned over some range and luminosity is maximized

Quadrupole and sextupole random walk

The problem:

- Quadrupoles influences the linear optics: e.g. phase advances between sextupoles and dispersion at sextupole locations

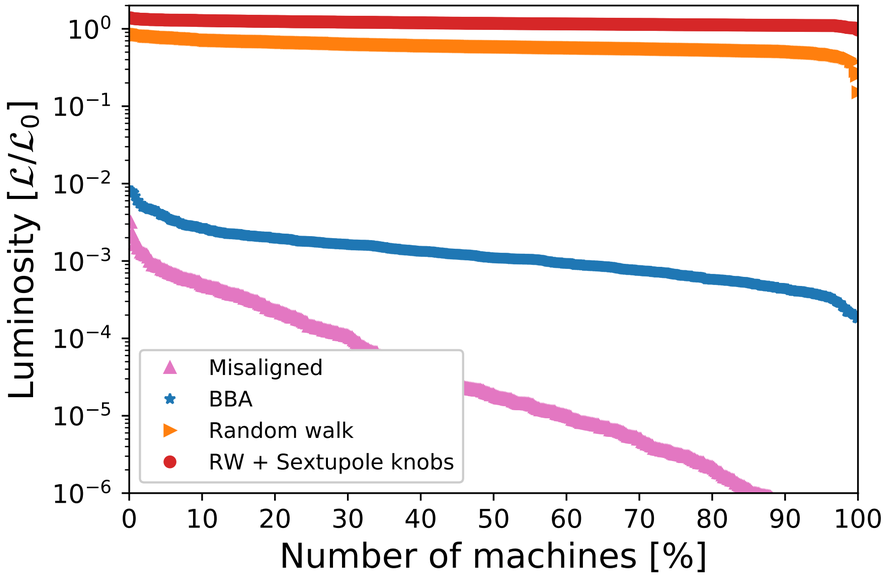
- For a given linear optics sextupole offsets can compensate many effects

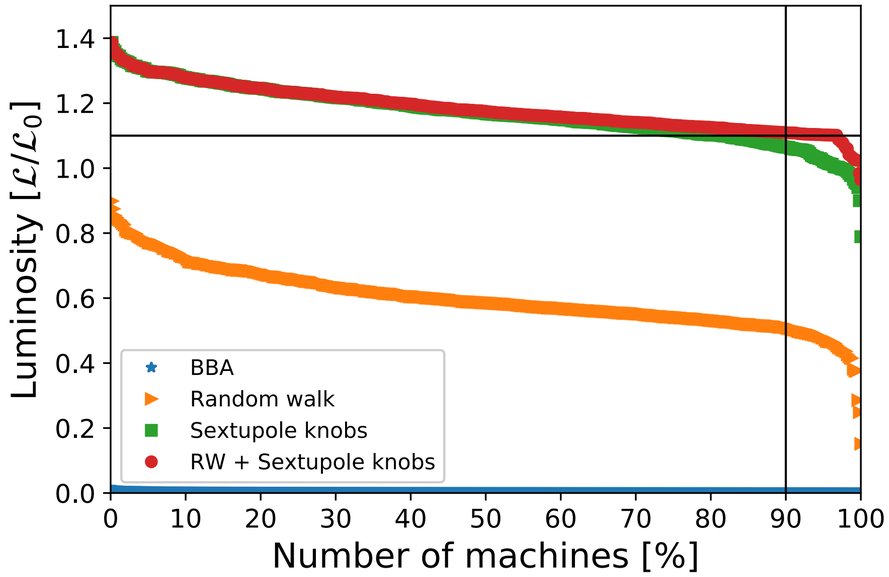
- What if the linear optics is not corrected well enough?
  - Scanning sextupole knobs gives same (sub-) optimum
  - Moving quadrupoles will only worsen luminosity

- Solution: quadrupoles and sextupoles together: **achieve similar but different scenario**

- Random walk tuning moving sextupoles and quadrupoles

Results: Luminosity histogram



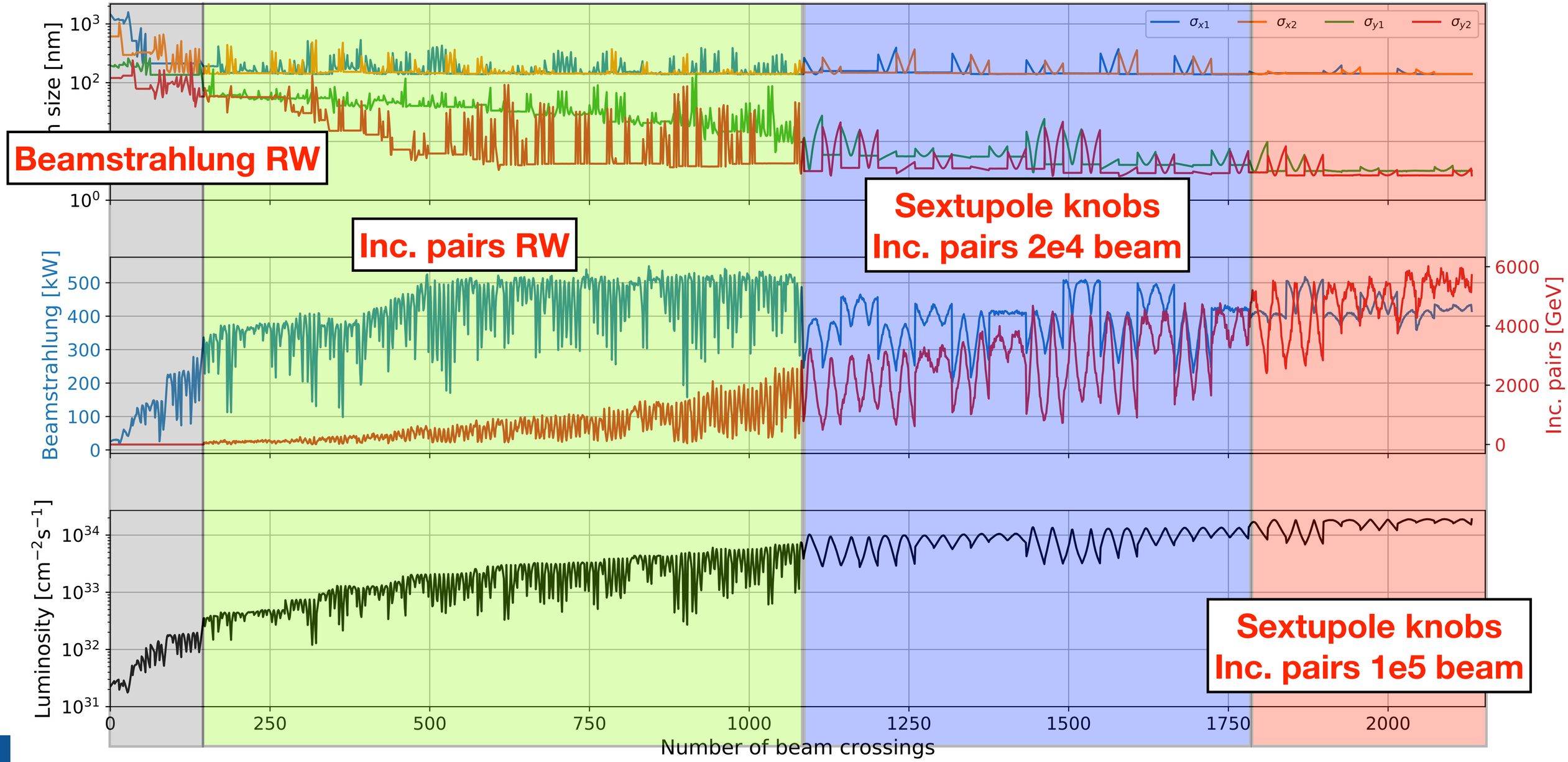


- Misaligned machine: 3-8 orders of magnitude from nominal luminosity
- After BBA: 2-4 orders of magnitude
- Random walk (beamstrahlung and pairs): within 10% of nominal luminosity

Final results: 484 machines successfully tuned

484/500 = **96.8% success rate**

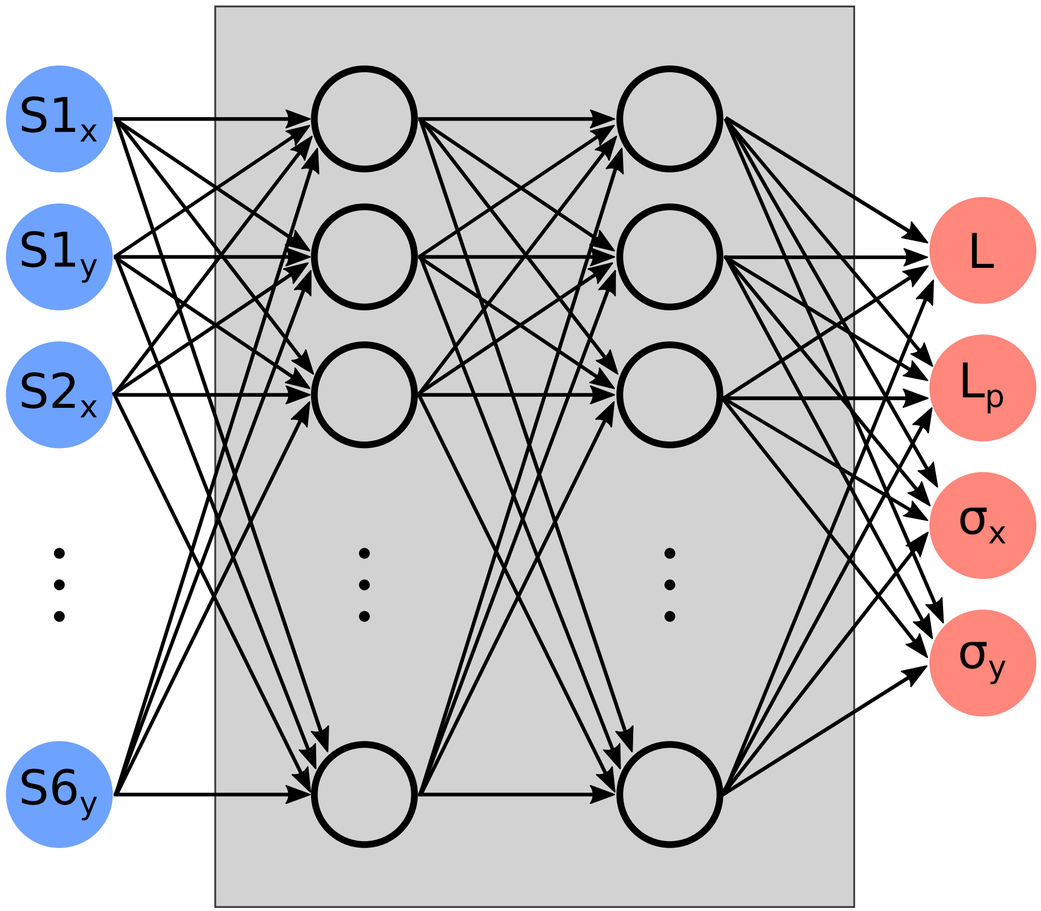
Tuning evolution of median machine



Machine Learning Application

Sextupole surrogate model

Model sextupole transverse positions to Luminosity. Goal: fast estimator



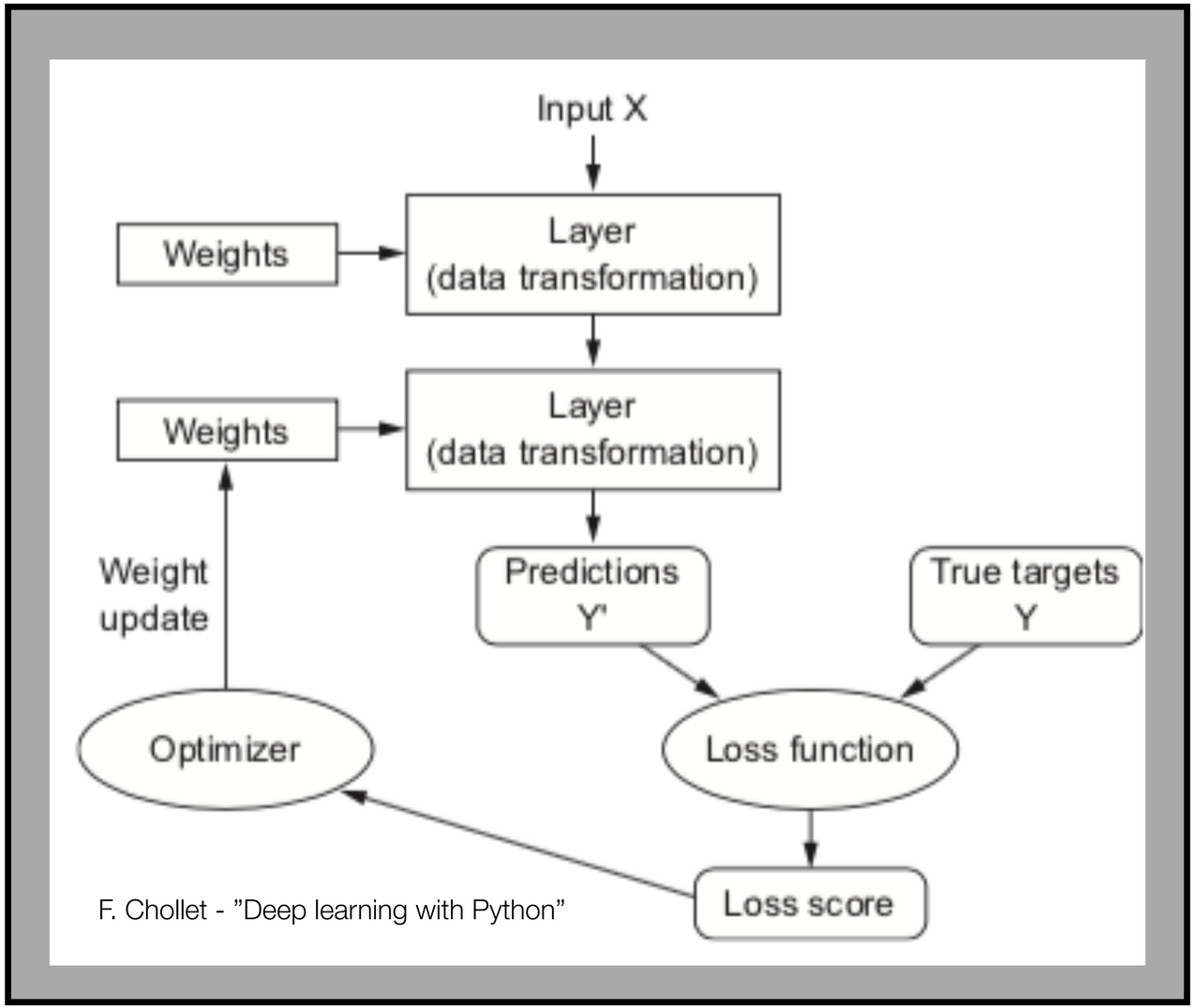
Data generation

- Tracking in PLACET, beam-beam in GUINEA-PIG
- Simulate perfect machine with sextupole transverse offsets (5, 10, 20 μm rms)
- 1 run = 10 random cases (less then 20 mins) 10,000 jobs at the time
- Generated about 450,000 data points

Machine Learning:

- Deep learning with artificial neural networks
- TensorFlow and Python library Keras

Model training

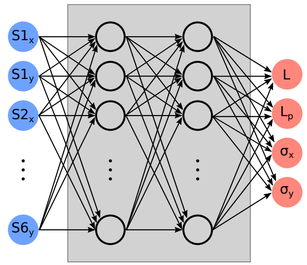


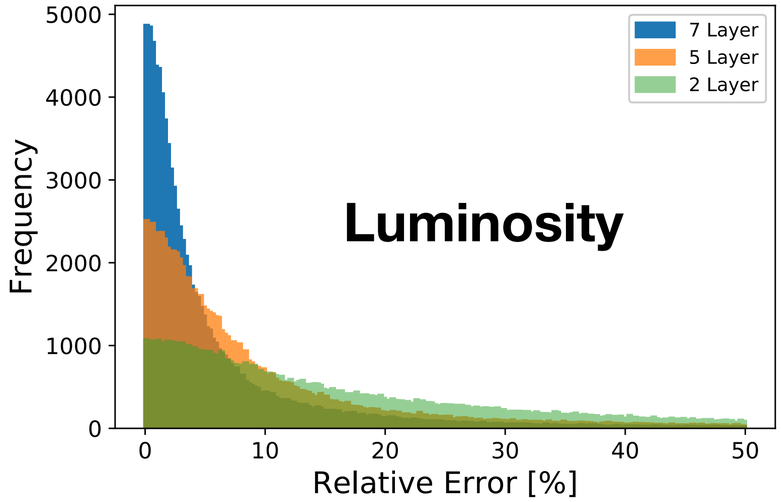
- Adjust weights and biases for each node to minimize the loss function
- Loss function: e.g. mean square error
- Algorithm: *backpropagation of errors*
  (gradient descent) adjust weights to minimize loss
- Implemented in TensorFlow
- Python Keras library to interface

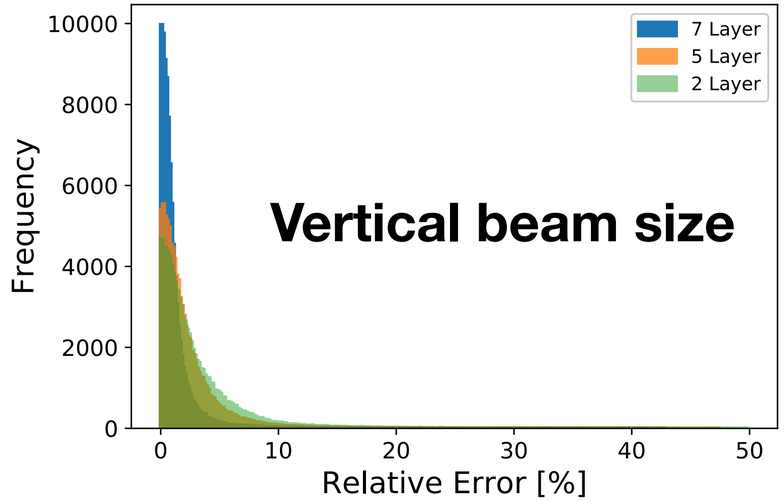
- Split data: training (80%), testing (20%)
- Testing data only used for model evaluation

Model performance

	Luminosity			Vertical beam size		
	2 Layers	5 Layers	7 Layers	2 Layers	5 Layers	7 Layers
Mean(|Rel_error|) [%]	29.1	11.7	6.5	3.0	2.3	1.2
Std(|Rel_error|) [%]	62.4	20.1	11.1	3.5	2.8	2.2
90% less than [%]	64.7	27.4	15.8	6.7	5.1	2.6



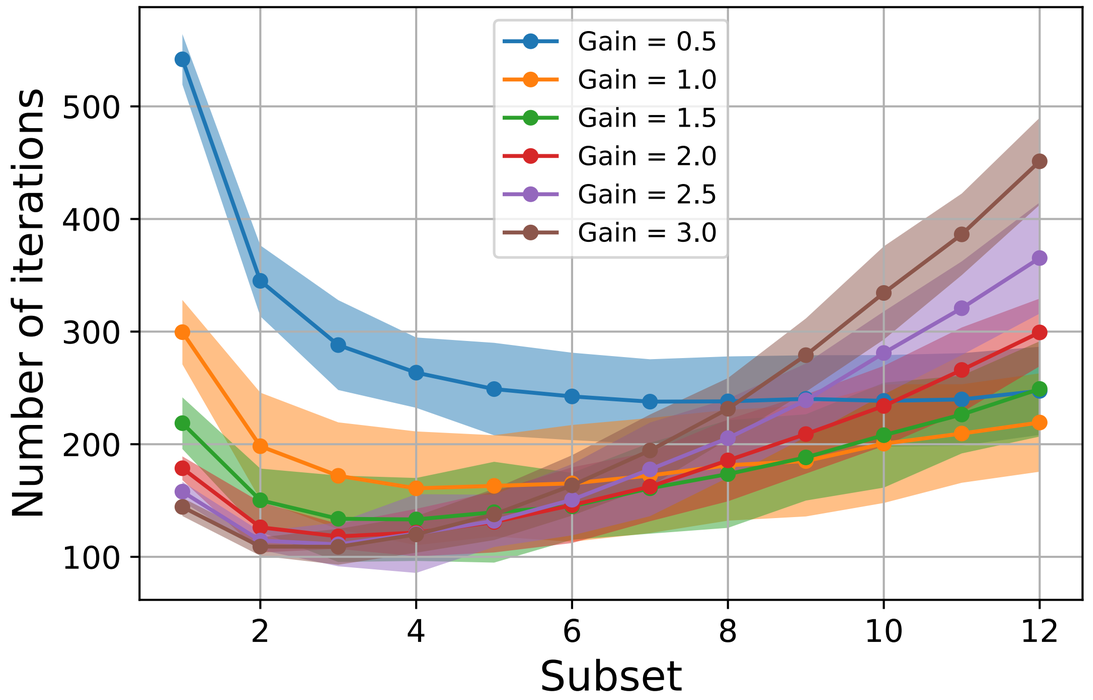




- It seems difficult to get the mean error below 5-6%
- A model in a more narrow range performs better
- Part of it comes from Luminosity uncertainty (~1%)
- Accuracy is not the most crucial aspect
- A model that correctly characterizes the behavior is very useful

Random walk algorithm

- ML model: 1000 iteration random walk tuning takes a few seconds
- Full-scale simulation: 1000 iterations random walk tuning takes ~8h
- Use ML model to optimize algorithm
- For each setting: 100 different random seeds, each tuned 100 times
- Random walk:
  - Select a subset out of the 12 DOF
  - Make steps in random direction:
    gain[-1, -0.5, 0.5, 1]*

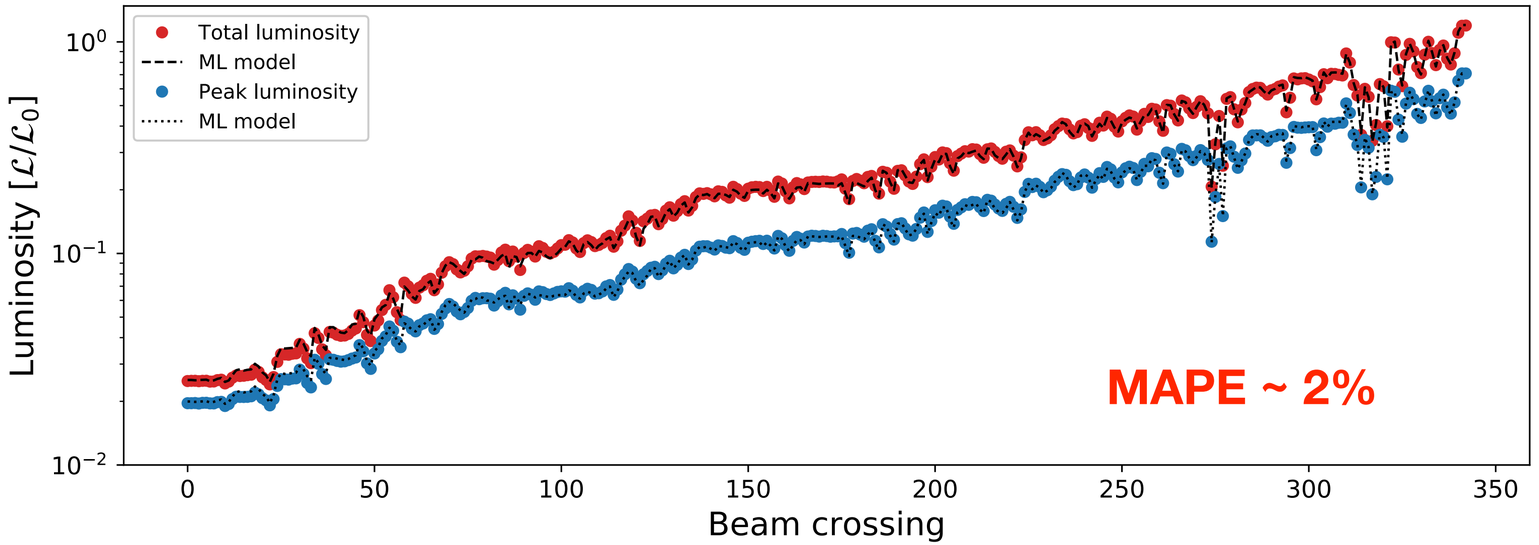


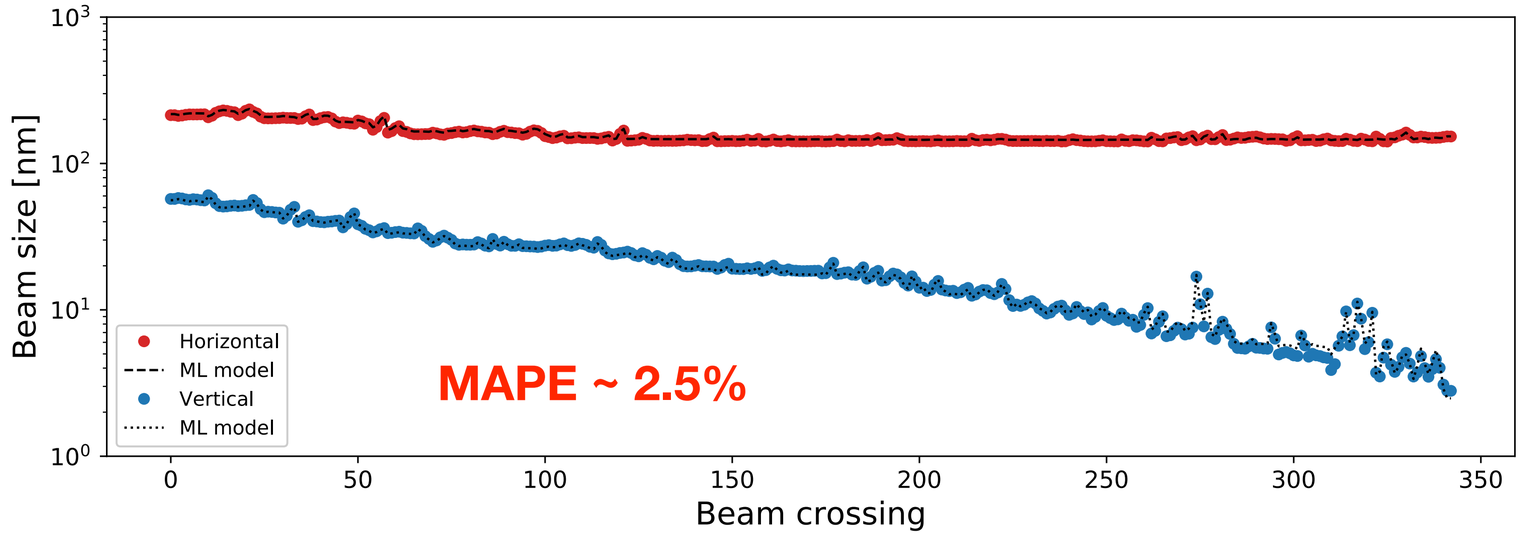
Compare with full simulation

Simulation

- Tuning of the perfect machine with sextupole transverse offsets only
- Use the normal tuning knobs and full-scale tracking (100,000 macroparticles)
- At each step: evaluate luminosity from ML model as well and save to file
- To evaluate predictive performance, mean absolute percentage error (MAPE):

$$\frac{1}{N}\sum_{i=1}^{N}\left|\frac{y_i-\tilde{y}_i}{y_i}\right|$$





Conclusions

- Compact Linear Collider
  - A proposed electron—positron linear collider at CERN
  - Nanometer beam sizes at IP
- Final-focus system
  - Highly nonlinear system
  - Imperfections have huge impact on luminosity
- Tuning studies
  - Reach nominal luminosity under imperfect condition
  - Monte Carlo study with static imperfections
  - BBA, random walk, sextupole knobs, combined random walk
  - Used beamstrahlung and incoherent pairs as tuning signals
  - Robust method with high success rate
- Machine Learning
  - Surrogate model for sextupoles in the final-focus system
  - Cross-checked with full simulation
  - Also tried extending model to include quadrupole offsets

Backup slides

Incoherent pairs

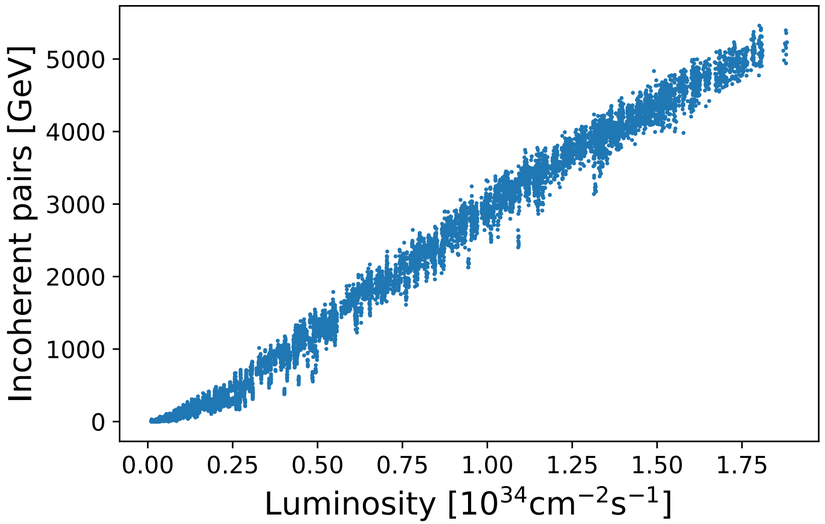
Three mechanisms for incoherent e^-e^+ production:

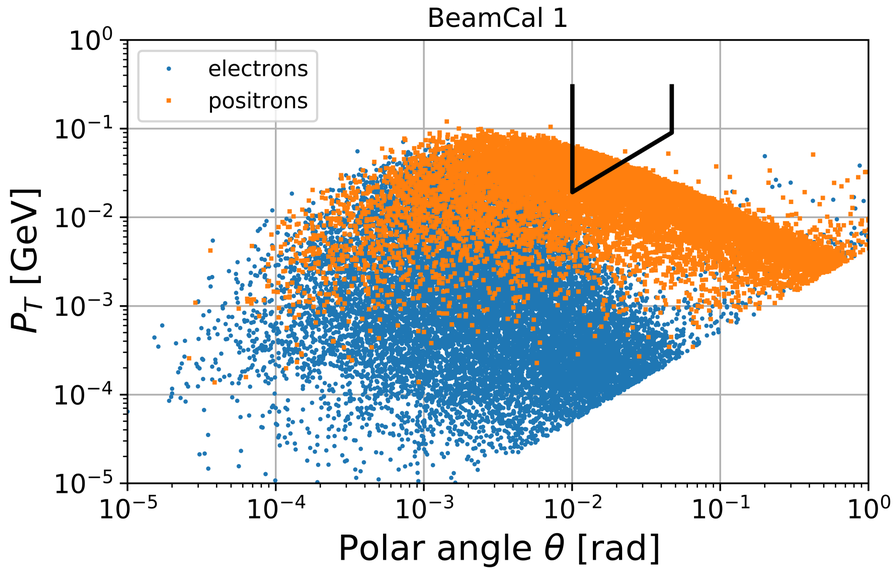
- **Breit—Wheeler:** $\gamma + \gamma \rightarrow e^- + e^+$ two real photons
- **Bethe—Heitler:** $\gamma + e^\pm \rightarrow e^\pm + e^- + e^+$ one real and one virtual photon
- **Landau—Lifshitz:** $e + e \rightarrow e + e + e^- + e^+$ two virtual photons

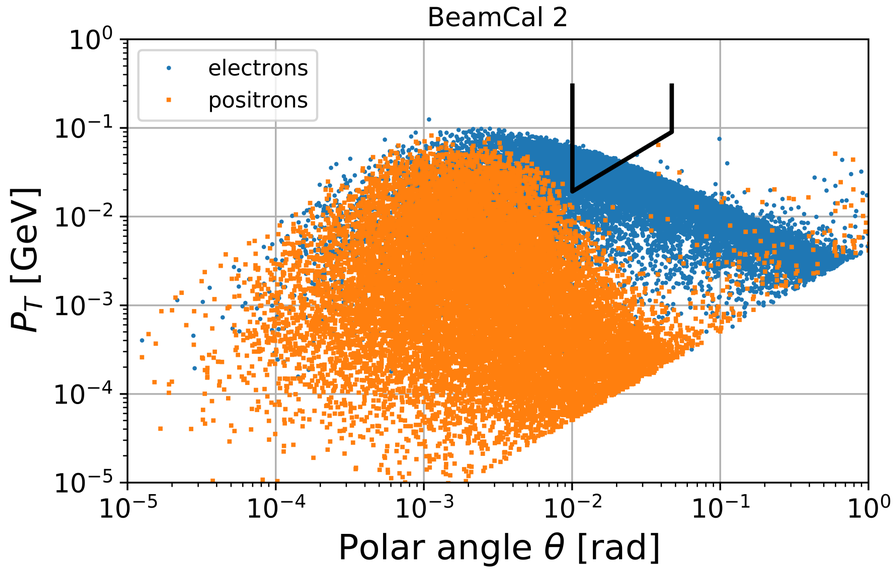
Incoherent pairs in BeamCal

- Compute and track pairs in GUINEA-PIG
  - deflection from EM fields of beam
  - Computationally intensive
- Make cut in p_T—θ plane
  - Particles with correct angles and high enough energy
  - Total energy of ~6000 GeV/bunch-crossing

- Luminosity vs. inc. pairs
  - 1000 cases, 10 simulation per case
  - Linear correlation but noisy

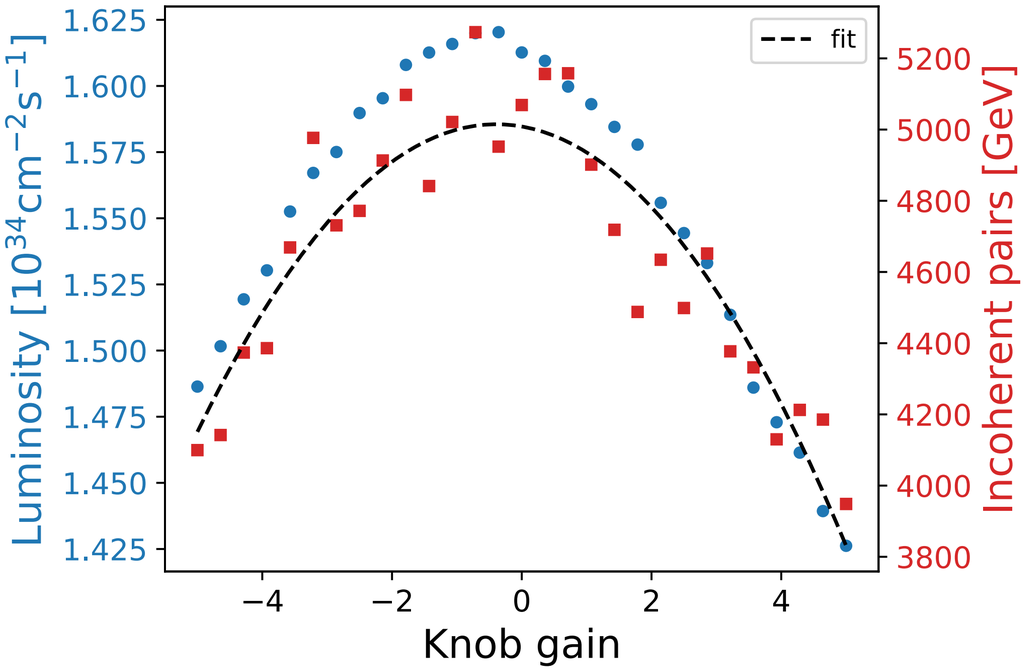


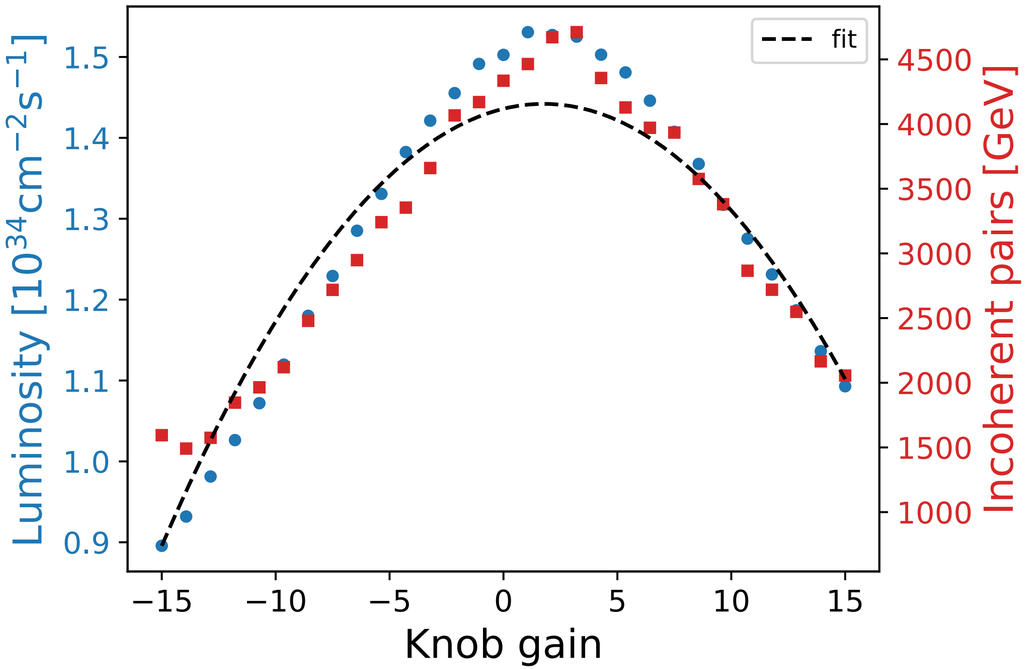




Dealing with noise

Example scan of sextupole knobs

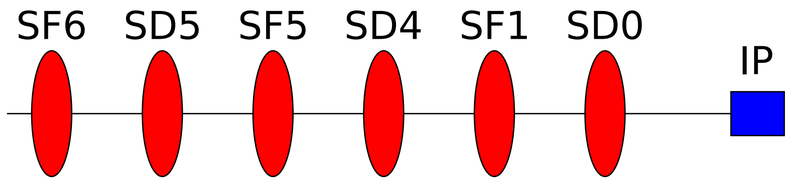




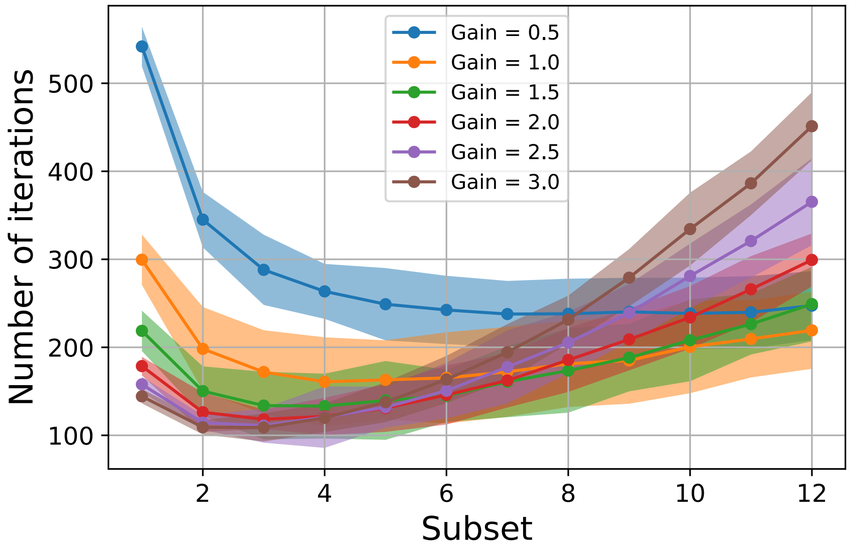
- Use mainy points (29 in this case) and make single parabolic fit
  - Previously parabolic minimizer on luminosity signal
  - Computationally beneficial: generate the points in parallel
- Scan over a sufficient range
- Seems robust

Random walk

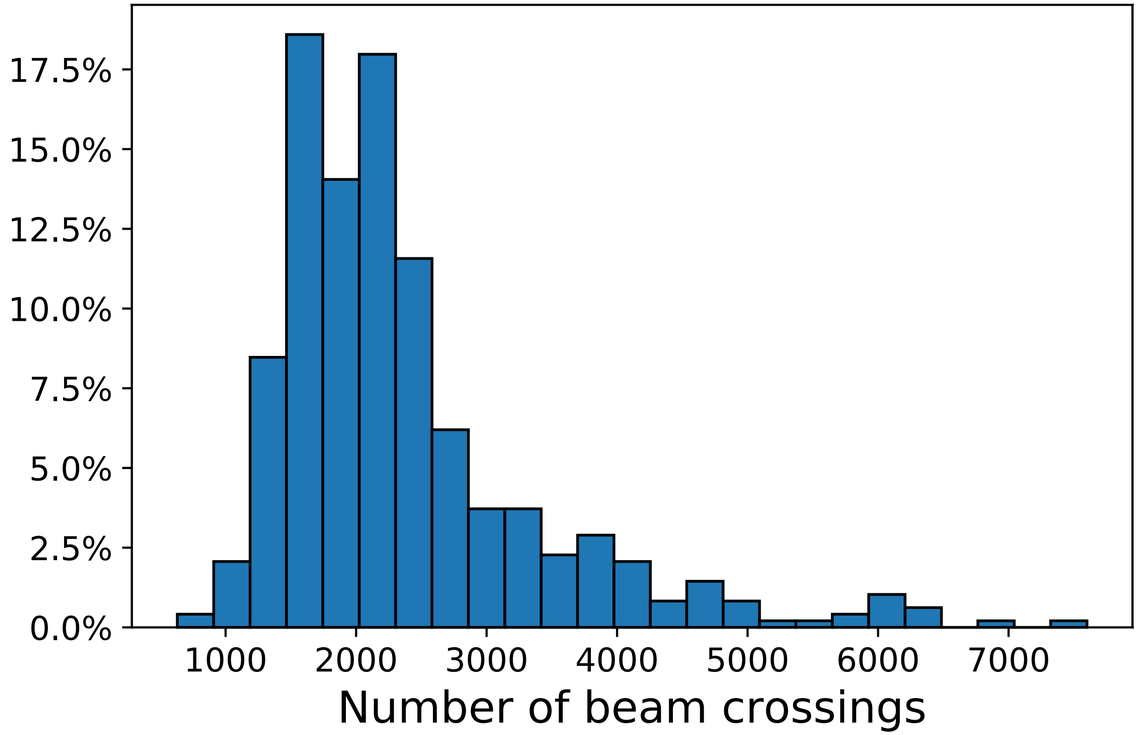
Random walk algorithm for sextupole transverse position



- Randomly select which beamline to tune
- Select a random subset (e.g. 6 out of 12 DOF)
- Select a random direction for that subset
- Perform a short scan: 7 points, single parabolic fit (parallel execution)
- Select point that optimizes signal
- Iterate



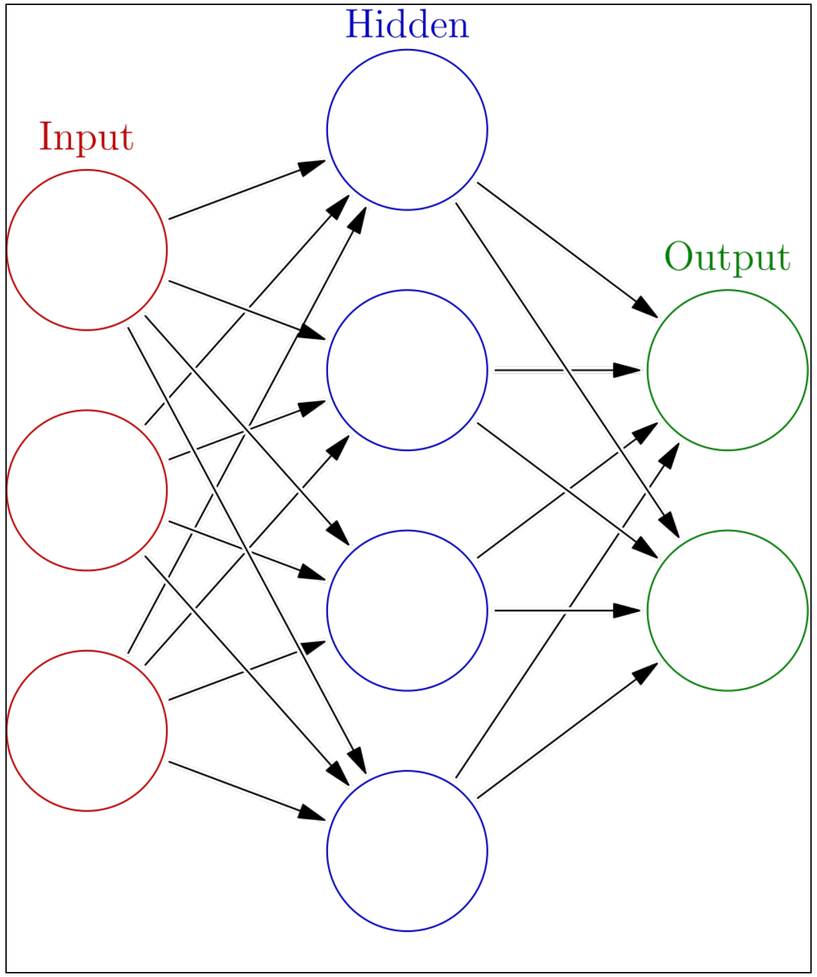
Tuning time



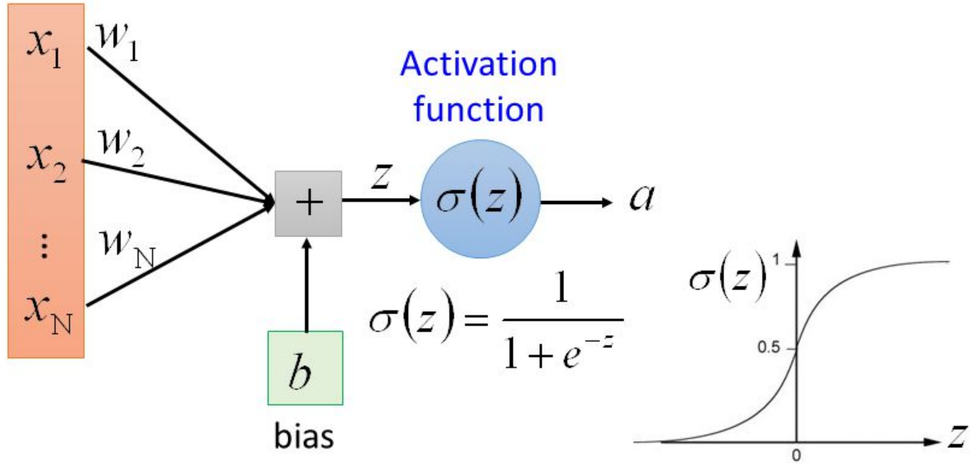
N_{min}	630
N_{max}	7601
N_{mean}	2371
N_{median}	2104

- Median number of iterations = 2104
- Close to 2x900 (one-beam tuning)
- Room for improvement

Artificial neural networks



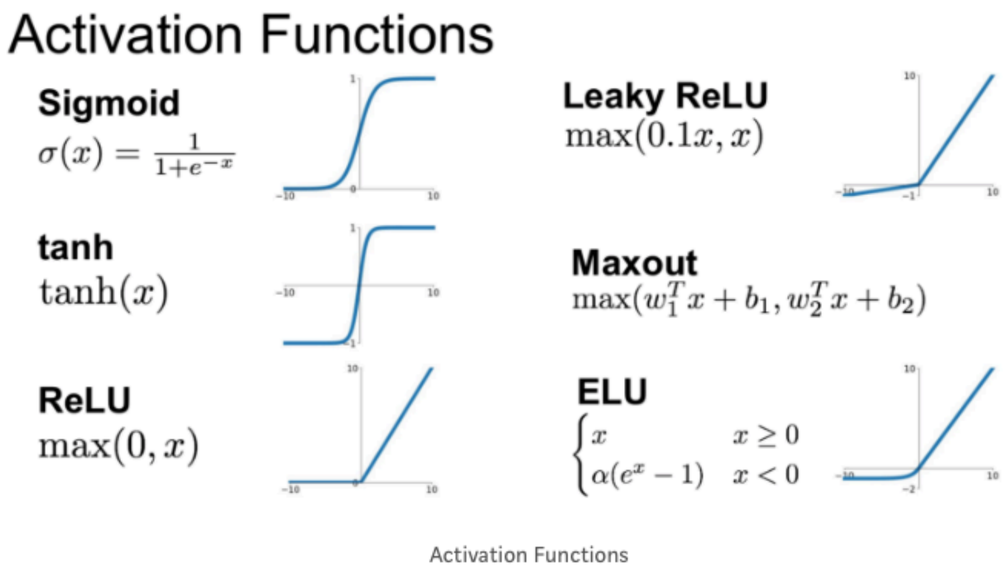
Single Neuron



z = Inputs to neuron scaled with weights plus a bias

output = activation function(z)

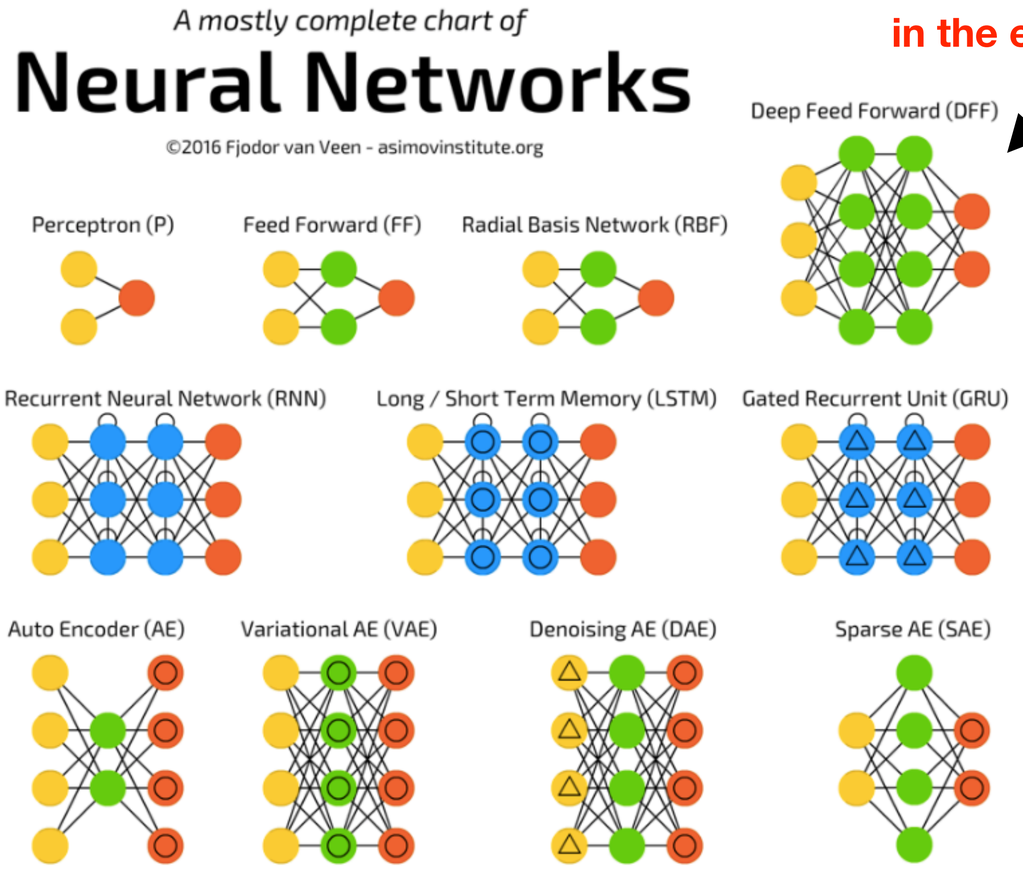
Design of Neural Network



What activation function?

Network architecture?

What we will use in the example



and many more…

TensorFlow and Keras

- TensorFlow
  - Open-source library for machine learning application. Developed by Google.

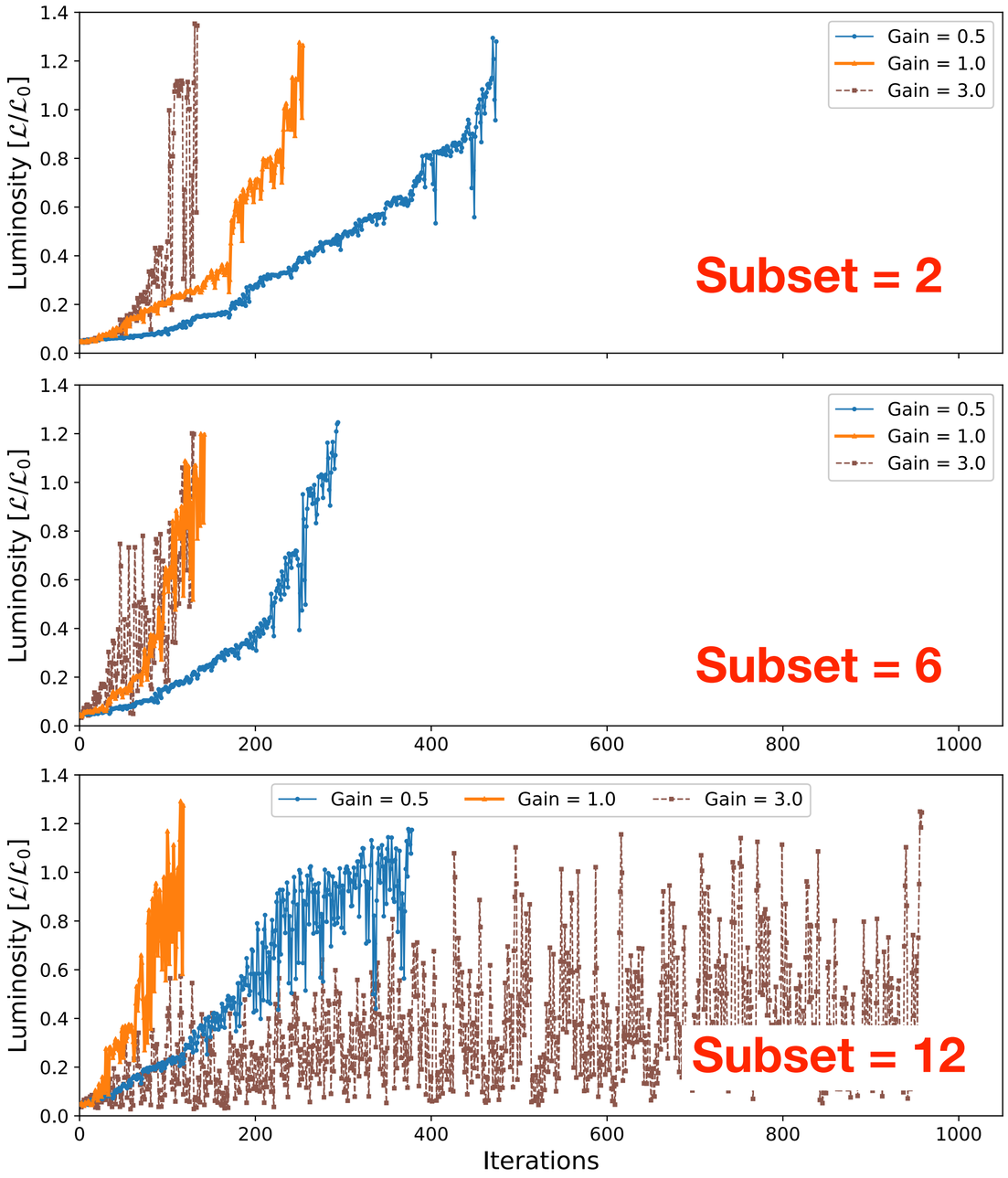
- Keras
  - Open-source, user-friendly Python library
  - Can interface TensorFlow (also other ML libraries)

- Good resources
  - Stanford courses on ML
  - F. Chollet - *”Deep Learning with Python”*

"[...] neural networks consist entirely of chains of tensor operations and that all of these tensor operations are just geometric transformations of the input data. It follows that you can interpret a neural network as a very complex geometric transformation in a high-dimensional space, implemented via a long series of simple steps”

from *Deep Learning with Python*

Full simulation



- Test random walk parameters on full simulation
- A few example