JUAS

Xsuite tutorial: Collective effects

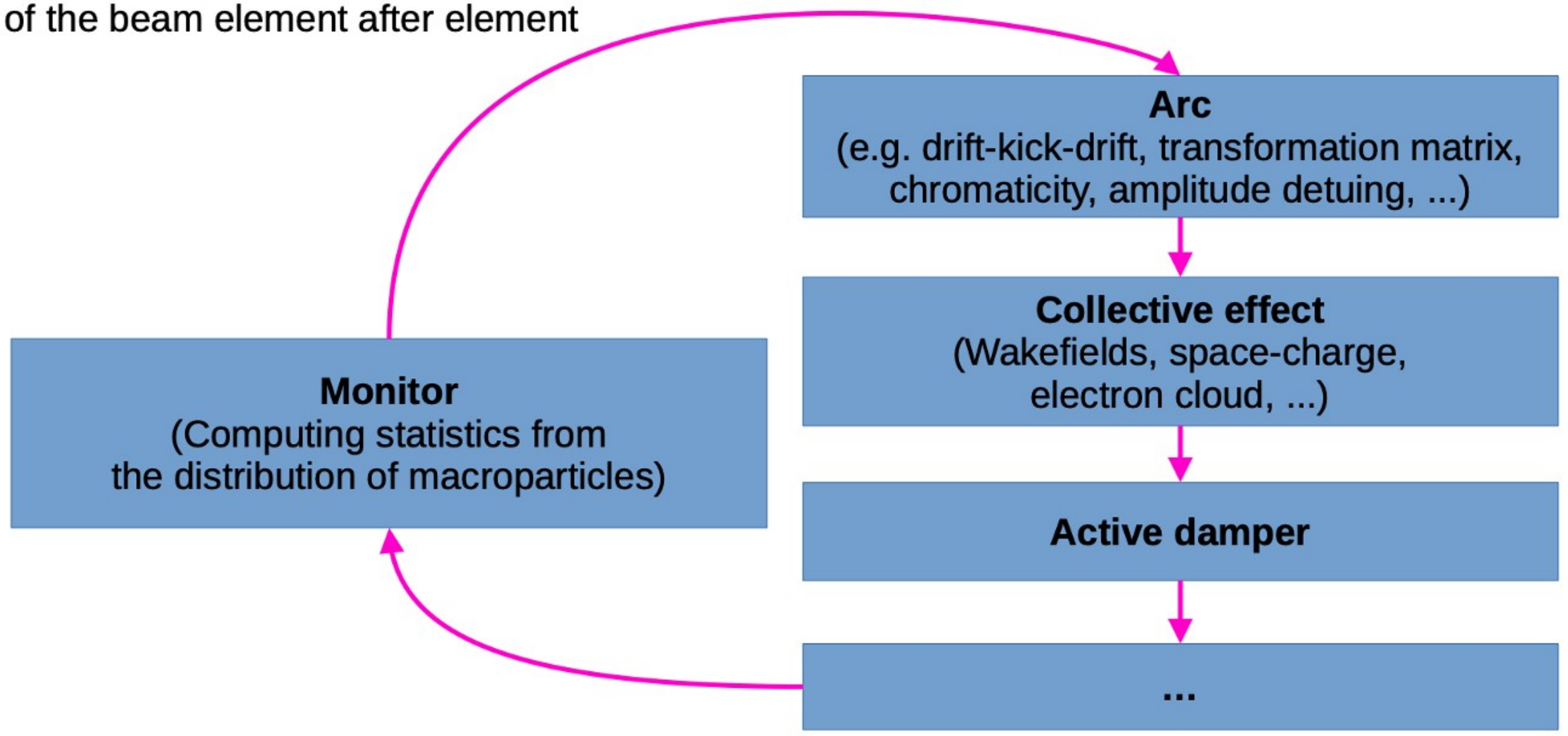
David Amorim, Elena de la Fuente García, Sébastien Joly

The tutorial was inspired from the CAS Intensity Limitations in Hadron Beams ones (many thanks for the material!)

Macro-particle tracking simulations

Data: coordinates (typically 6D) of a given number of macroparticles O(10^6)

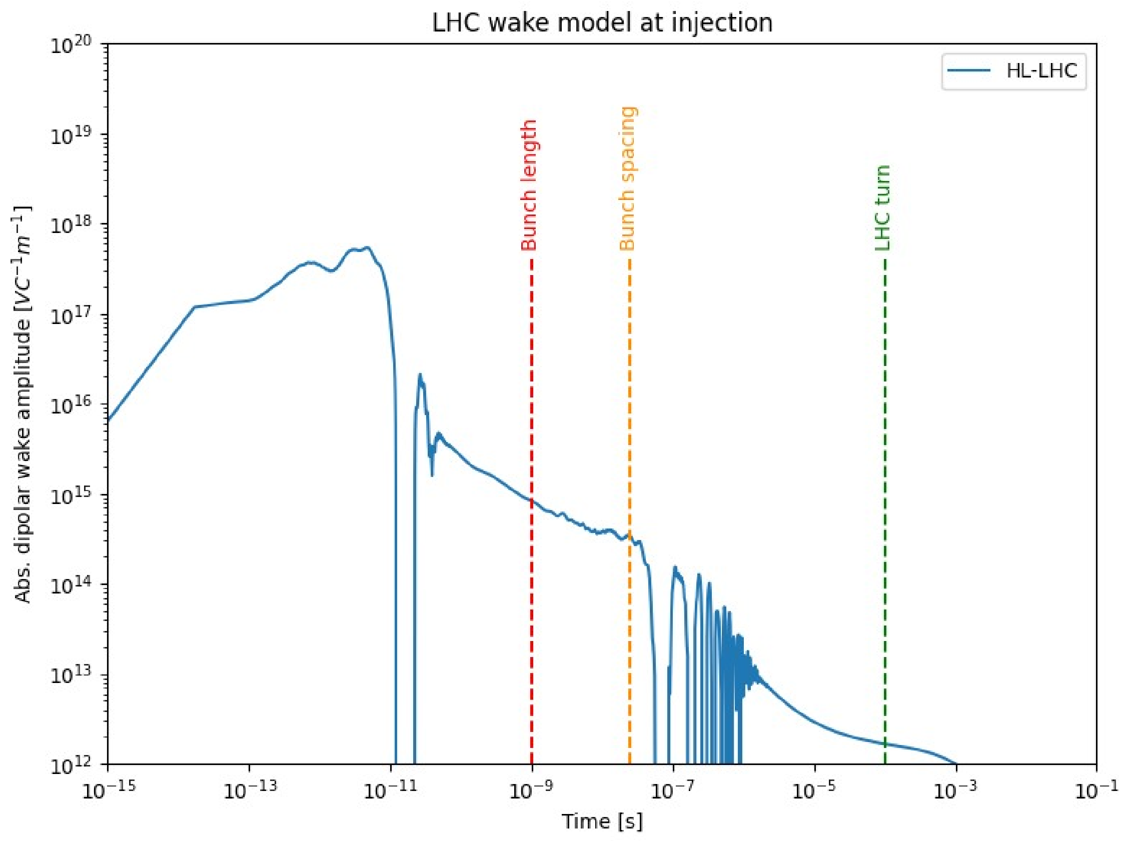
Process: Update the coordinate of the beam element after element


Arc
(e.g. drift-kick-drift, transformation matrix, chromaticity, amplitude detuing, ...)
Collective effect
(Wakefields, space-charge, electron cloud, ...)
Active damper
...
Monitor
(Computing statistics from the distribution of macroparticles)

Output: Average beam position vs time, r.m.s. beam sizes vs time, ...

HL-LHC wake model

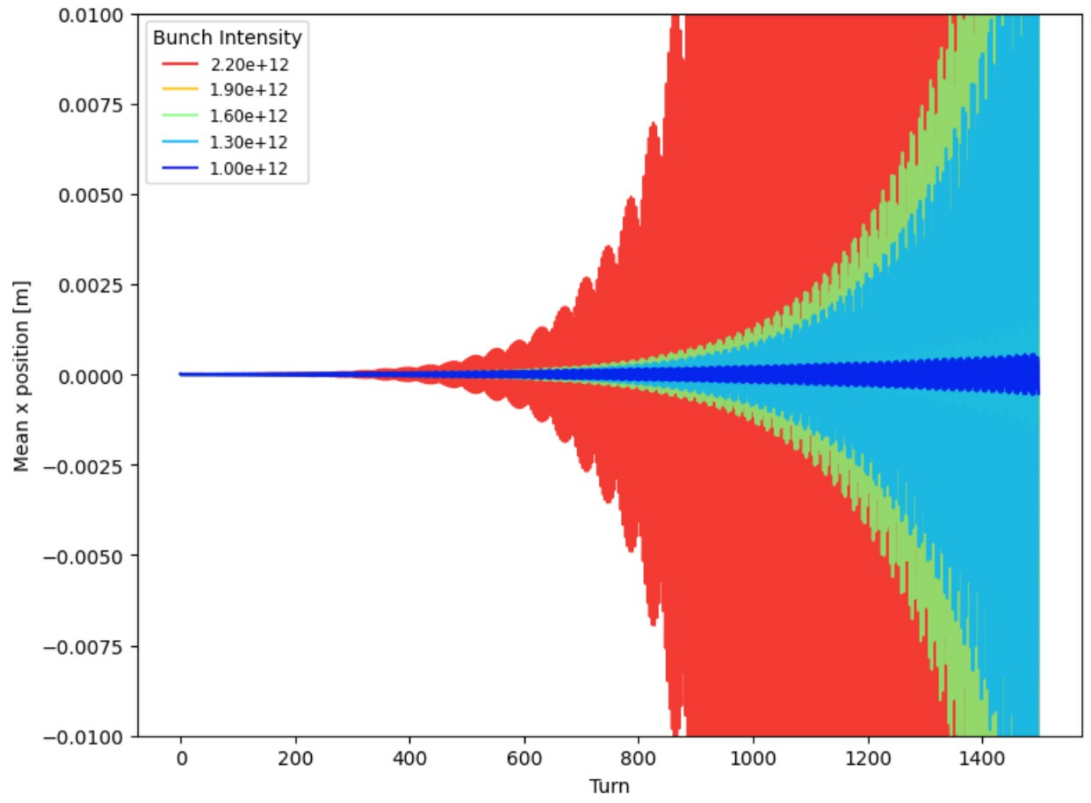
- The HL-LHC wake model includes the main wake contributors (vacuum chamber, collimators, vacuum instrumentation, beam dump, etc)
- The wakefield produced by a particle can interact with its own bunch, the following or even turns later!

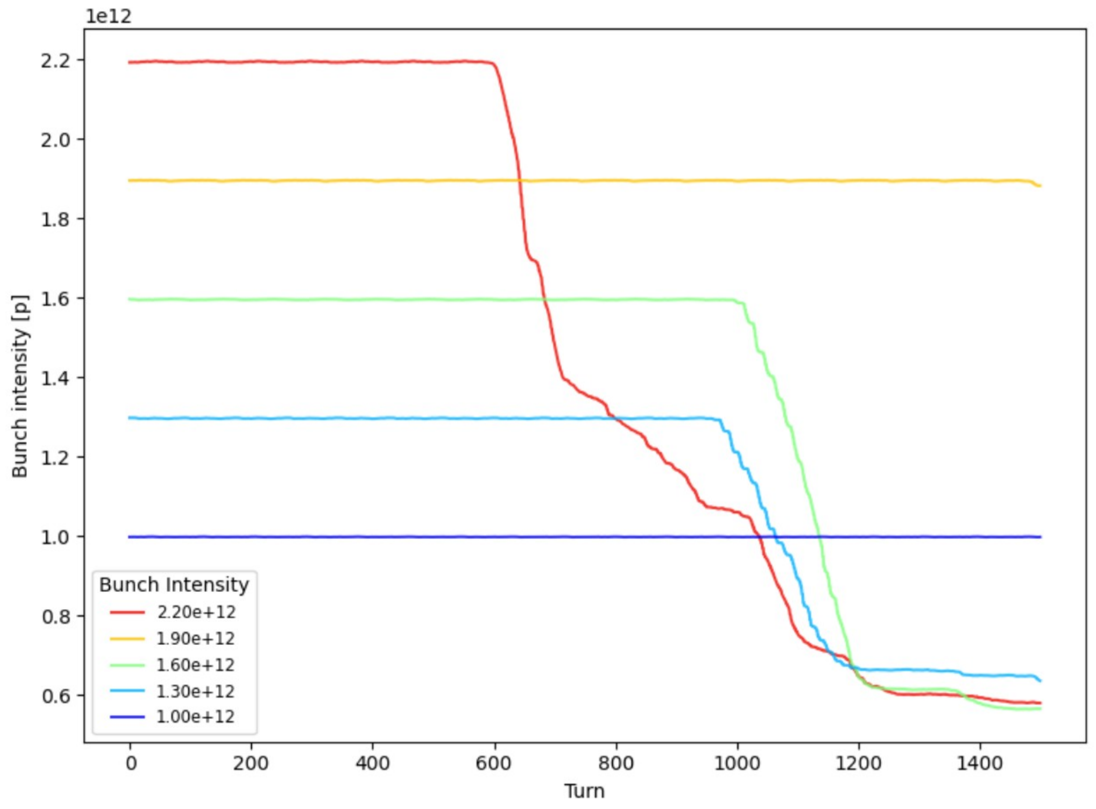


Instability signature

- Exponential growth of the bunch transverse position → bunch losses once it hits the vacuum chamber

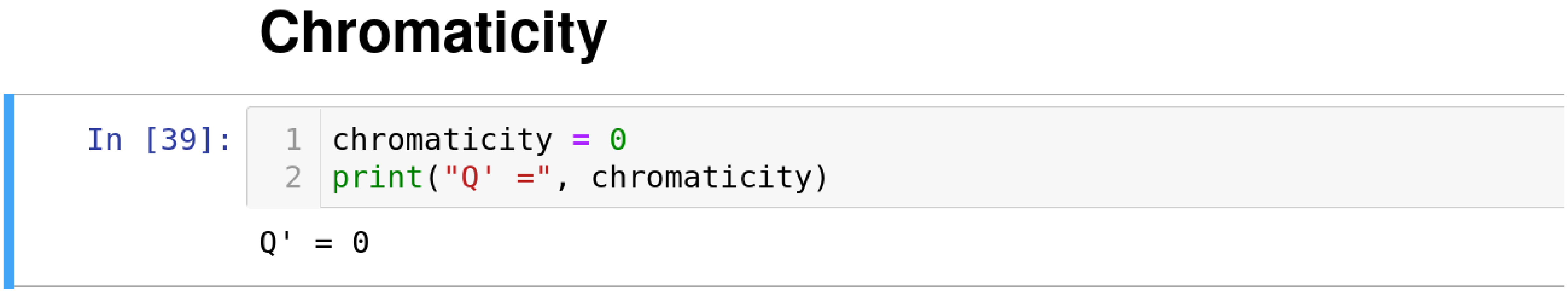
Vacuum chamber ⌀ = 1 cm





Task 1

- Scan chromaticity (stay within the range [-30,30]), when is the beam stable?



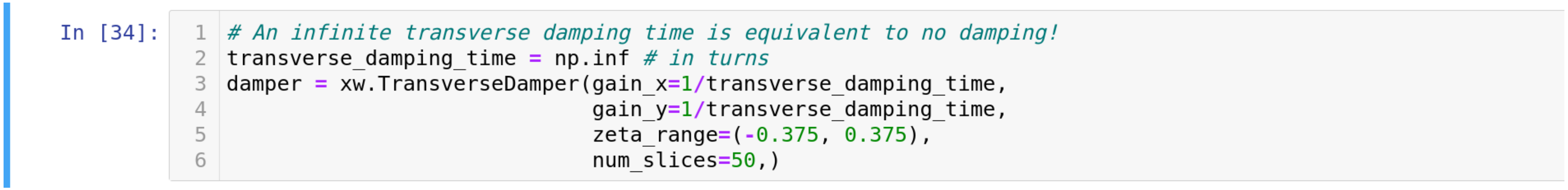
Chromaticity

```
In [39]:
chromaticity = 0
print("Q' =", chromaticity)

Q' = 0
```

- Does the sign of chromaticity play a role?
- Is having a large chromaticity usually a good idea?

Task 2

- Scan the transverse feedback damping time (stay within the range [1,∞]), when is the beam stable?



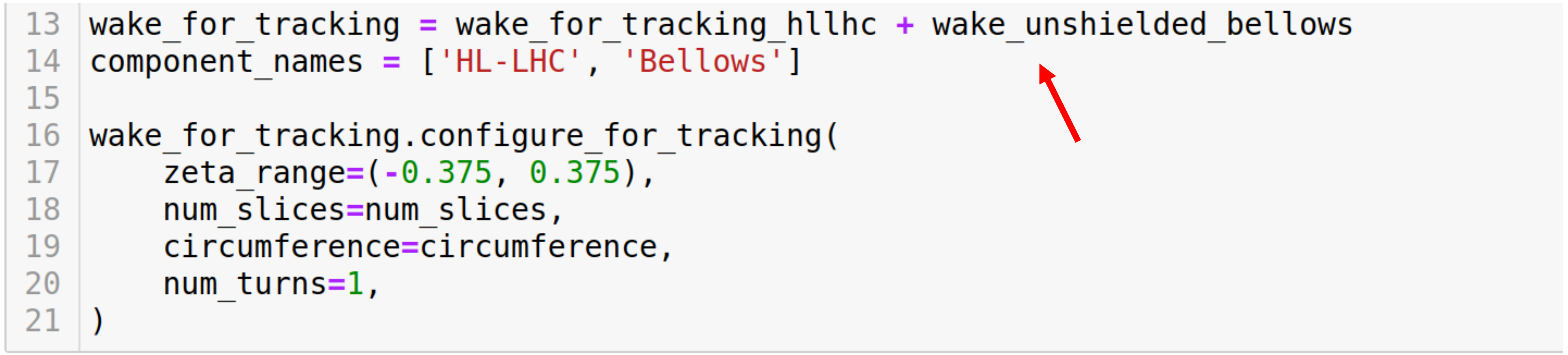
In [34]:

```python
# An infinite transverse damping time is equivalent to no damping!
transverse_damping_time = np.inf # in turns
damper = xw.TransverseDamper(gain_x=1/transverse_damping_time,
                             gain_y=1/transverse_damping_time,
                             zeta_range=(-0.375, 0.375),
                             num_slices=50,)
```

- Which damper gain would be required to stabilize the beam for a Q' = 0?

Task 3 (Bonus)

- Replace the unshielded bellows' impedance by the shielded ones, what happens?

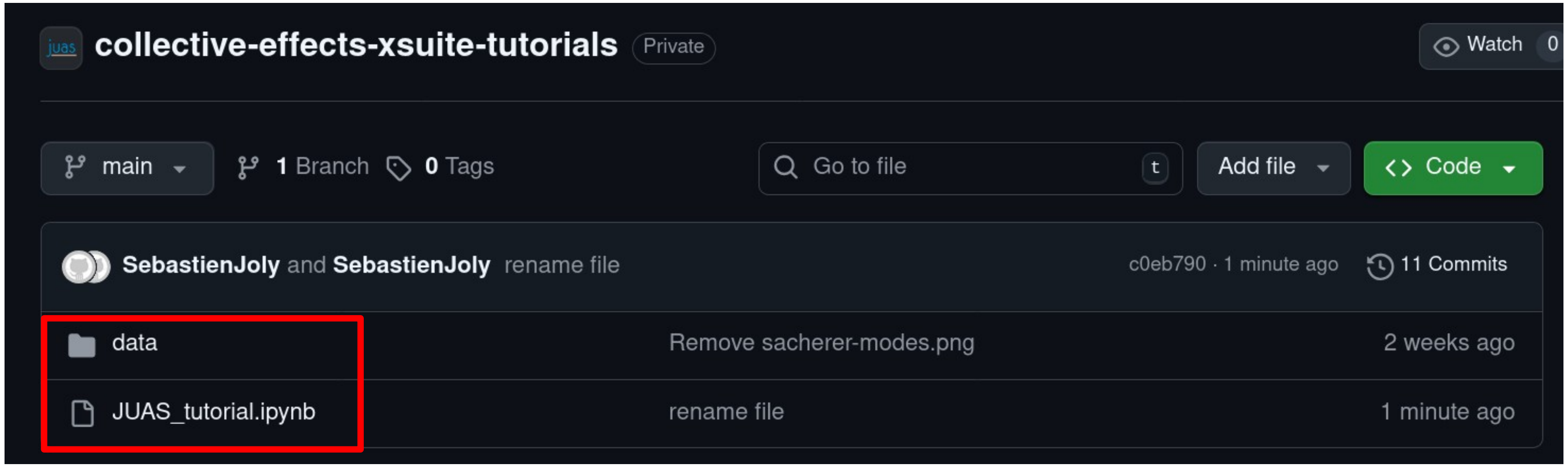


```python
wake_for_tracking = wake_for_tracking_hllhc + wake_unshielded_bellows
component_names = ['HL-LHC', 'Bellows']

wake_for_tracking.configure_for_tracking(
    zeta_range=(-0.375, 0.375),
    num_slices=num_slices,
    circumference=circumference,
    num_turns=1,
)
```

How to run the notebook

Download files from github

- https://github.com/JUAS-tutorials/collective-effects-xsuite-tutorials


collective-effects-xsuite-tutorials Private

Watch 0

main 1 Branch 0 Tags Go to file t Add file Code

SebastienJoly and SebastienJoly rename file c0eb790 · 1 minute ago 11 Commits

data	Remove sacherer-modes.png	2 weeks ago
JUAS_tutorial.ipynb	rename file	1 minute ago

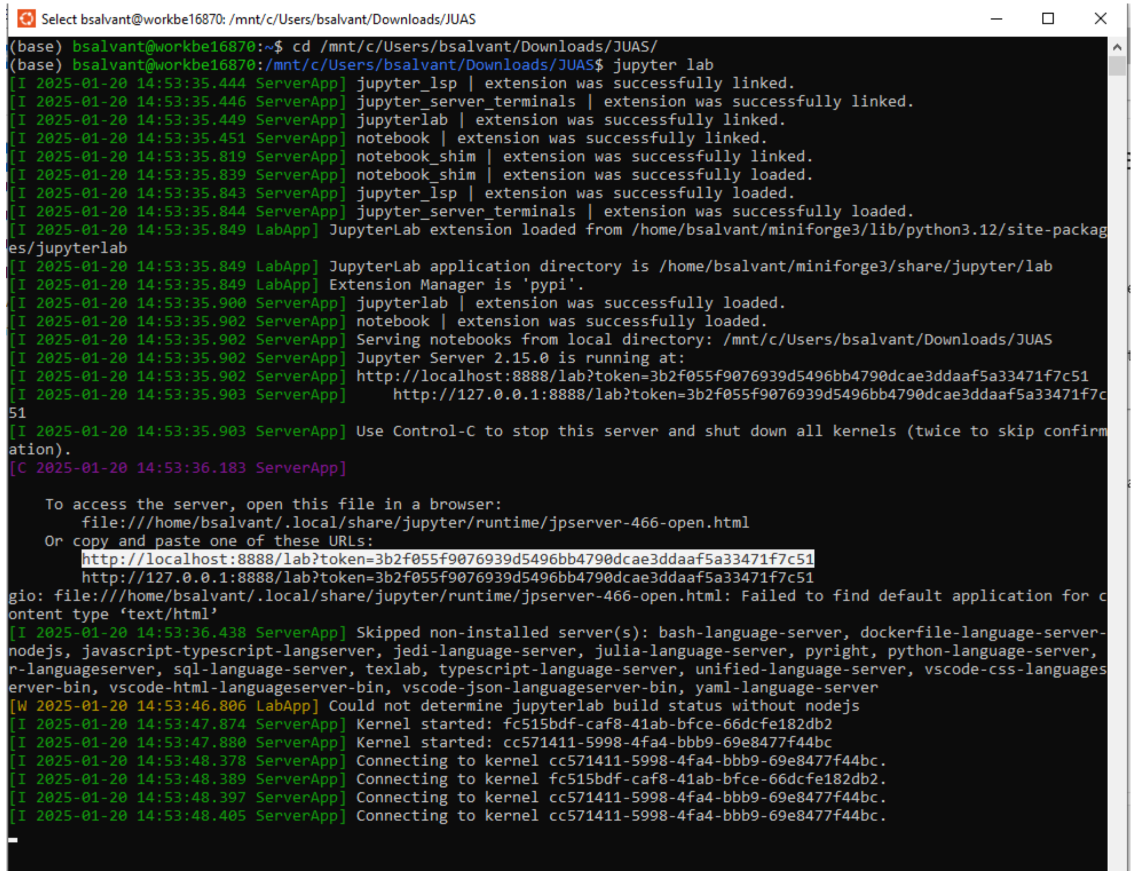
Save the files:

- Code → git clone https://github.com/JUAS-tutorials/collective-effects-xsuite-tutorials.git
- Code → Download ZIP

- Remember where you saved the files, you will need their location to open the notebook!

Open notebook in jupyter lab

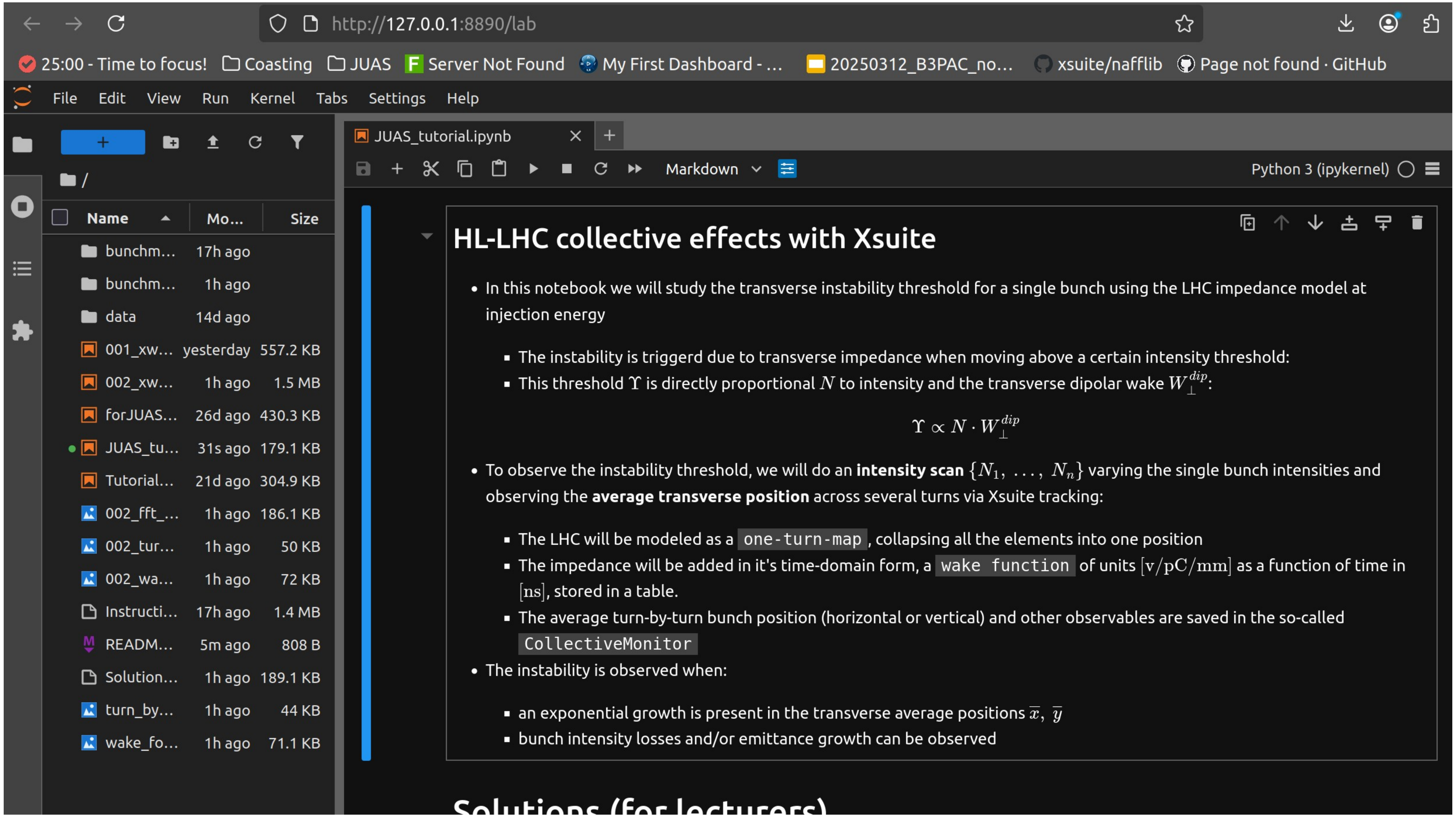
- In Ubuntu, change directory to the folder where the files are located
- Open jupyter lab
- Paste the webpage address
  http://localhost:8888/lab?token=[…]
  to any browser



Select bsalvant@workbe16870: /mnt/c/Users/bsalvant/Downloads/JUAS

```
(base) bsalvant@workbe16870:~$ cd /mnt/c/Users/bsalvant/Downloads/JUAS/
(base) bsalvant@workbe16870:/mnt/c/Users/bsalvant/Downloads/JUAS$ jupyter lab
[I 2025-01-20 14:53:35.444 ServerApp] jupyter_lsp | extension was successfully linked.
[I 2025-01-20 14:53:35.446 ServerApp] jupyter_server_terminals | extension was successfully linked.
[I 2025-01-20 14:53:35.449 ServerApp] jupyterlab | extension was successfully linked.
[I 2025-01-20 14:53:35.451 ServerApp] notebook | extension was successfully linked.
[I 2025-01-20 14:53:35.819 ServerApp] notebook_shim | extension was successfully linked.
[I 2025-01-20 14:53:35.839 ServerApp] notebook_shim | extension was successfully loaded.
[I 2025-01-20 14:53:35.843 ServerApp] jupyter_lsp | extension was successfully loaded.
[I 2025-01-20 14:53:35.844 ServerApp] jupyter_server_terminals | extension was successfully loaded.
[I 2025-01-20 14:53:35.849 LabApp] JupyterLab extension loaded from /home/bsalvant/miniforge3/lib/python3.12/site-packag
es/jupyterlab
[I 2025-01-20 14:53:35.849 LabApp] JupyterLab application directory is /home/bsalvant/miniforge3/share/jupyter/lab
[I 2025-01-20 14:53:35.849 LabApp] Extension Manager is 'pypi'.
[I 2025-01-20 14:53:35.900 ServerApp] jupyterlab | extension was successfully loaded.
[I 2025-01-20 14:53:35.902 ServerApp] notebook | extension was successfully loaded.
[I 2025-01-20 14:53:35.902 ServerApp] Serving notebooks from local directory: /mnt/c/Users/bsalvant/Downloads/JUAS
[I 2025-01-20 14:53:35.902 ServerApp] Jupyter Server 2.15.0 is running at:
[I 2025-01-20 14:53:35.902 ServerApp] http://localhost:8888/lab?token=3b2f055f9076939d5496bb4790dcae3ddaaf5a33471f7c51
[I 2025-01-20 14:53:35.903 ServerApp]     http://127.0.0.1:8888/lab?token=3b2f055f9076939d5496bb4790dcae3ddaaf5a33471f7c
51
[I 2025-01-20 14:53:35.903 ServerApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirm
ation).
[C 2025-01-20 14:53:36.183 ServerApp]

    To access the server, open this file in a browser:
        file:///home/bsalvant/.local/share/jupyter/runtime/jpserver-466-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/lab?token=3b2f055f9076939d5496bb4790dcae3ddaaf5a33471f7c51
        http://127.0.0.1:8888/lab?token=3b2f055f9076939d5496bb4790dcae3ddaaf5a33471f7c51
gio: file:///home/bsalvant/.local/share/jupyter/runtime/jpserver-466-open.html: Failed to find default application for c
ontent type ‘text/html’
[I 2025-01-20 14:53:36.438 ServerApp] Skipped non-installed server(s): bash-language-server, dockerfile-language-server-
nodejs, javascript-typescript-langserver, jedi-language-server, julia-language-server, pyright, python-language-server,
r-languageserver, sql-language-server, texlab, typescript-language-server, unified-language-server, vscode-css-languages
erver-bin, vscode-html-languageserver-bin, vscode-json-languageserver-bin, yaml-language-server
[W 2025-01-20 14:53:46.806 LabApp] Could not determine jupyterlab build status without nodejs
[I 2025-01-20 14:53:47.874 ServerApp] Kernel started: fc515bdf-caf8-41ab-bfce-66dcfe182db2
[I 2025-01-20 14:53:47.880 ServerApp] Kernel started: cc571411-5998-4fa4-bbb9-69e8477f44bc
[I 2025-01-20 14:53:48.378 ServerApp] Connecting to kernel cc571411-5998-4fa4-bbb9-69e8477f44bc.
[I 2025-01-20 14:53:48.389 ServerApp] Connecting to kernel fc515bdf-caf8-41ab-bfce-66dcfe182db2.
[I 2025-01-20 14:53:48.397 ServerApp] Connecting to kernel cc571411-5998-4fa4-bbb9-69e8477f44bc.
[I 2025-01-20 14:53:48.405 ServerApp] Connecting to kernel cc571411-5998-4fa4-bbb9-69e8477f44bc.
```

Open notebook in jupyter lab and run it!



Name	Mo...	Size
bunchm...	17h ago	
bunchm...	1h ago	
data	14d ago	
001_xw...	yesterday	557.2 KB
002_xw...	1h ago	1.5 MB
forJUAS...	26d ago	430.3 KB
JUAS_tu...	31s ago	179.1 KB
Tutorial...	21d ago	304.9 KB
002_fft_...	1h ago	186.1 KB
002_tur...	1h ago	50 KB
002_wa...	1h ago	72 KB
Instructi...	17h ago	1.4 MB
READM...	5m ago	808 B
Solution...	1h ago	189.1 KB
turn_by...	1h ago	44 KB
wake_fo...	1h ago	71.1 KB

HL-LHC collective effects with Xsuite

- In this notebook we will study the transverse instability threshold for a single bunch using the LHC impedance model at injection energy
  - The instability is triggerd due to transverse impedance when moving above a certain intensity threshold:
  - This threshold Υ is directly proportional N to intensity and the transverse dipolar wake $W_{\perp}^{dip}$:

$$\Upsilon \propto N \cdot W_{\perp}^{dip}$$

- To observe the instability threshold, we will do an **intensity scan** $\{N_1, \ldots, N_n\}$ varying the single bunch intensities and observing the **average transverse position** across several turns via Xsuite tracking:
  - The LHC will be modeled as a `one-turn-map`, collapsing all the elements into one position
  - The impedance will be added in it's time-domain form, a `wake function` of units $[\mathrm{v/pC/mm}]$ as a function of time in $[\mathrm{ns}]$, stored in a table.
  - The average turn-by-turn bunch position (horizontal or vertical) and other observables are saved in the so-called `CollectiveMonitor`
- The instability is observed when:
  - an exponential growth is present in the transverse average positions $\bar{x}$, $\bar{y}$
  - bunch intensity losses and/or emittance growth can be observed

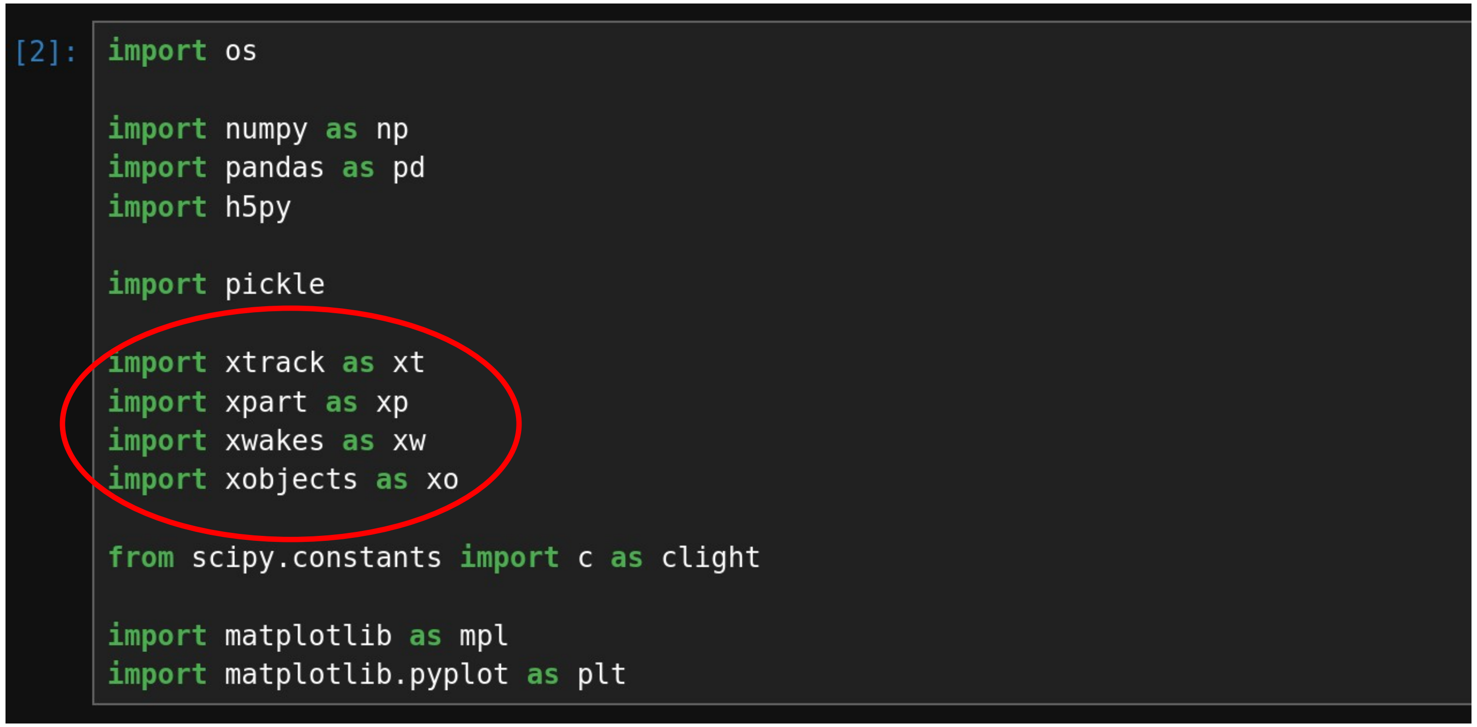
Solutions (for lecturers)

Notebook file structure

Structure of the iPython notebook (1):importing libraries

Import xtrack, xpart, xwakes and xobjects from Xsuite

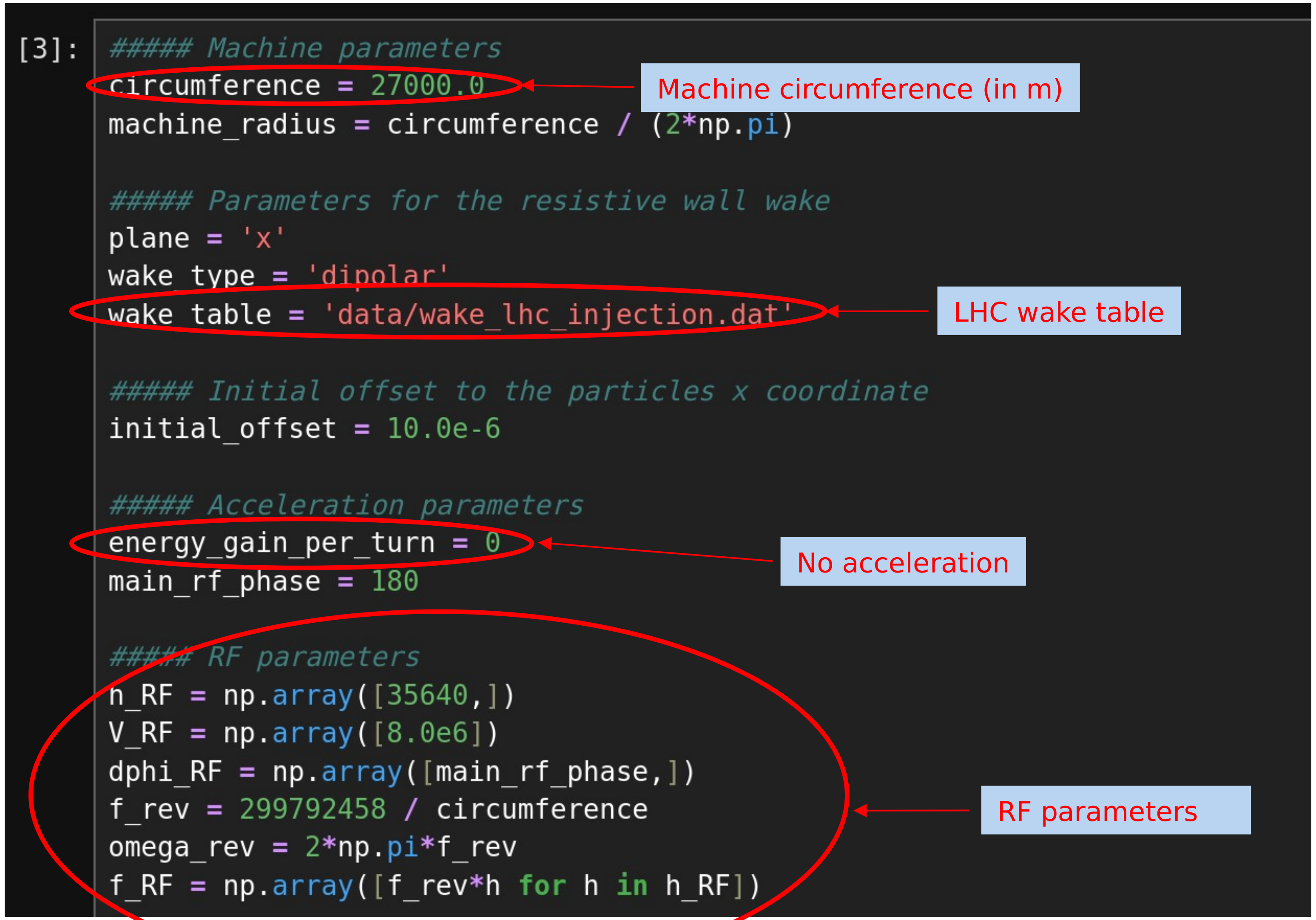
and import other required libraries and functions



```
[2]: import os

     import numpy as np
     import pandas as pd
     import h5py

     import pickle

     import xtrack as xt
     import xpart as xp
     import xwakes as xw
     import xobjects as xo

     from scipy.constants import c as clight

     import matplotlib as mpl
     import matplotlib.pyplot as plt
```

Structure of the iPython notebook (2): input parameters

Set machine and beam parameters



```
[3]: ##### Machine parameters
     circumference = 27000.0
     machine_radius = circumference / (2*np.pi)

     ##### Parameters for the resistive wall wake
     plane = 'x'
     wake_type = 'dipolar'
     wake_table = 'data/wake_lhc_injection.dat'

     ##### Initial offset to the particles x coordinate
     initial_offset = 10.0e-6

     ##### Acceleration parameters
     energy_gain_per_turn = 0
     main_rf_phase = 180

     ##### RF parameters
     h_RF = np.array([35640,])
     V_RF = np.array([8.0e6])
     dphi_RF = np.array([main_rf_phase,])
     f_rev = 299792458 / circumference
     omega_rev = 2*np.pi*f_rev
     f_RF = np.array([f_rev*h for h in h_RF])
```

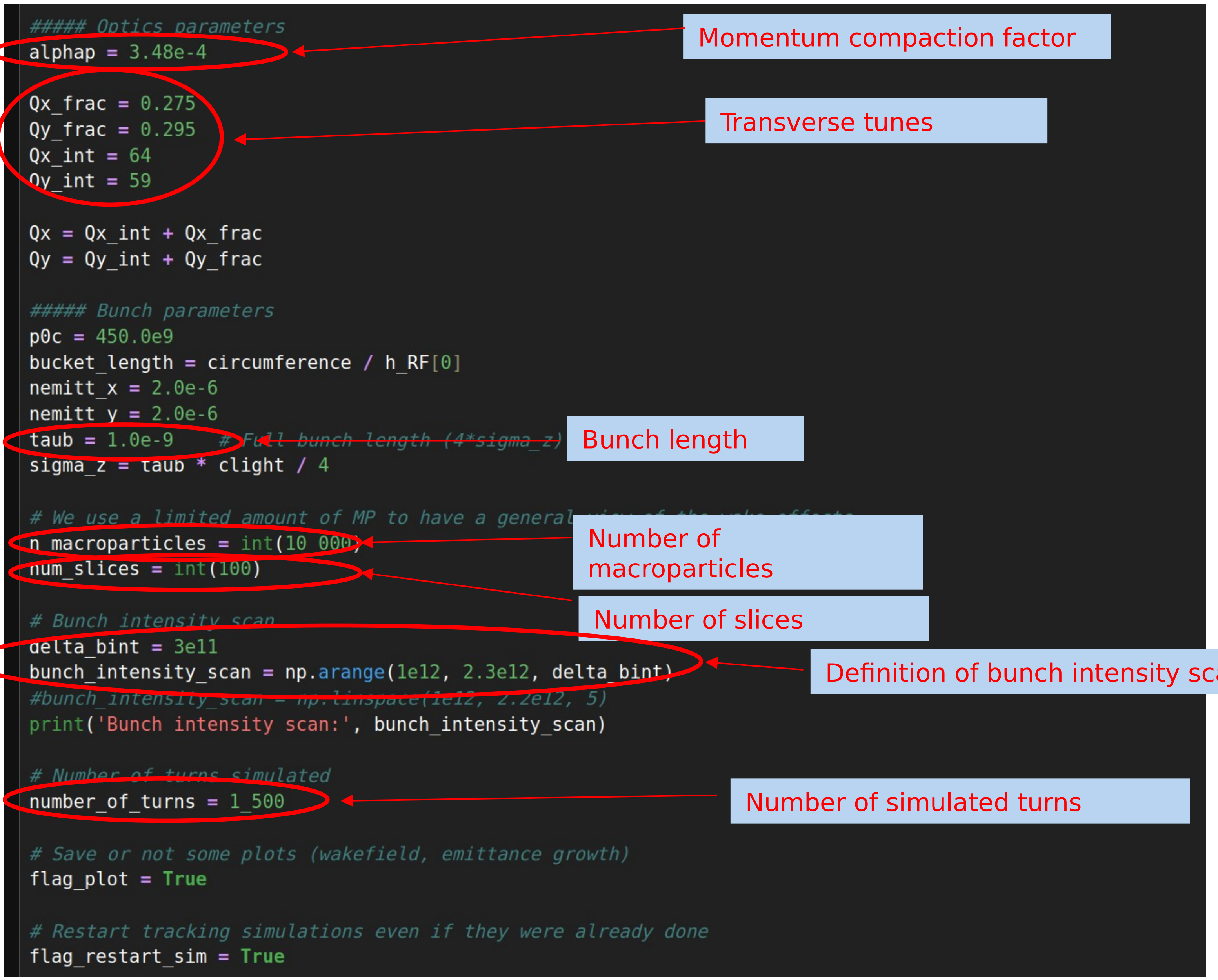
Machine circumference (in m)

LHC wake table

No acceleration

RF parameters

Structure of the Python notebook (3): input parameters



```
##### Optics parameters
alphap = 3.48e-4

Qx_frac = 0.275
Qy_frac = 0.295
Qx_int = 64
Qy_int = 59

Qx = Qx_int + Qx_frac
Qy = Qy_int + Qy_frac

##### Bunch parameters
p0c = 450.0e9
bucket_length = circumference / h_RF[0]
nemitt_x = 2.0e-6
nemitt_y = 2.0e-6
taub = 1.0e-9     # Full bunch length (4*sigma_z)
sigma_z = taub * clight / 4

# We use a limited amount of MP to have a general
n_macroparticles = int(10_000)
num_slices = int(100)

# Bunch intensity scan
delta_bint = 3e11
bunch_intensity_scan = np.arange(1e12, 2.3e12, delta_bint)
#bunch_intensity_scan = np.linspace(1e12, 2.2e12, 5)
print('Bunch intensity scan:', bunch_intensity_scan)

# Number of turns simulated
number_of_turns = 1_500

# Save or not some plots (wakefield, emittance growth)
flag_plot = True

# Restart tracking simulations even if they were already done
flag_restart_sim = True
```

Optics parameters

Initial bunch parameters

Simulation parameters

Momentum compaction factor

Transverse tunes

Bunch length

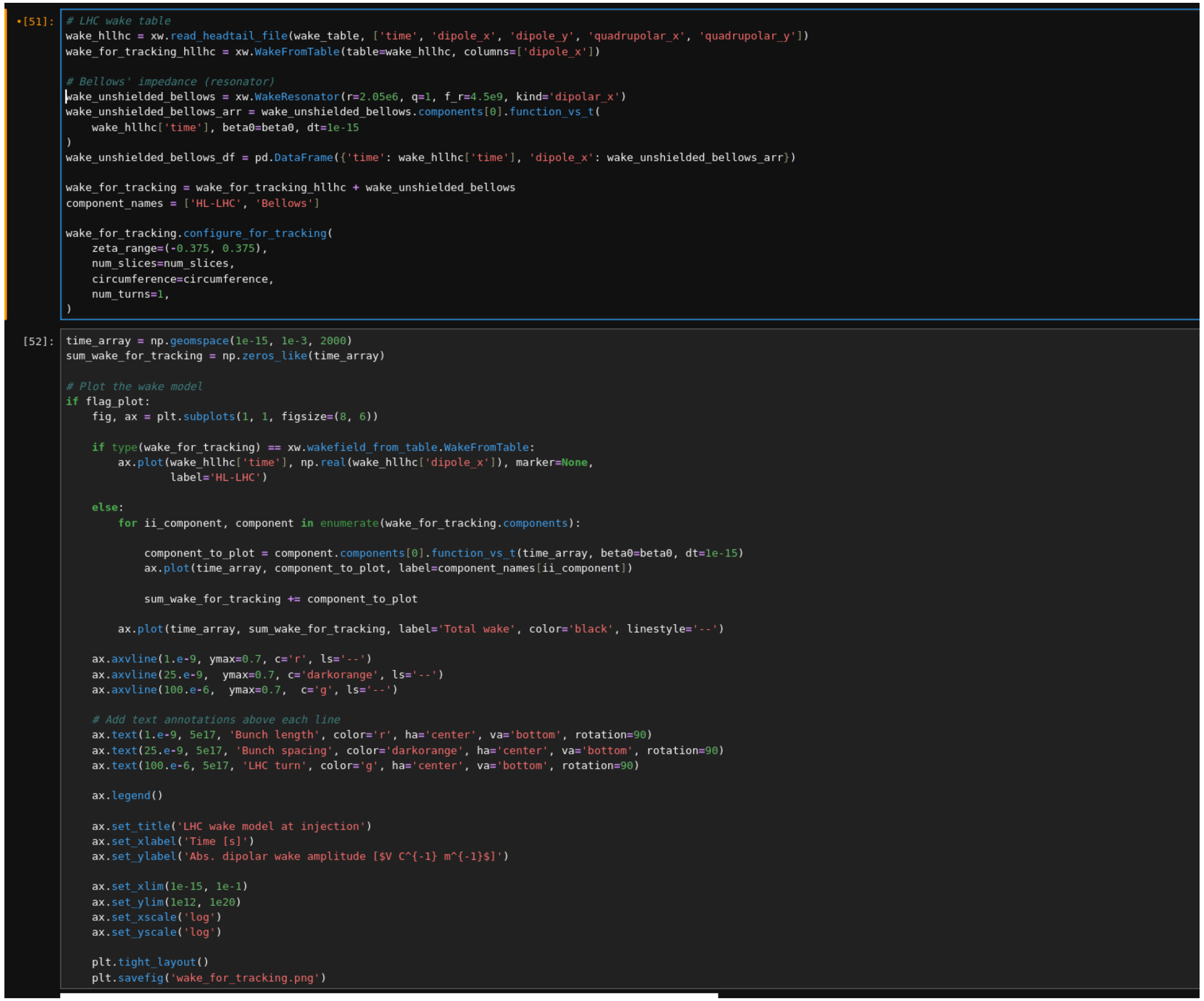
Number of macroparticles

Number of slices

Definition of bunch intensity scan

Number of simulated turns

Structure of the iPython notebook (4):



Import and configure wakes

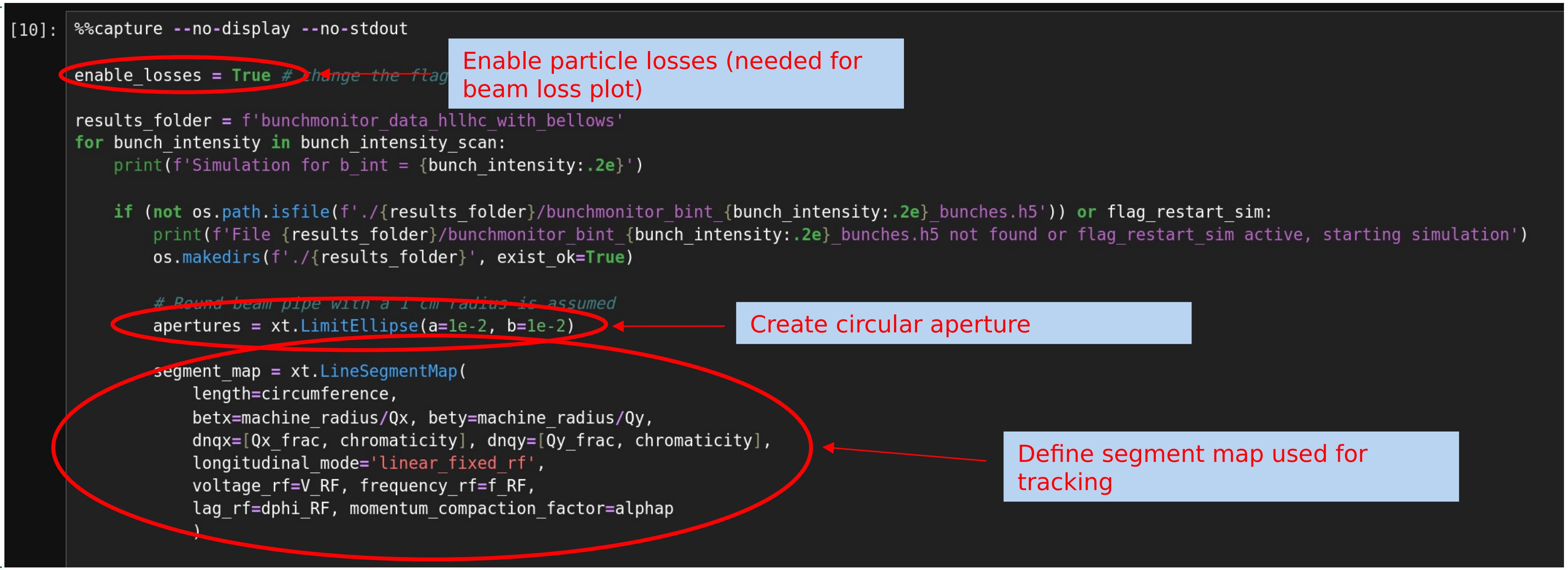
```
•[51]: # LHC wake table
wake_hllhc = xw.read_headtail_file(wake_table, ['time', 'dipole_x', 'dipole_y', 'quadrupolar_x', 'quadrupolar_y'])
wake_for_tracking_hllhc = xw.WakeFromTable(table=wake_hllhc, columns=['dipole_x'])

# Bellows' impedance (resonator)
wake_unshielded_bellows = xw.WakeResonator(r=2.05e6, q=1, f_r=4.5e9, kind='dipolar_x')
wake_unshielded_bellows_arr = wake_unshielded_bellows.components[0].function_vs_t(
    wake_hllhc['time'], beta0=beta0, dt=1e-15
)
wake_unshielded_bellows_df = pd.DataFrame({'time': wake_hllhc['time'], 'dipole_x': wake_unshielded_bellows_arr})

wake_for_tracking = wake_for_tracking_hllhc + wake_unshielded_bellows
component_names = ['HL-LHC', 'Bellows']

wake_for_tracking.configure_for_tracking(
    zeta_range=(-0.375, 0.375),
    num_slices=num_slices,
    circumference=circumference,
    num_turns=1,
)
```

Plot the input wakes

```
[52]: time_array = np.geomspace(1e-15, 1e-3, 2000)
sum_wake_for_tracking = np.zeros_like(time_array)

# Plot the wake model
if flag_plot:
    fig, ax = plt.subplots(1, 1, figsize=(8, 6))

    if type(wake_for_tracking) == xw.wakefield_from_table.WakeFromTable:
        ax.plot(wake_hllhc['time'], np.real(wake_hllhc['dipole_x']), marker=None,
                label='HL-LHC')

    else:
        for ii_component, component in enumerate(wake_for_tracking.components):

            component_to_plot = component.components[0].function_vs_t(time_array, beta0=beta0, dt=1e-15)
            ax.plot(time_array, component_to_plot, label=component_names[ii_component])

            sum_wake_for_tracking += component_to_plot

        ax.plot(time_array, sum_wake_for_tracking, label='Total wake', color='black', linestyle='--')

    ax.axvline(1.e-9, ymax=0.7, c='r', ls='--')
    ax.axvline(25.e-9,  ymax=0.7, c='darkorange', ls='--')
    ax.axvline(100.e-6,  ymax=0.7,  c='g', ls='--')

    # Add text annotations above each line
    ax.text(1.e-9, 5e17, 'Bunch length', color='r', ha='center', va='bottom', rotation=90)
    ax.text(25.e-9, 5e17, 'Bunch spacing', color='darkorange', ha='center', va='bottom', rotation=90)
    ax.text(100.e-6, 5e17, 'LHC turn', color='g', ha='center', va='bottom', rotation=90)

    ax.legend()

    ax.set_title('LHC wake model at injection')
    ax.set_xlabel('Time [s]')
    ax.set_ylabel('Abs. dipolar wake amplitude [$V C^{-1} m^{-1}$]')

    ax.set_xlim(1e-15, 1e-1)
    ax.set_ylim(1e12, 1e20)
    ax.set_xscale('log')
    ax.set_yscale('log')

    plt.tight_layout()
    plt.savefig('wake_for_tracking.png')
```

Structure of the iPython notebook (5):

Loop over bunch intensities to be scanned



```
[10]: %%capture --no-display --no-stdout

      enable_losses = True # change the flag

      results_folder = f'bunchmonitor_data_hllhc_with_bellows'
      for bunch_intensity in bunch_intensity_scan:
          print(f'Simulation for b_int = {bunch_intensity:.2e}')

          if (not os.path.isfile(f'./{results_folder}/bunchmonitor_bint_{bunch_intensity:.2e}_bunches.h5')) or flag_restart_sim:
              print(f'File {results_folder}/bunchmonitor_bint_{bunch_intensity:.2e}_bunches.h5 not found or flag_restart_sim active, starting simulation')
              os.makedirs(f'./{results_folder}', exist_ok=True)

              # Round beam pipe with a 1 cm radius is assumed
              apertures = xt.LimitEllipse(a=1e-2, b=1e-2)

              segment_map = xt.LineSegmentMap(
                  length=circumference,
                  betx=machine_radius/Qx, bety=machine_radius/Qy,
                  dnqx=[Qx_frac, chromaticity], dnqy=[Qy_frac, chromaticity],
                  longitudinal_mode='linear_fixed_rf',
                  voltage_rf=V_RF, frequency_rf=f_RF,
                  lag_rf=dphi_RF, momentum_compaction_factor=alphap
                  )
```

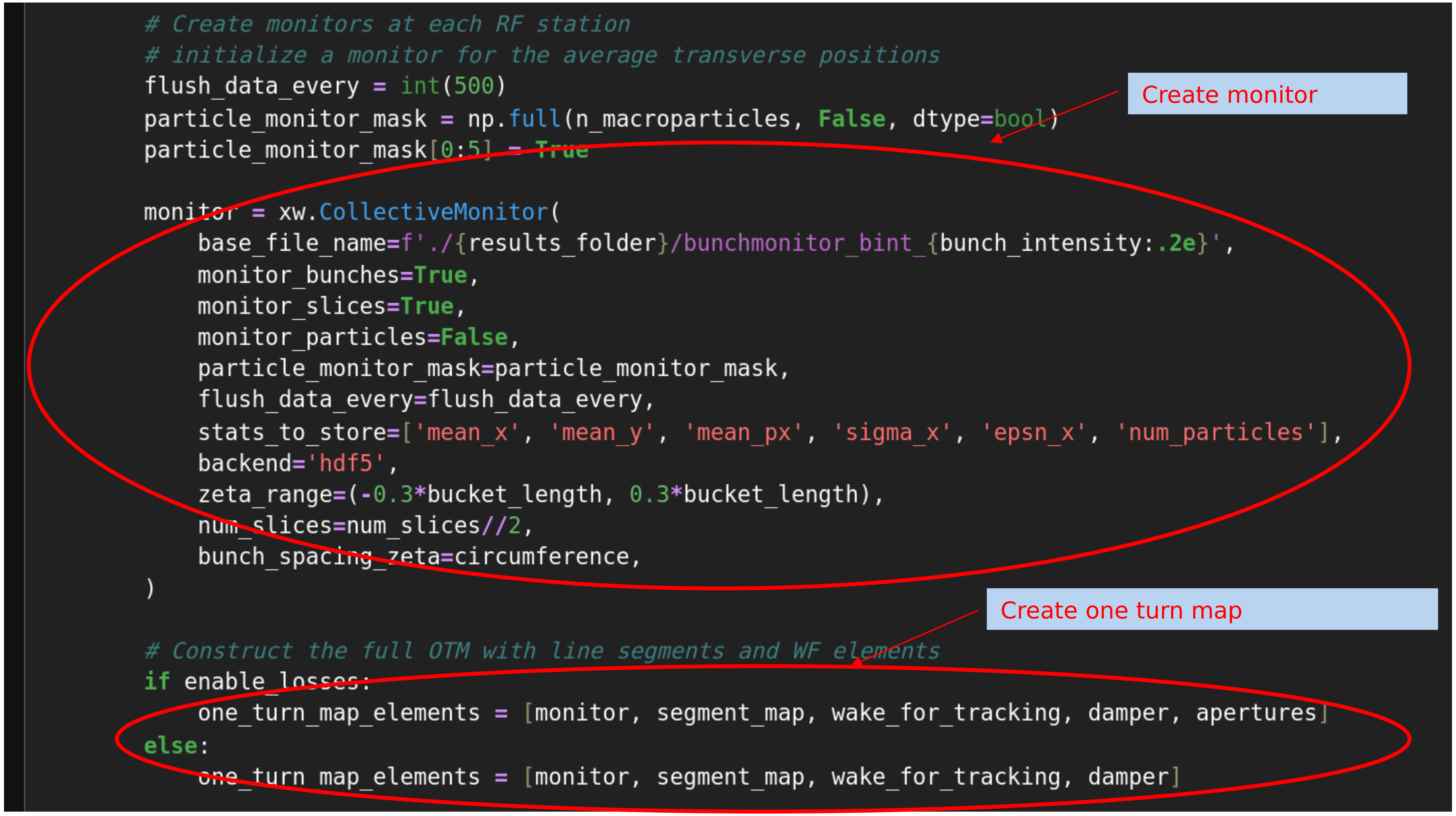
Enable particle losses (needed for beam loss plot)

Create circular aperture

Define segment map used for tracking

Structure of the iPython notebook (6):

Loop over bunch intensities to be scanned



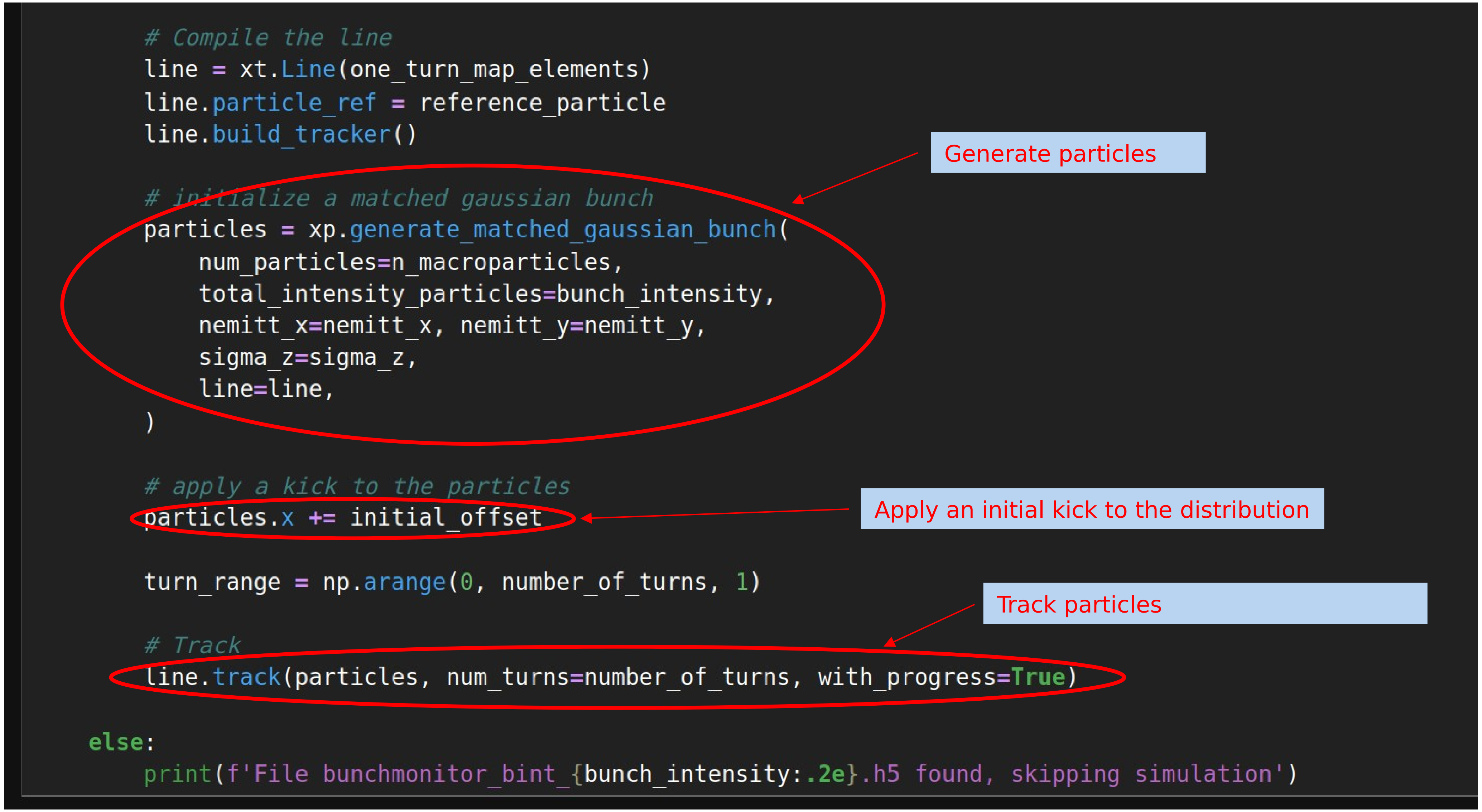
```
# Create monitors at each RF station
# initialize a monitor for the average transverse positions
flush_data_every = int(500)
particle_monitor_mask = np.full(n_macroparticles, False, dtype=bool)
particle_monitor_mask[0:5] = True

monitor = xw.CollectiveMonitor(
    base_file_name=f'./{results_folder}/bunchmonitor_bint_{bunch_intensity:.2e}',
    monitor_bunches=True,
    monitor_slices=True,
    monitor_particles=False,
    particle_monitor_mask=particle_monitor_mask,
    flush_data_every=flush_data_every,
    stats_to_store=['mean_x', 'mean_y', 'mean_px', 'sigma_x', 'epsn_x', 'num_particles'],
    backend='hdf5',
    zeta_range=(-0.3*bucket_length, 0.3*bucket_length),
    num_slices=num_slices//2,
    bunch_spacing_zeta=circumference,
)

# Construct the full OTM with line segments and WF elements
if enable_losses:
    one_turn_map_elements = [monitor, segment_map, wake_for_tracking, damper, apertures]
else:
    one_turn_map_elements = [monitor, segment_map, wake_for_tracking, damper]
```

Create monitor

Create one turn map

Structure of the iPython notebook (7):

Loop over bunch intensities to be scanned



```
        # Compile the line
        line = xt.Line(one_turn_map_elements)
        line.particle_ref = reference_particle
        line.build_tracker()

        # initialize a matched gaussian bunch
        particles = xp.generate_matched_gaussian_bunch(
            num_particles=n_macroparticles,
            total_intensity_particles=bunch_intensity,
            nemitt_x=nemitt_x, nemitt_y=nemitt_y,
            sigma_z=sigma_z,
            line=line,
        )

        # apply a kick to the particles
        particles.x += initial_offset

        turn_range = np.arange(0, number_of_turns, 1)

        # Track
        line.track(particles, num_turns=number_of_turns, with_progress=True)

    else:
        print(f'File bunchmonitor_bint_{bunch_intensity:.2e}.h5 found, skipping simulation')
```

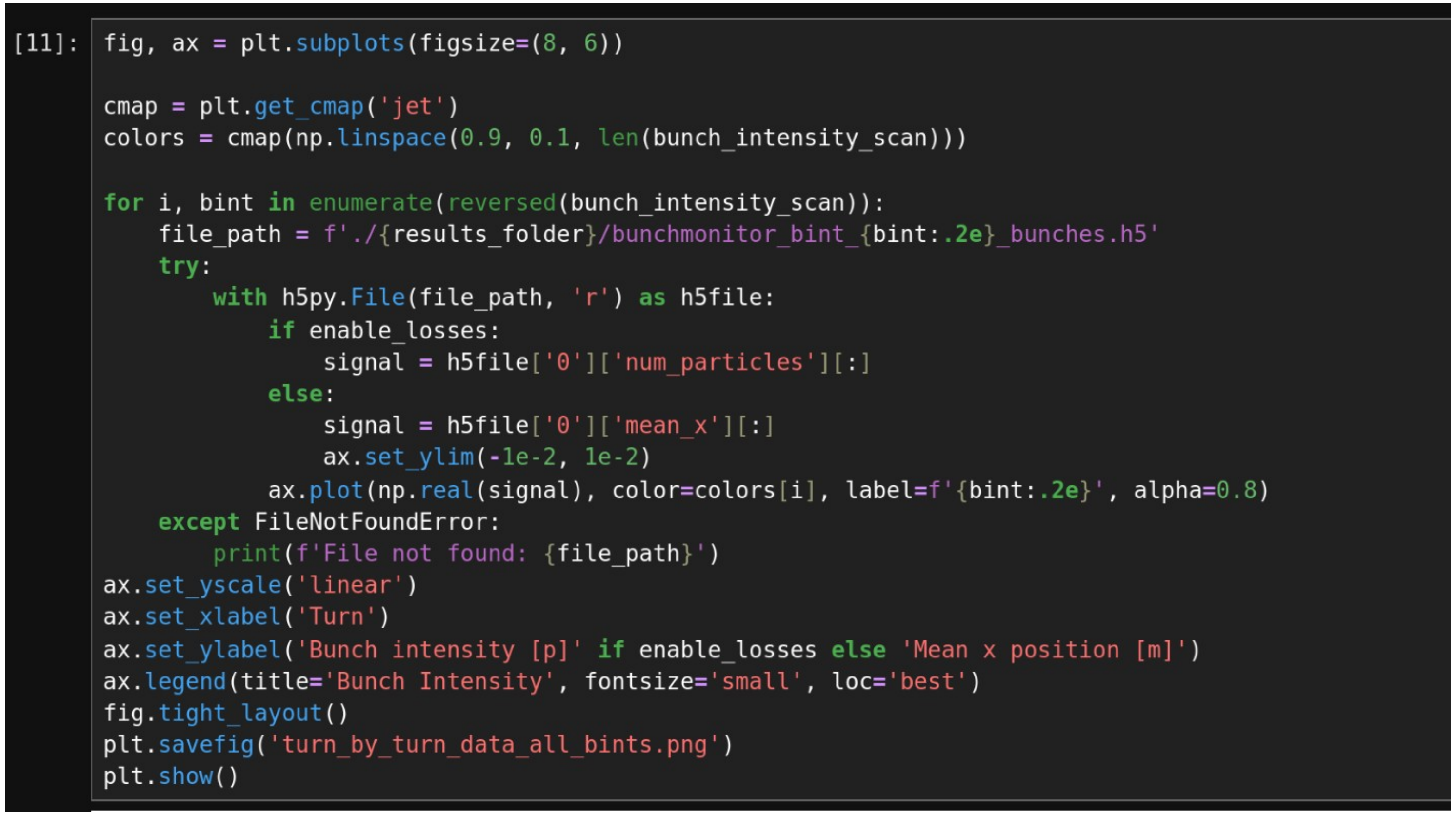
Generate particles

Apply an initial kick to the distribution

Track particles

Structure of the Python notebook (7)

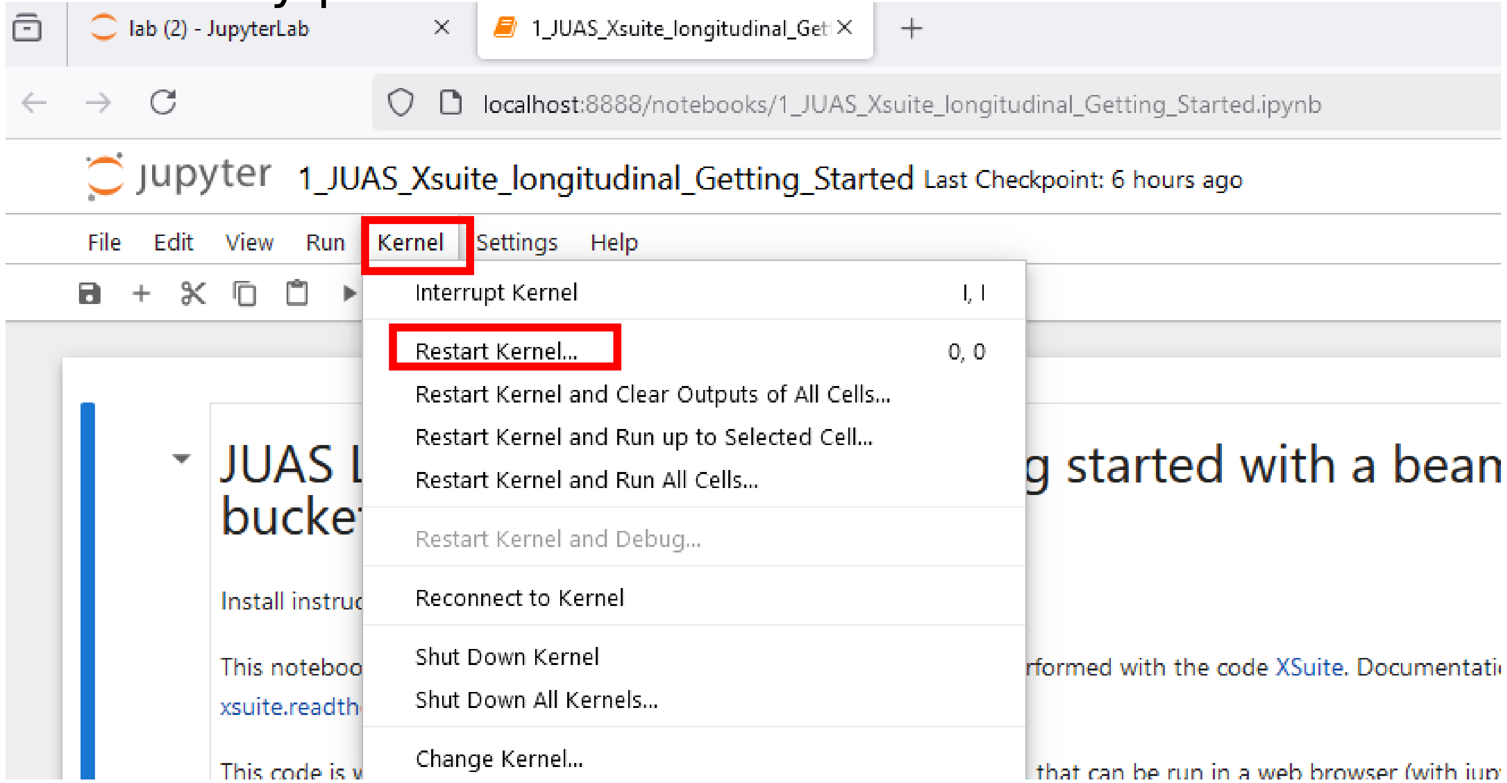
Plot average horizontal positions/intensity of the bunch turn by turn



```
[11]: fig, ax = plt.subplots(figsize=(8, 6))

      cmap = plt.get_cmap('jet')
      colors = cmap(np.linspace(0.9, 0.1, len(bunch_intensity_scan)))

      for i, bint in enumerate(reversed(bunch_intensity_scan)):
          file_path = f'./{results_folder}/bunchmonitor_bint_{bint:.2e}_bunches.h5'
          try:
              with h5py.File(file_path, 'r') as h5file:
                  if enable_losses:
                      signal = h5file['0']['num_particles'][:]
                  else:
                      signal = h5file['0']['mean_x'][:]
                      ax.set_ylim(-1e-2, 1e-2)
                  ax.plot(np.real(signal), color=colors[i], label=f'{bint:.2e}', alpha=0.8)
          except FileNotFoundError:
              print(f'File not found: {file_path}')
      ax.set_yscale('linear')
      ax.set_xlabel('Turn')
      ax.set_ylabel('Bunch intensity [p]' if enable_losses else 'Mean x position [m]')
      ax.legend(title='Bunch Intensity', fontsize='small', loc='best')
      fig.tight_layout()
      plt.savefig('turn_by_turn_data_all_bints.png')
      plt.show()
```

ipython cheat sheet

- To restart the kernel at any point: “Kernel” → “Restart”


lab (2) - JupyterLab
1_JUAS_Xsuite_longitudinal_Get
localhost:8888/notebooks/1_JUAS_Xsuite_longitudinal_Getting_Started.ipynb
jupyter
1_JUAS_Xsuite_longitudinal_Getting_Started Last Checkpoint: 6 hours ago
File
Edit
View
Run
Kernel
Settings
Help
Interrupt Kernel I, I
Restart Kernel... 0, 0
Restart Kernel and Clear Outputs of All Cells...
Restart Kernel and Run up to Selected Cell...
Restart Kernel and Run All Cells...
Restart Kernel and Debug...
Reconnect to Kernel
Shut Down Kernel
Shut Down All Kernels...
Change Kernel...

- To only interrupt the kernel: “Kernel” → “Interrupt”
- Shift-enter: execute current cell and move to next cell
- Ctrl-enter: execute current cell and stay on current cell