Simulation of Beam Losses of Colliding Beams at the Large Hadron Collider

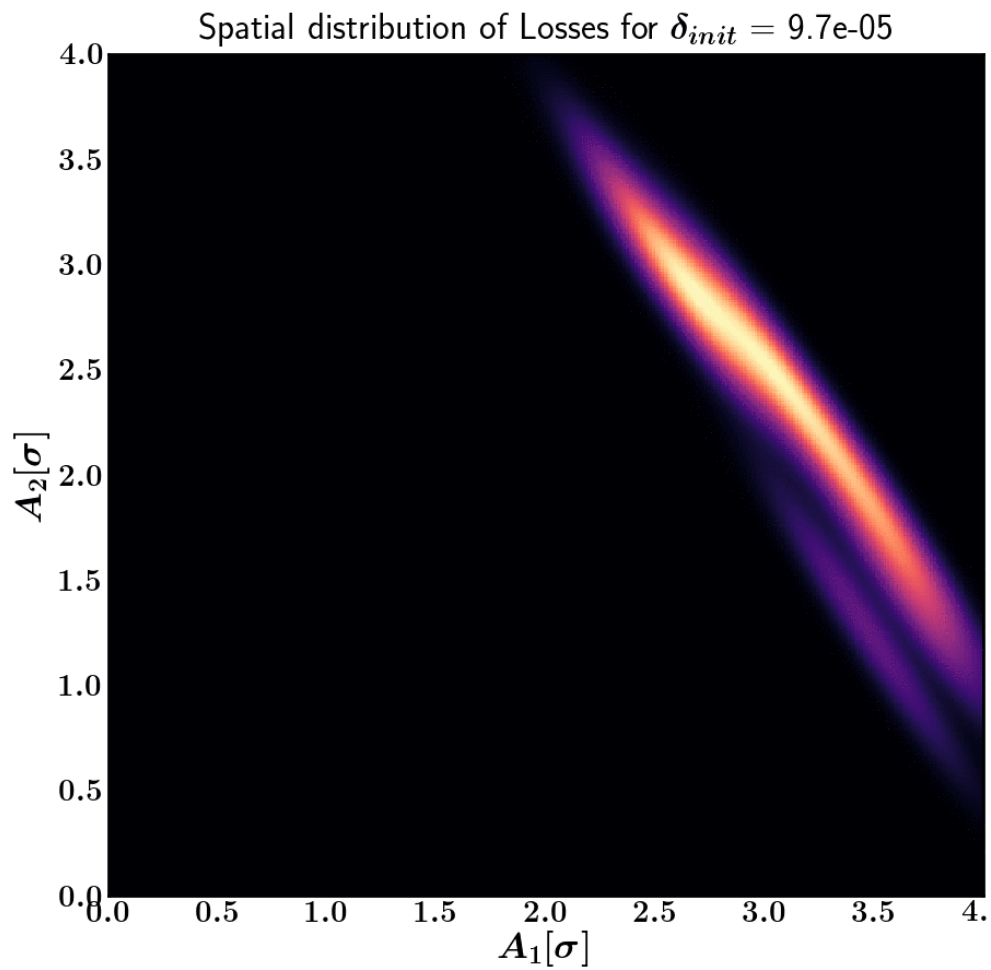
Julie Malewicz

Supervisors:
Giovanni Iadarola and Konstantinos Paraschou

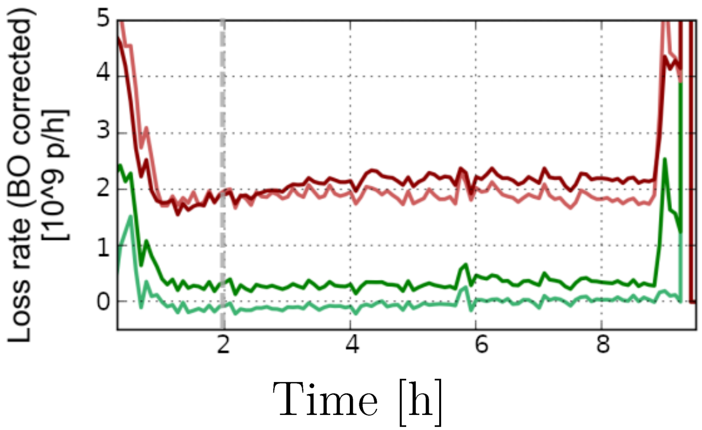
Special thanks to Riccardo De Maria and Martin Schwinzerl

Table of Contents

- Introduction
- Defining Losses
  - Monte Carlo
- Simulation of Beam Losses
  - Initialization
  - Variance-reducing techniques
  - Results
- Conclusions
- References



Motivations



- Intensity loss observed in the LHC cannot be fully explained by standard luminosity decay
  - Instead, we realize that non-linear effects, such as **beam-beam interactions**, cannot be neglected and play a major part

- With a new tracking software such as **Sixtracklib**, the purpose will be to find how can we most effectively model, compute, visualize and finally draw conclusions about the observed luminosity losses in the LHC?

- *Goal: Simulate 30 minutes of beam (or 20 million turns), and compare results to actual observed losses of about 0.1%*

Tools

- **GPUs**
  - **(4)** in CNAF Bologna (shared among 3 people)
  - **(12)** in HTCondor (that other people are also in competition for)
  - 10,000 particles need 4.5 days on 1 GPU for 20,000,000 turns in the full LHC lattice

- **Sixtracklib**
  - simple Python interface
  - supports GPU parallelization

- To use the barebones Sixtracklib software, we still had to reinvent some wheels, such as normalizing the fully linearly coupled motion.
- We also had to work out how to define the beam distribution in the longitudinal plane that accounts for non-linear synchrotron motion

Source: https://github.com/kparasch/sixtracklib_tracking_lhc

Quantifying Losses

In order to visualize losses in the accelerator, we use **relative intensity** as proxy:

$$I(t) = \sum_{particle\ i} K(t, t_0^{(i)})$$

where $K(t,\ t_0^{(i)}) = \begin{cases} 1 & \text{if} \quad t < t_0 \\ 0 & \text{if} \quad t \geq t_0 \end{cases}$ and t_0 is the time at which the particle is lost (found using Sixtracklib tracking)

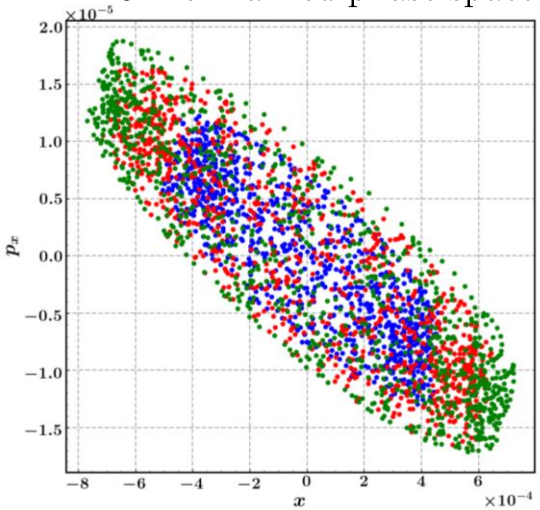
- However, there are about 10^{11} particles in the bunch – it would be impossible to simulate every single one of them.
- We therefore assume they follow a distribution $\rho(x,\ p_x,\ y,\ p_y,\ \zeta,\ \delta)$, and our sum becomes the following integral:

$$I(t) = \int K\Big(t, t_0(x, p_x, y, p_y, \zeta, \delta)\Big)\rho(x, p_x, y, p_y, \zeta, \delta)\ dx\ dp_x\ dy\ dp_y\ d\zeta\ d\delta$$

Quantifying Losses: *normalization*

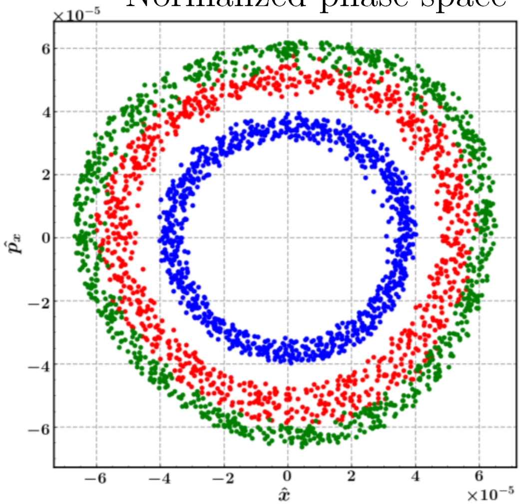
Due to couplings, such as dispersion, we must apply **normalization of fully coupled motion** to our coordinates:

Unnormalized phase-space



$$\begin{pmatrix} \hat{x} \\ \hat{p_x} \\ \hat{y} \\ \hat{p_y} \\ \hat{\zeta} \\ \hat{\delta} \end{pmatrix} = W^{-1} \left[\begin{pmatrix} x \\ p_x \\ y \\ p_y \\ \zeta \\ \delta \end{pmatrix} - \begin{pmatrix} x_{CO} \\ p_{x_{CO}} \\ y_{CO} \\ p_{y_{CO}} \\ \zeta_{CO} \\ \delta_{CO} \end{pmatrix} \right]$$

Normalized phase-space



and the integral transforms to:

$$\int K\big(t, t_0(\hat{x},\, \hat{p_x},\, \hat{y},\, \hat{p_y},\, \hat{\zeta},\, \hat{\delta})\big) \cdot \rho(\hat{x},\, \hat{p_x},\, \hat{y},\, \hat{p_y},\, \hat{\zeta},\, \hat{\delta})\; d\hat{x}\, d\hat{p_x}\, d\hat{y}\, d\hat{p_y}\, d\hat{\zeta}\, d\hat{\delta}$$

Quantifying Losses: *factorization*

We assume ρ can be factorized as $g(\hat{x},\ \hat{p_x},\ \hat{y},\ \hat{p_y}) \cdot h(\hat{\zeta},\ \hat{\delta})$ where:

$$g(\hat{x},\ \hat{p_x},\ \hat{y},\ \hat{p_y}) \sim e^{-\frac{\hat{x}^2+\hat{p_x}^2}{2\epsilon_1}} \cdot e^{-\frac{\hat{y}^2+\hat{p_y}^2}{2\epsilon_2}}$$

describes the *transverse* distribution

- Defining our coordinates to be within $n \times \sigma$, where σ is the standard deviation of our Gaussian distribution g, means that:

$$\left(\frac{\hat{x}}{\sqrt{\epsilon_1}}\right)^2 + \left(\frac{\hat{p_x}}{\sqrt{\epsilon_1}}\right)^2 + \left(\frac{\hat{y}}{\sqrt{\epsilon_2}}\right)^2 + \left(\frac{\hat{p_y}}{\sqrt{\epsilon_2}}\right)^2 \leq n^2$$

Quantifying Losses: *factorization*

We assume ρ can be factorized as $g(\hat{x},\ \hat{p_x},\ \hat{y},\ \hat{p_y}) \cdot h(\hat{\zeta},\ \hat{\delta})$ where:

$$h(\hat{\zeta},\ \hat{\delta})\ d\hat{\zeta}\ d\hat{\delta} \sim e^{-\frac{J_3}{\epsilon_3}}\ dJ_3\ d\phi_3$$

accounts for the *longitudinal* plane,

The longitudinal plane is simpler in that it is **barely affected by the transverse plane** – yet, more complicated, as we have to take into account the non-linearity of the force.

- where we define J_3 as follows: $J_3 = \frac{1}{2}\left(\frac{4qV_{RF}}{\omega_{RF} P_0 C_0} \cdot \sin^2 \frac{\omega_{RF}\zeta}{2c} + \eta\delta^2\right)$
- and ϵ_3 is chosen so as to match the bunch length such that $\langle \zeta^2 \rangle \ = (0.075\text{m})^2$

Quantifying Losses

Now,

$$I(t) = \int K\big(t, t_0(\hat{x}, \hat{p_x}, \hat{y}, \hat{p_y}, J_3,\ \phi_3)\big) \cdot g(\hat{x}, \hat{p_x}, \hat{y}, \hat{p_y})\ h(J_3)\ d\hat{x}\ d\hat{p_x}\ d\hat{y}\ d\hat{p_y}\ dJ_3\ d\phi_3$$

And furthermore, we assume that t_0 is independent of the longitudinal angle ϕ_3, further simplifying:

$$I(t) = \int K\big(t, t_0(\hat{x},\ \hat{p_x},\ \hat{y},\ \hat{p_y},\ J_3)\big) \cdot g(\hat{x},\ \hat{p_x},\ \hat{y},\ \hat{p_y})\ h(J_3)\ d\hat{x}\ d\hat{p_x}\ d\hat{y}\ d\hat{p_y}\ dJ_3$$

$$= \int F(t, J_3) \cdot h(J_3)\ dJ_3$$

where:

$$\boxed{F(t,\ J_3) = \int K\big(t, t_0(\hat{x},\ \hat{p_x},\ \hat{y},\ \hat{p_y},\ J_3)\big) \cdot g(\hat{x},\ \hat{p_x},\ \hat{y},\ \hat{p_y})\ d\hat{x}\ d\hat{p_x}\ d\hat{y}\ d\hat{p_y}}$$

Quantifying Losses

$$\boxed{F(t,\ J_3) = \int K\big(t, t_0(\hat{x},\ \hat{p_x},\ \hat{y},\ \hat{p_y},\ J_3)\big) \cdot g(\hat{x},\ \hat{p_x},\ \hat{y},\ \hat{p_y})\ d\hat{x}\ d\hat{p_x}\ d\hat{y}\ d\hat{p_y}}$$

This integral becomes impossible to compute analytically – therefore motivating our shift to a **Monte Carlo numerical integration**.

Monte Carlo integration is a widely used and proven method for multi-dimensional integrals

Its advantages?

- The possibility to always add more data points without worrying about where to generate them
- The ability to easily quantify the error on integration

Monte Carlo Integration

- Allows for fast and efficient numerical computation of multi-dimensional definite integrals, of the form:

$$\int_{\Omega} F(\vec{x})\, d\vec{x}, \text{ where } \Omega \text{ has volume } V = \int_{\Omega}\, d\vec{x}$$

- Defined as representing the solution to a problem as a parameter of a hypothetical random distribution, from which **statistical estimates** can be obtained.

- The law of large number ensures that by increasing our sample size M:

$$\lim_{M \mapsto \infty} \frac{1}{M} \sum_{i}^{M} F(\vec{x_i}) \longmapsto \frac{1}{V} \int_{\Omega} F(\vec{x})\, d\vec{x}$$

Monte Carlo Integration: *errors*

- The standard deviation (**error**) of the Monte Carlo estimate is

$$\pm\sqrt{\frac{\mathrm{Var(F)}}{M}}$$

- where M is the number of samples used in the integration

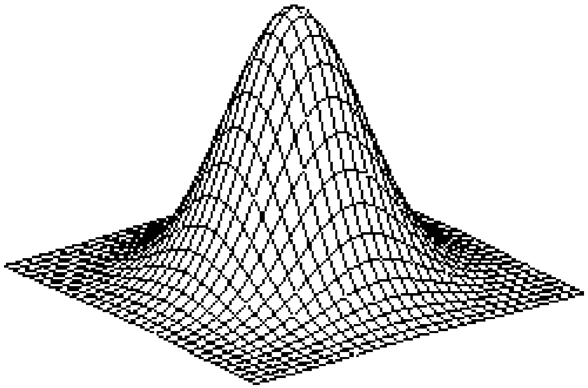
- The variance can be reduced by increasing M, though after a certain threshold of precision, the improvement clearly becomes inefficient computationally.

Simulation parameters

Bunch Intensity :	1.25×10^{11} protons
Energy :	6,500 GeV
Octupole magnets' current :	550 A
Bunch separation :	25 ns
Transverse normalized emittance :	2×10^{-6} m / 2×10^{-6} m
RMS bunch length :	0.075 m
Half crossing angle :	160 μrad
Betatron tunes :	62.31 / 60.32
β at IP1 and IP5:	30 cm

We use a typical *mask* file as used for Sixtrack simulations to generate the thin lattice. The beam-beam lenses are setup independently. 6D modelling is used for head-on interactions and 4D modelling for the long range interaction

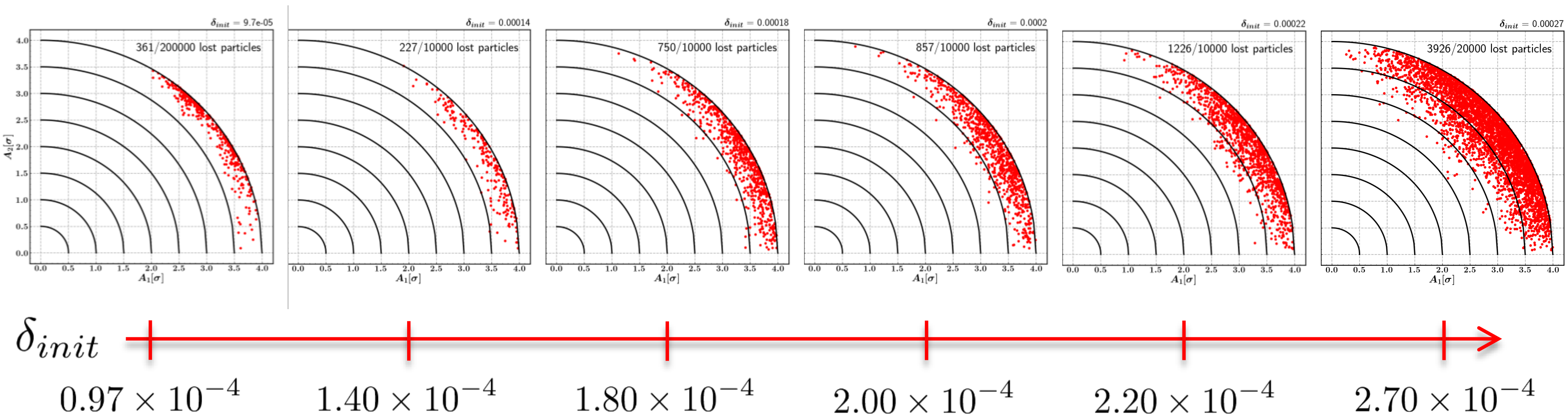
Generating the initial beam distribution



- Transverse distribution in Ω:
  - Our input into Sixtracklib was decided to be a **hypersphere of radius 4σ** in the normalized phase-space, populated **uniformly** and **randomly**
  - It will be constructed with particles of same initial momentum deviation δ
- Longitudinal distribution in Λ :
  - We repeat the above steps for different δ, so as to be able to get enough datapoints, to compute the final integral:

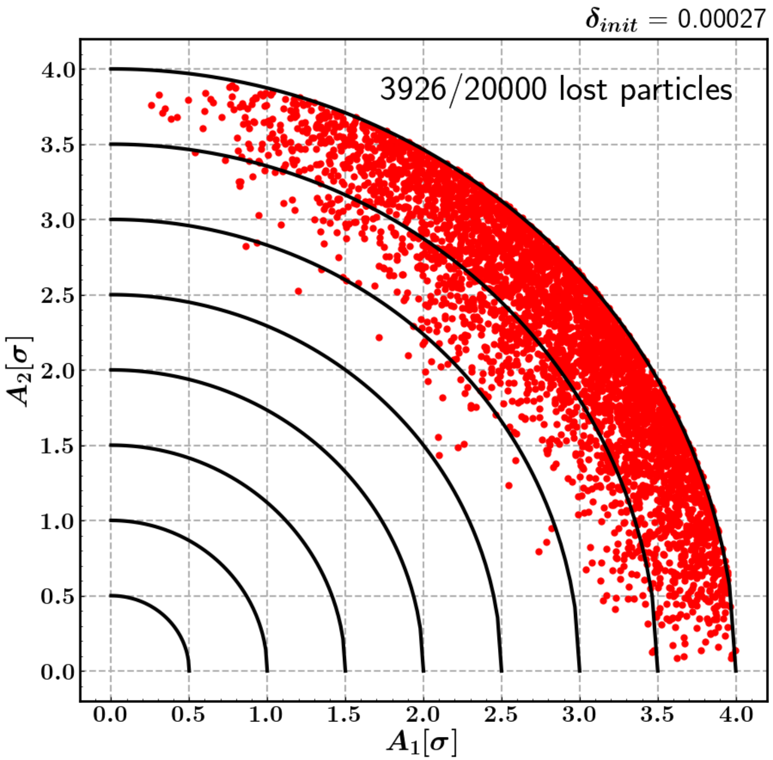
$$I(t) = \int_{\Lambda} F(t,\ J_3)\ h(J_3)\ dJ_3$$

Transverse phase-space localization of losses



As can be expected, the larger the initial momentum deviation from the closed orbit, the more significant the losses

Transverse phase-space localization of losses



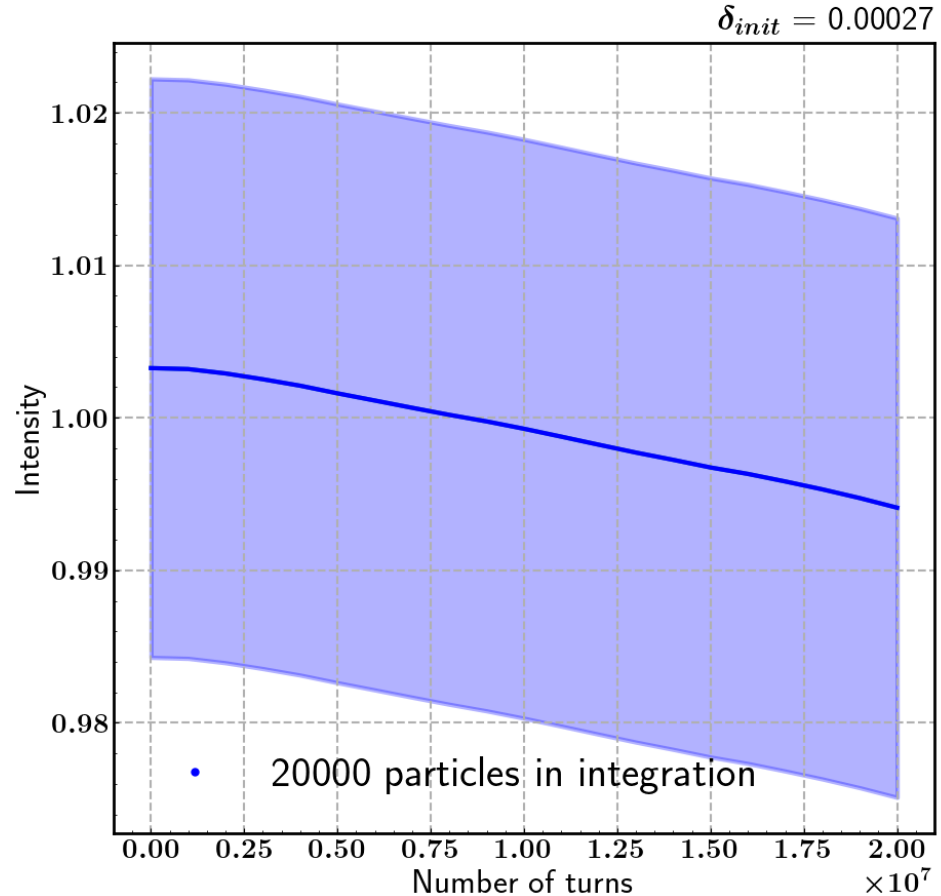
$$\delta_{init} = 2.70 \times 10^{-4}$$

We will now focus our attention onto the dataset with largest momentum deviation

We started by performing a crude Monte Carlo integration, over the totality of our 20,000 particles:

$$F(t,\ J_3) = \int_\Omega K(t,\ t_0) \cdot g\ d\hat{x}\ d\hat{p}_x\ d\hat{y}\ d\hat{p}_y$$

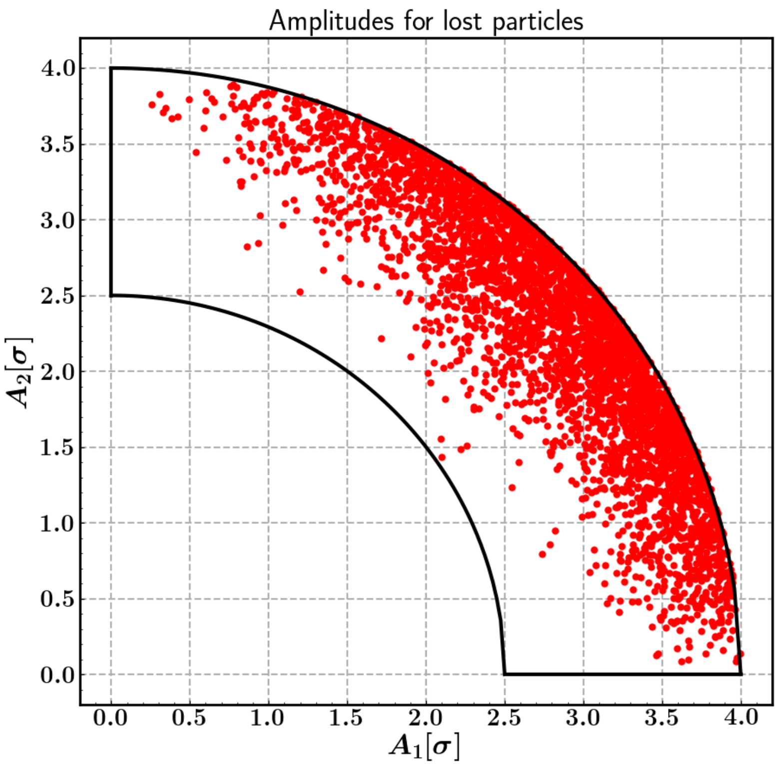
$$\Omega\ = [0,\ 4\sigma]$$



Step 1:

Crude Monte Carlo

How can the variance be reduced?



$$\mathrm{Var}(F) = \frac{\mathrm{Var}(K \cdot g)}{M}$$

As mentioned earlier, one way to reduce the error interval on the intensity is to increase our sample size.

However, it is often preferable to intervene directly on the source of the uncertainty: F.

There is no need to accumulate errors by using Monte Carlo integration over regions where no losses were recorded.

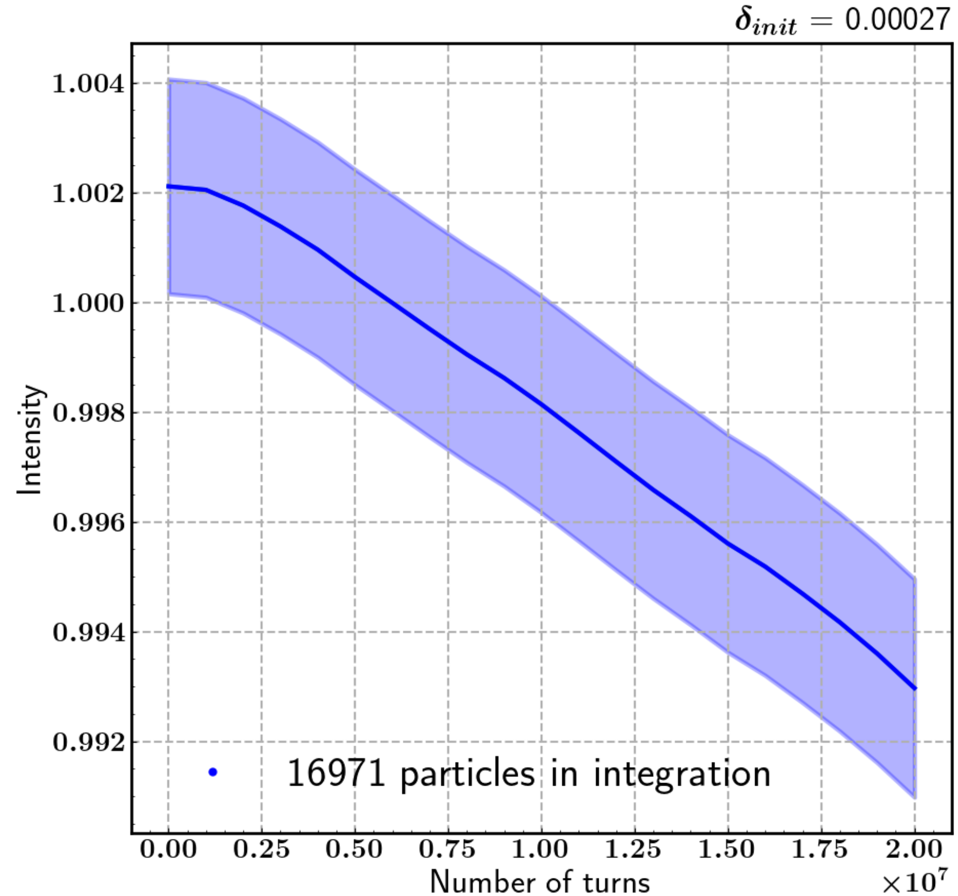
Reduced Sampling

This time, we only integrate in the subregion where losses were recorded, and calculate the inner shell $F(t,\ J_3)\Big|_{\Omega_0}$ analytically

$$F(t,\ J_3) = F(t,\ J_3)\Big|_{\Omega_0} + \int_{\Omega_1} K(t,\ t_0) \cdot g\ d\hat{x}\ d\hat{p_x}\ d\hat{y}\ d\hat{p_y}$$

$$\Omega_0 = \left[0,\ 2.5\sigma\right]$$
$$\Omega_1 = \left[2.5\sigma,\ 4\sigma\right]$$

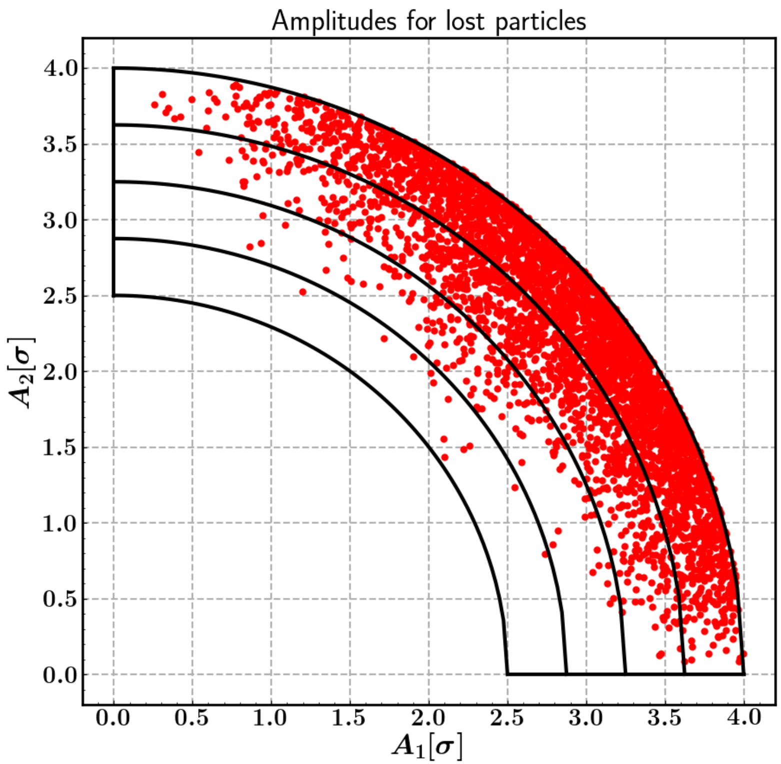
$$F(t,\ J_3)\Big|_{\Omega_0} = \int 1\ \cdot\ g\ d\hat{x}\ d\hat{p_x}\ d\hat{y}\ d\hat{p_y}$$



Step 2:

Reduced sampling

How can the variance be reduced?



Additionally, given that our density function g depends exclusively on J_1 and J_2, points in close proximity in phase space will have similar values for $K \cdot g$: treating them region by region should therefore reduce the variance significantly.

$$F(t,\ J_3) = F(t,\ J_3)\Big|_{\Omega_0} + \int_{\Omega_{1'}} K(t,\ t_0) \cdot g\ d\hat{x}\ d\hat{p}_x\ d\hat{y}\ d\hat{p}_y + \int_{\Omega_{2'}} K(t,\ t_0) \cdot g\ d\hat{x}\ d\hat{p}_x\ d\hat{y}\ d\hat{p}_y + \ldots$$

This commonly known variance-reducing technique is known as:

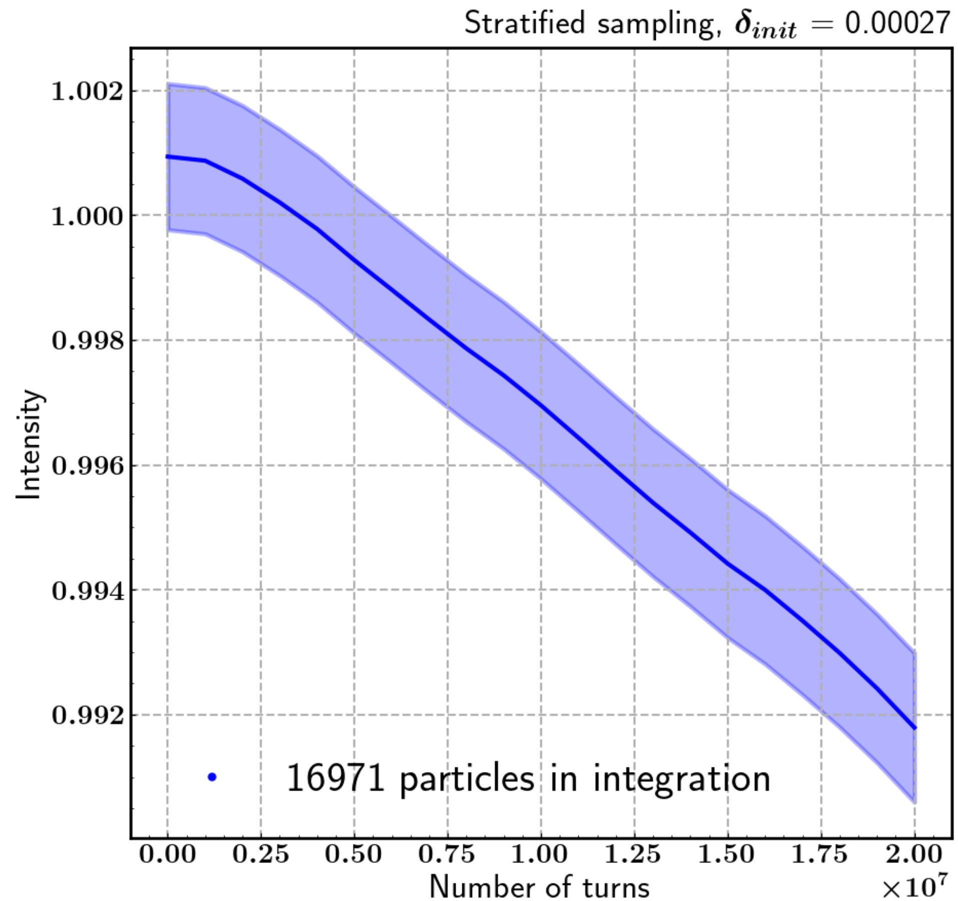
$\boxed{\text{Stratified sampling}}$

Source: https://en.wikipedia.org/wiki/Stratified_sampling

Stratified sampling allows us to leave the integrand untouched, while reducing the variance to the sum of that of each subregion:

$$\begin{aligned} F(t,\ J_3) = F(t,\ J_3)\Big|_{\Omega_0} & \\ &+ \int_{\Omega_{1'}} K(t,\ t_0)\cdot g\ d\hat{x}\ d\hat{p_x}\ d\hat{y}\ d\hat{p_y} \\ &+ \int_{\Omega_{2'}} K(t,\ t_0)\cdot g\ d\hat{x}\ d\hat{p_x}\ d\hat{y}\ d\hat{p_y} \\ &+ \ldots \end{aligned}$$

$$\Omega_0 = \left[0,\ 2.5\sigma\right]$$
$$\Omega_{1',2',\ldots} = \left[2.5\sigma,\ 2.8\sigma\right], \left[2.8\sigma,\ 3.1\sigma\right], \ldots$$



Step 3:
Stratified sampling

How can the variance be reduced?

- We can
- Why count the remaining particles when we could instead completely put them aside and focus on the lost ones?
- In short, we define:

$$F(t,\ J_3) = \int \Big(K(t,\ t_0) \cdot g \Big)\ d\hat{x}\ d\hat{p_x}\ d\hat{y}\ d\hat{p_y}$$

take the derivative

$$\frac{\partial}{\partial t}\Big(F(t,\ J_3) \Big) = -\int \Big(\delta(t - t_0) \cdot g \Big)\ d\hat{x}\ d\hat{p_x}\ d\hat{y}\ d\hat{p_y}$$

integrate once more

$$\int_0^t \frac{\partial F}{\partial t'} dt' = -\int_0^t \int_\Omega \Big(\delta(t - t_0) \cdot g \Big)\ d\hat{x}\ d\hat{p_x}\ d\hat{y}\ d\hat{p_y}\ dt$$

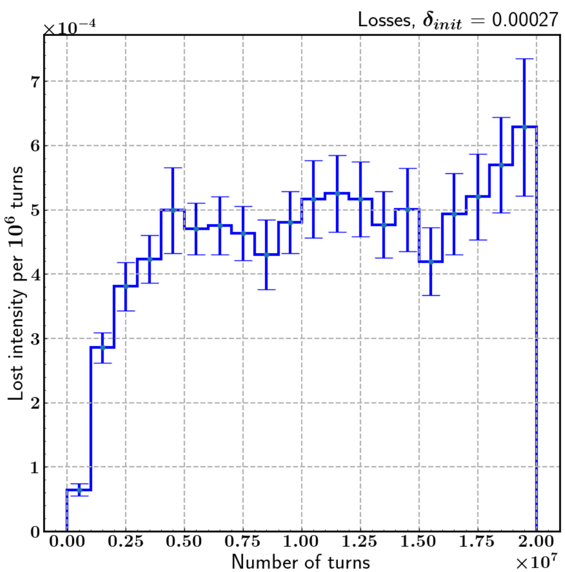
split time integral

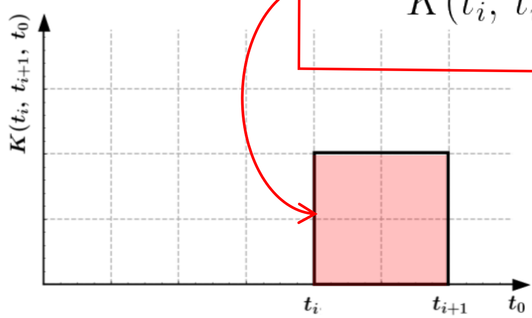
$$F(t,\ J_3) = 1 - \sum_i^{t/\Delta t - 1} \int_\Omega \int_{t_i}^{t_{i+1}} \delta(t - t_0) \cdot g(\hat{x},\ \hat{p_x},\ \hat{y},\ \hat{p_y})\ dt\ d\hat{x}\ d\hat{p_x}\ d\hat{y}\ d\hat{p_y}\ dt$$

The Losses Integral

- This algebraic manipulation gives us a much **smaller variance** on our Monte Carlo estimate and provides **direct access to losses** as a function of time.

$$\boxed{F(t,\ J_3) = 1 - \sum_{i}^{t/\Delta t - 1} \int_{\Omega} \int_{t_i}^{t_{i+1}} \delta(t - t_0) \cdot g(\hat{x},\ \hat{p_x},\ \hat{y},\ \hat{p_y})\, dt\, d\hat{x}\, d\hat{p_x}\, d\hat{y}\, d\hat{p_y}\, dt}$$



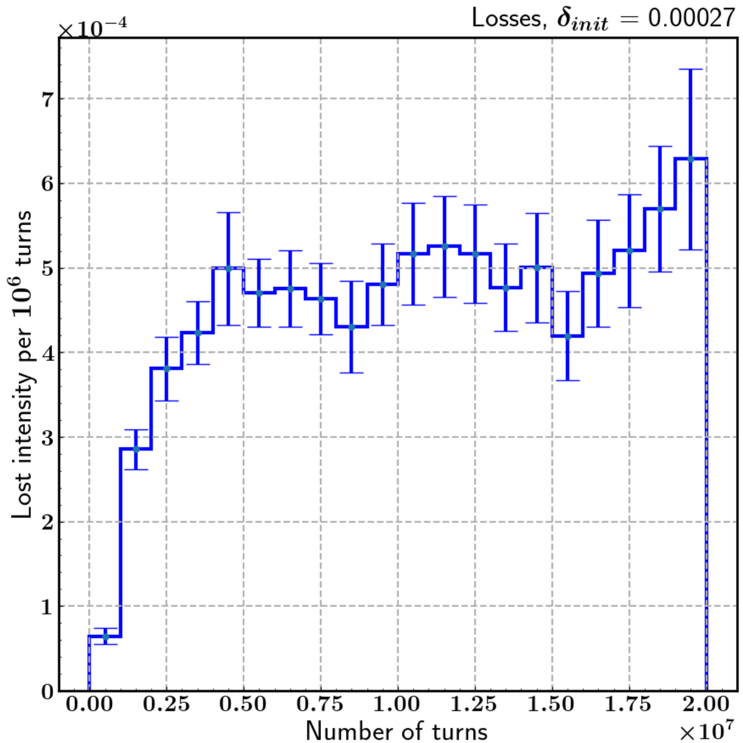


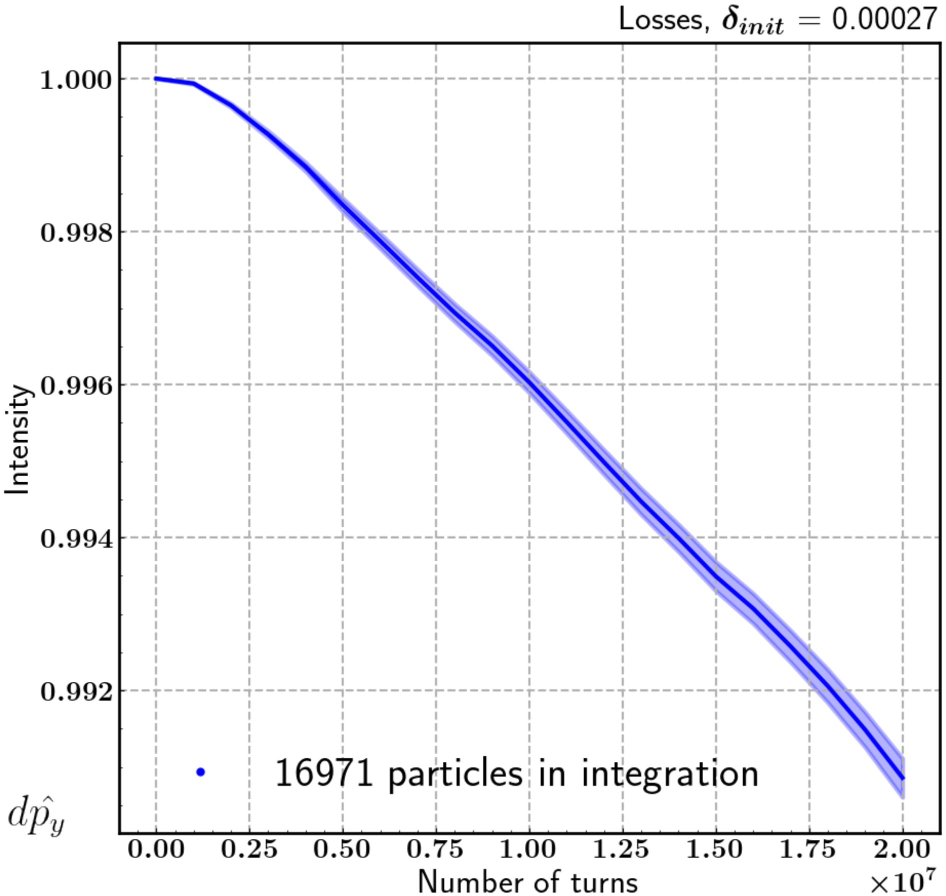
$$K(t_i,\ t_{i+1}\ t_0) = \int_{t_i}^{t_{i+1}} \delta(t - t_0)\, dt$$

$$= \begin{cases} 1, & t_i < t_0 < t_{i+1} \\ 0, & \text{otherwise} \end{cases}$$

$$L(t_i,\ t_{i+1}\ J_3) = \int_{\Omega} \Big(K(t_i,\ t_{i+1}\ t_0) \cdot g \Big)\, d\hat{x}\, d\hat{p_x}\, d\hat{y}\, d\hat{p_y}$$

$$F(t,\ J_3) = 1 - \sum_{i}^{t/\Delta t - 1} L(t_i,\ t_{i+1},\ J_3)$$





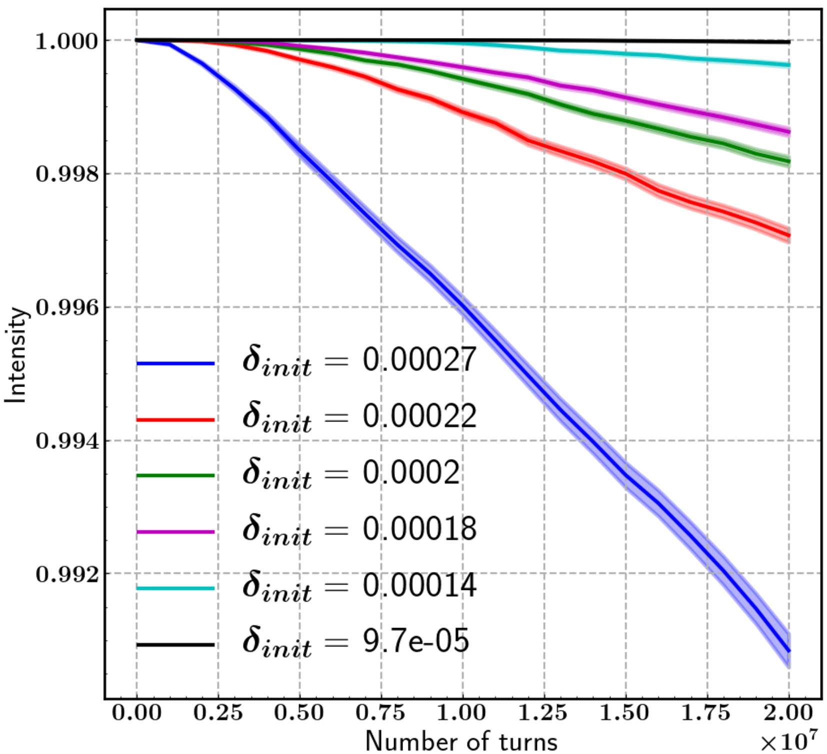
$$L(t_i,\ t_{i+1}\ J_3) = \int_\Omega \Big(K(t_i,\ t_{i+1}\ t_0)\cdot g\Big)\ d\hat{x}\ d\hat{p_x}\ d\hat{y}\ d\hat{p_y}$$

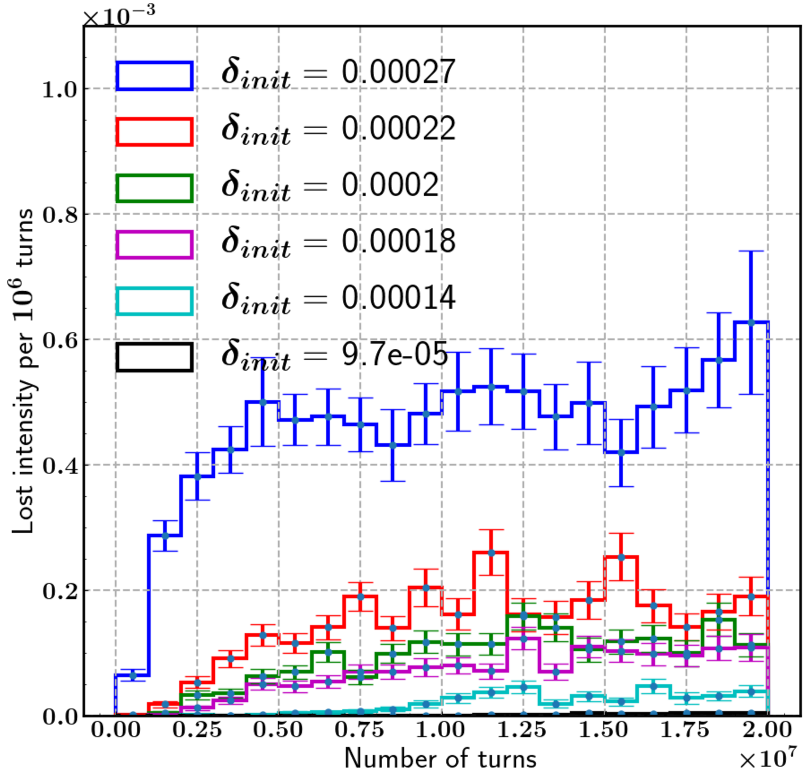
$$F(t,\ J_3) = 1 - \sum_i^{t/\Delta t - 1} L(t_i,\ t_{i+1},\ J_3)$$

Variance on intensity is **substantially reduced** – and we now have direct calculation of the losses

Step 4:
Losses Integral

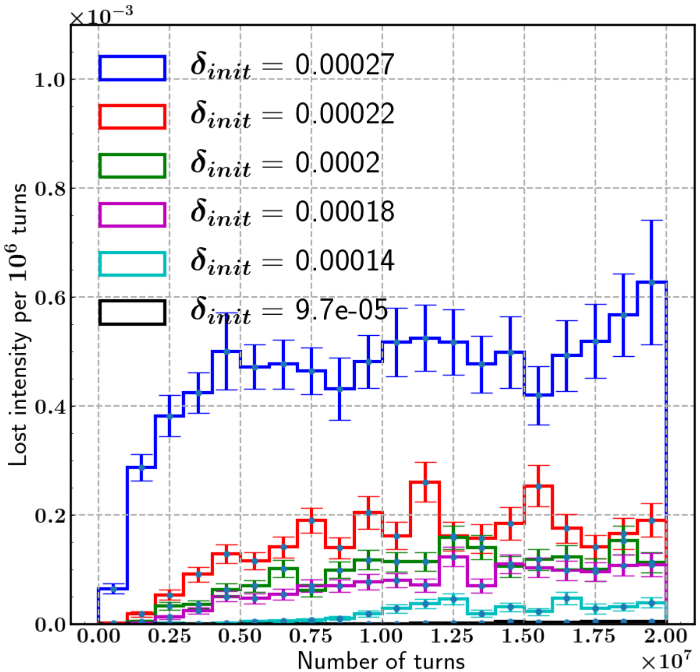
Intensity and losses by initial momentum deviations δ_{init}

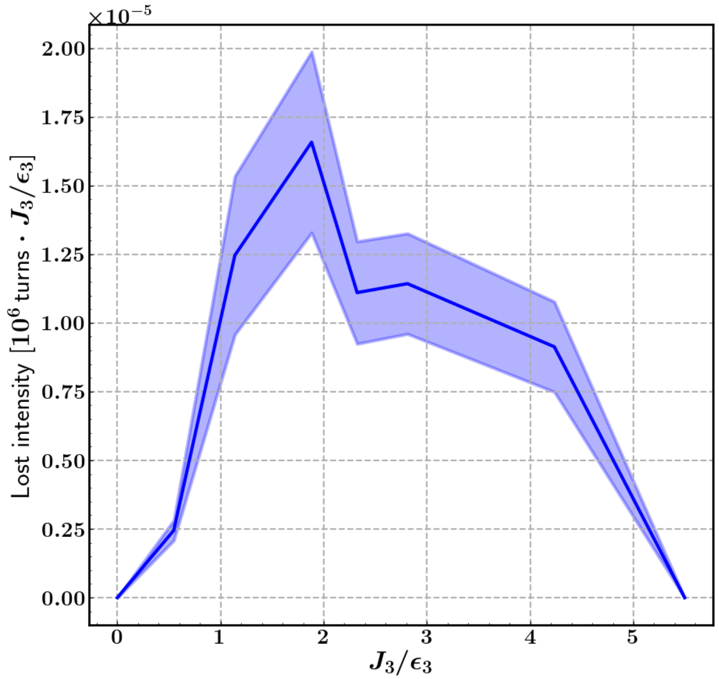




- Larger momentum deviation means more relative losses,
- All losses seem to quickly rise to a constant level

Losses as a function of longitudinal amplitude *(for last bin of time)*



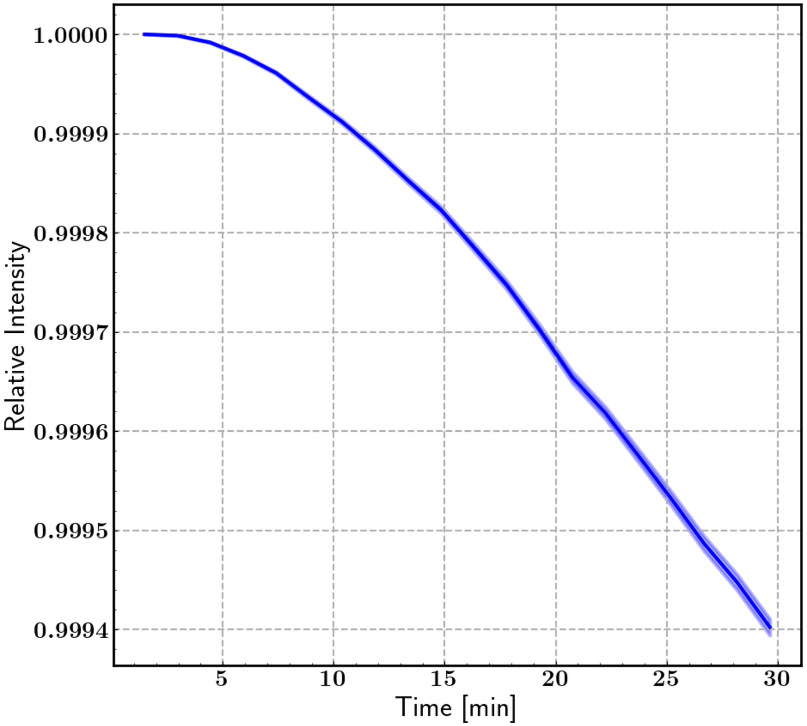


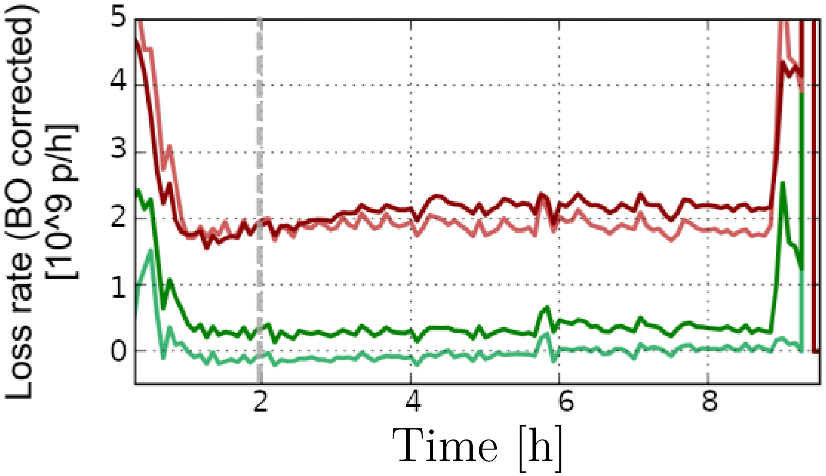
$$\int F(t, J_3) \cdot h(J_3)\, dJ_3$$

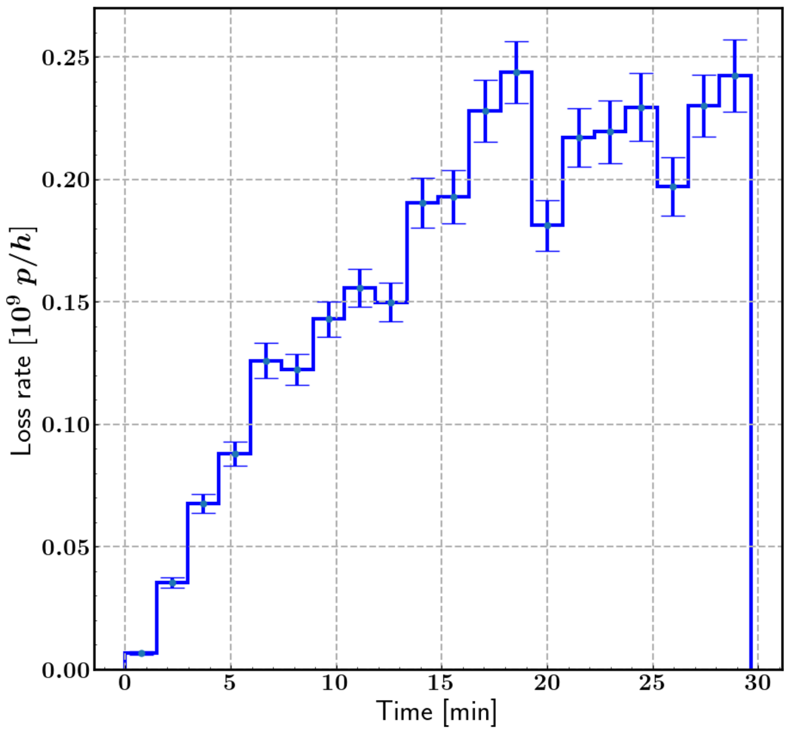
δ_{init}	J_3/ϵ_3
9.7×10^{-5}	0.55
1.4×10^{-5}	1.14
1.8×10^{-5}	1.88
2.0×10^{-5}	2.32
2.2×10^{-5}	2.81
2.7×10^{-4}	4.23

We use linear interpolation to estimate the integral and propagate errors. We then repeat the process for each different bin of time.

Total losses







We expected a **rapid decrease** of the loss, yet instead, we found an increase. Nevertheless, both plots seem to converge to very similar numbers.

Conclusions

- Defined mathematical procedure to evaluate beam losses from tracking simulations
- Made use of GPUs with Sixtracklib
- Managed to simulate some losses for 30 min time of LHC beam with the full lattice and beam-beam effects
- Learned about Monte Carlo integration, variance-reduction, and how to use them at our advantage to make computation of losses more efficient

Suggestions for further improvement:

- Using stratified sampling on the calculation of losses did not show any improvement, but it can be used in combination with importance sampling
- Study how effective extending the bounds of the initial distribution would be (ex: up to 5σ instead of 4)
- Simulate a longer timescale, to see if losses decrease at some point

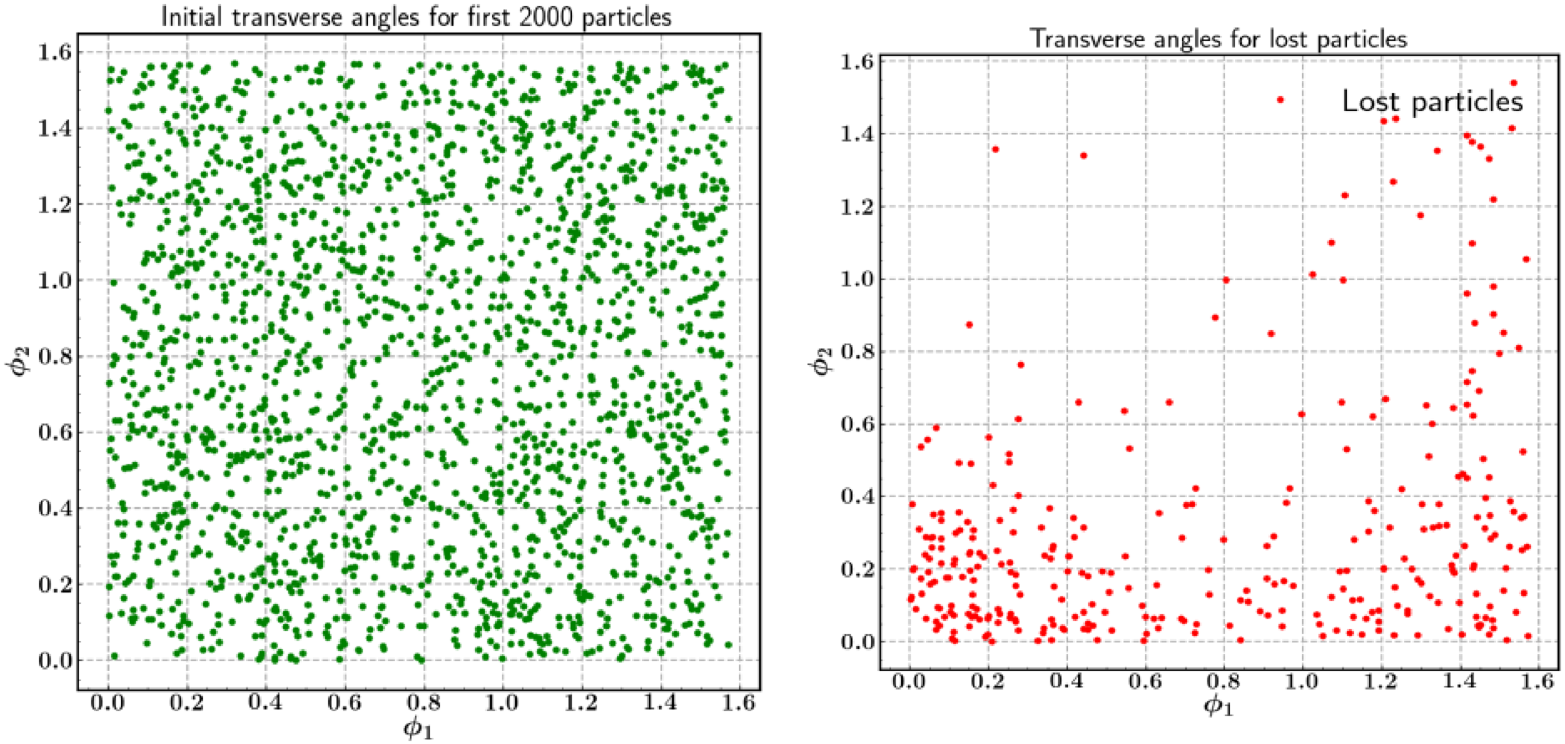
References

- Fred James. *Monte Carlo Theory and Practice*. Reports on Progress in Physics, 1980.
- Andrzej Wolski. *Beam Dynamics in High Energy Particle Accelerators*. University of Liverpool, UK, 2014.
- Alex J. Dragt. *Lie Methods for Nonlinear Dynamics with Applications to Accelerator Physics*. University of Maryland, College Park, 1991-2019.
- W. W. MacKay. *Comment on the Healy's Symplectification Algorithm*. Brookhaven National Laboratory, NY, 2006
- F. Christoph Iselin. *The MAD Program (Methodical Accelerator Design): Physical Methods Manual*. CERN, Geneva, Switzerland, 1994.
- Klaus Willle. *The Physics of Particle Accelerators: an introduction*. Oxford, 2000.

Appendices

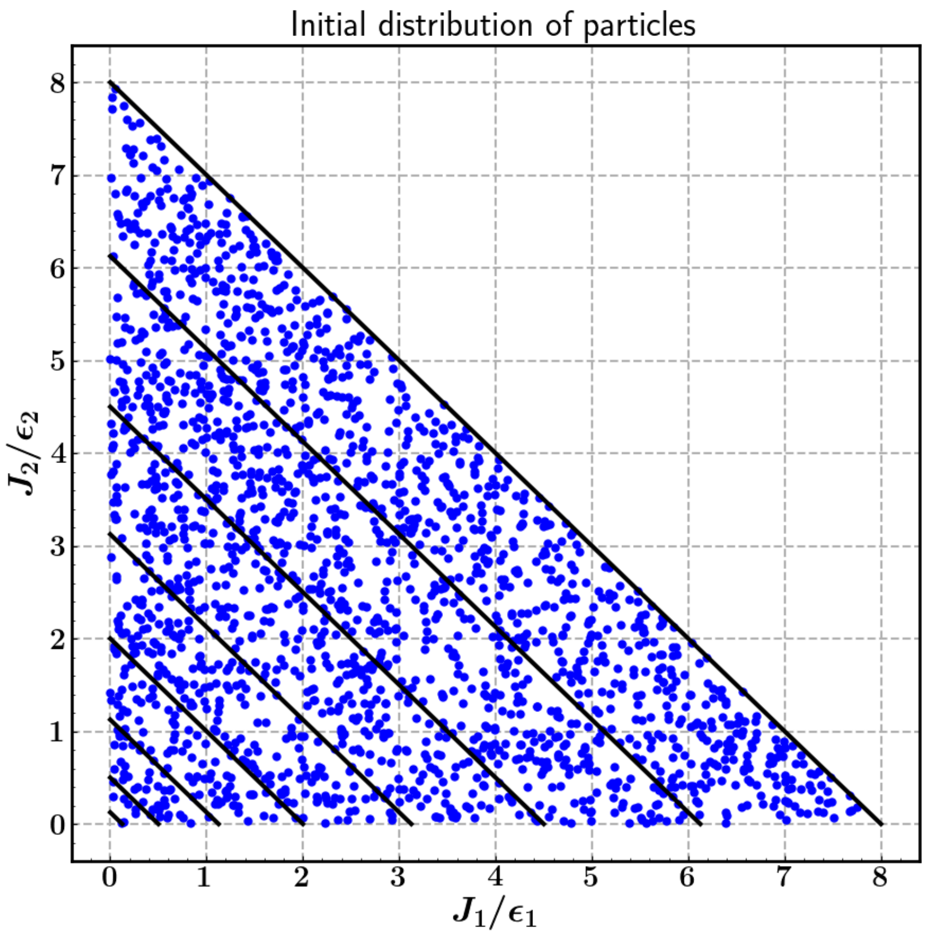
- Transverse angle distribution of initial beam vs. lost particles
- Initial transverse distribution of the particles
- Phase-space localization of losses (J, A, and ϕ)

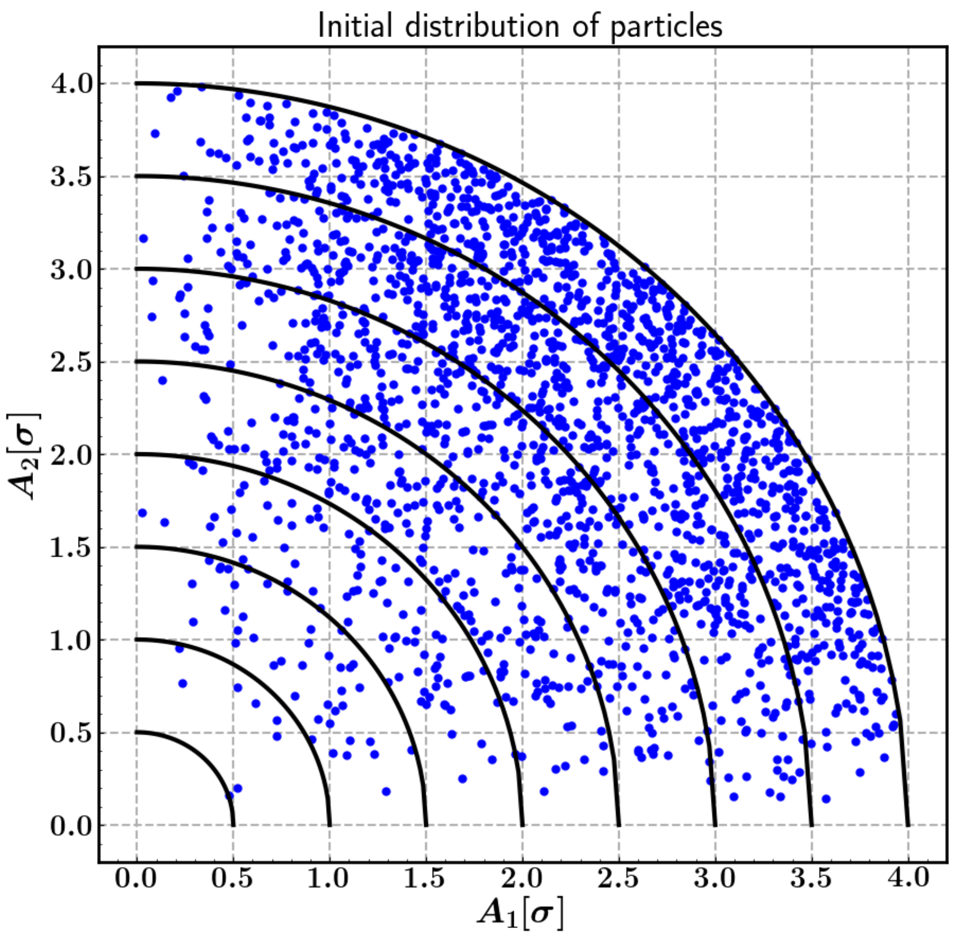
Transverse Angle distribution



The clear asymmetric dependence on ϕ is not quite understood yet and would need to be investigated further

Initial distribution of particles





(represented: 2,000 out of 20,000 particles for the [$\delta i = 0.00027$] dataset)

Phase-space localization of losses

$\delta_{init} =$ 0.97×10^{-5} 1.40×10^{-5} 1.80×10^{-5} 2.00×10^{-5} 2.20×10^{-5} 2.70×10^{-5}

Longitudinal ϕ

Longitudinal J

Longitudinal A