Follow-up on compressed convolution

Ivan Karpov

BLonD code development meeting, 12.03.2021

Motivation

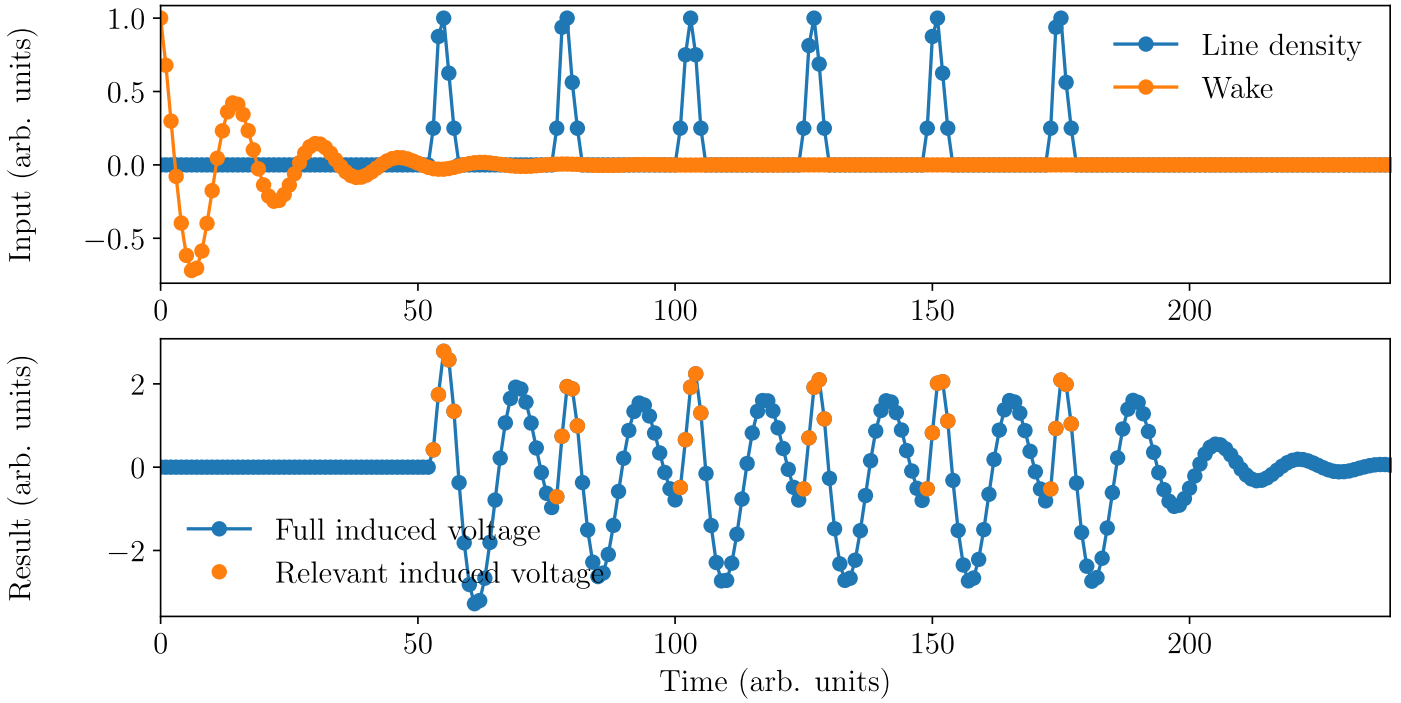
We aim to speed up simulations for beam with periodic gaps (LHC type in SPS, LHC, FCC, etc.)

→ Compressed wake calculation can be implemented (similarly to PyHEADTAIL J. Komppula, K. Li, & N. Mounet, PyHEADTAIL Meeting #19, 2018)

Method is based on:

- Picking relevant data and removing the rest

- Using fast FFT convolution



Motivation

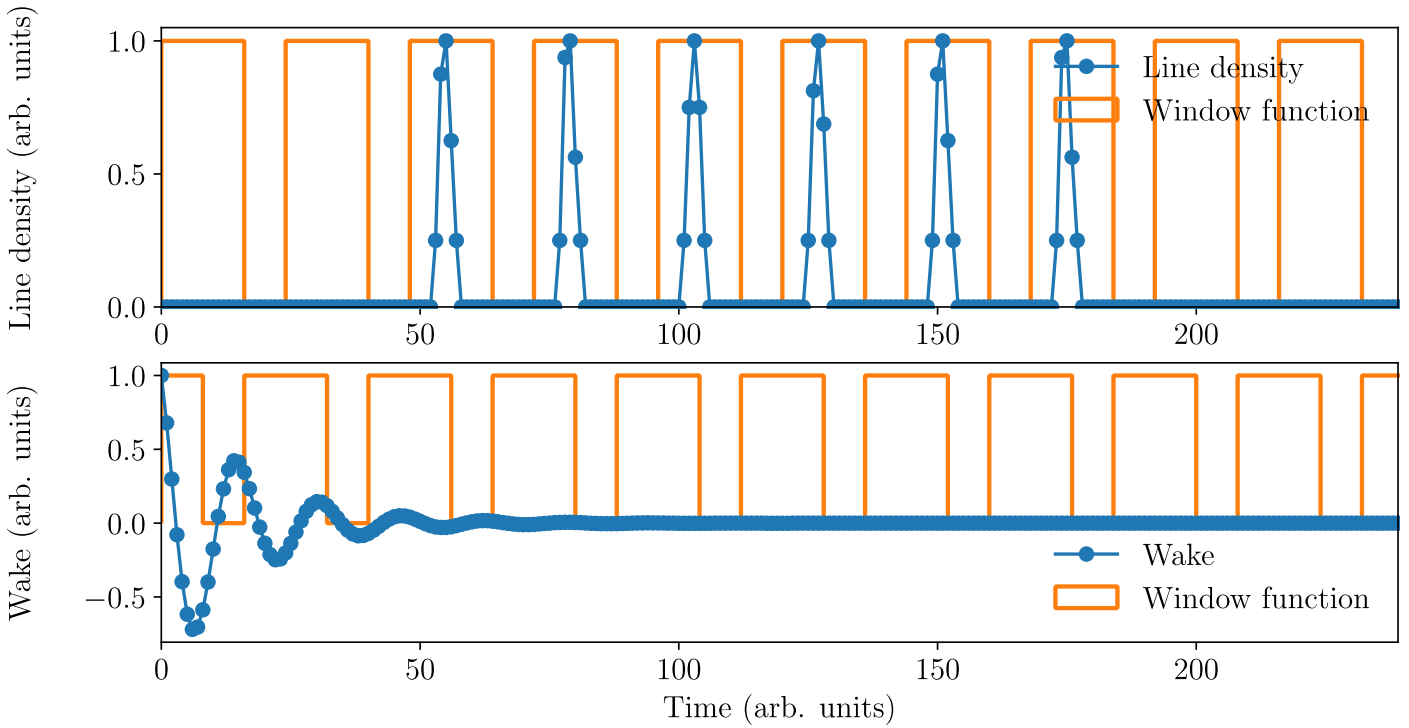
We aim to speed up simulations for beam with periodic gaps (LHC type in SPS, LHC, FCC, etc.)

→ Compressed wake calculation can be implemented (similarly to PyHEADTAIL J. Komppula, K. Li, & N. Mounet, PyHEADTAIL Meeting #19, 2018)

Method is based on:

- Picking relevant data and removing the rest

- Using fast FFT convolution



Motivation

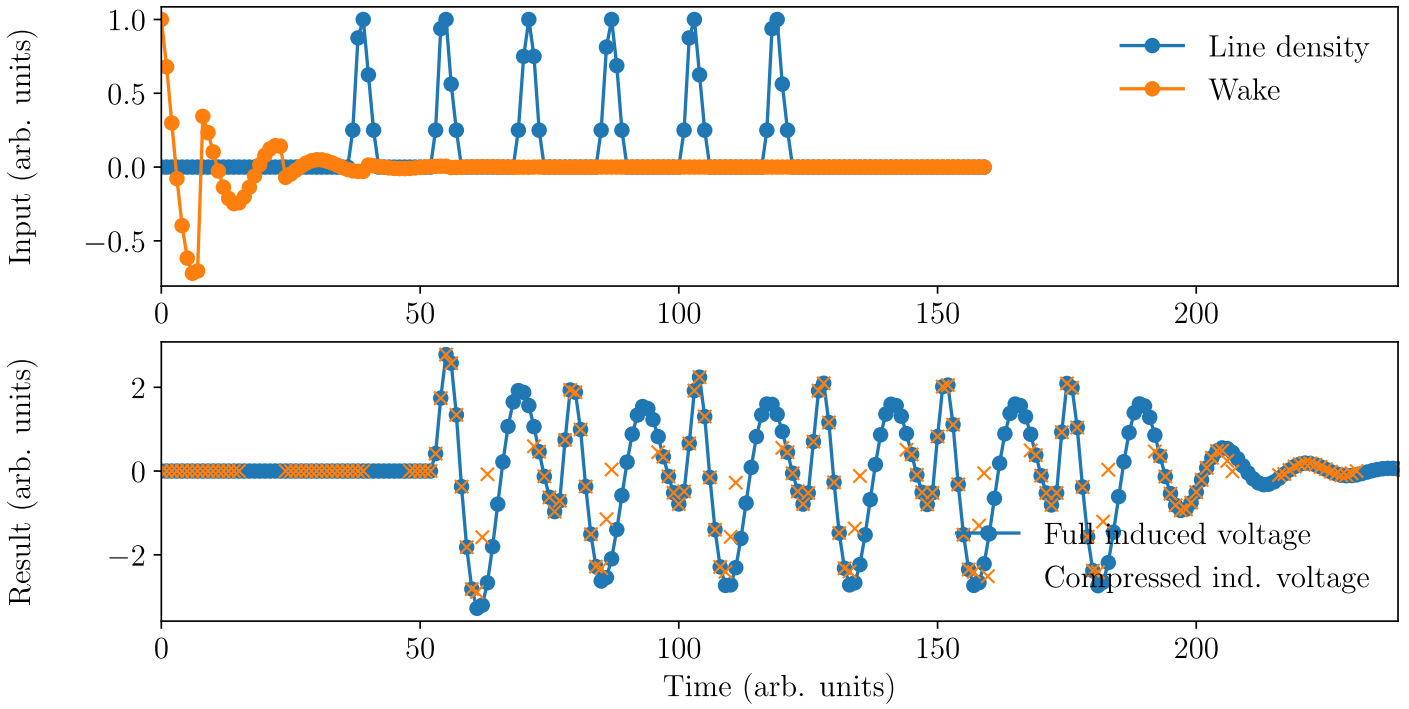
We aim to speed up simulations for beam with periodic gaps (LHC type in SPS, LHC, FCC, etc.)

→ Compressed wake calculation can be implemented (similarly to PyHEADTAIL J. Komppula, K. Li, & N. Mounet, PyHEADTAIL Meeting #19, 2018)

Method is based on:

- Picking relevant data and removing the rest

- Using fast FFT convolution



First implementation in BLonD*

Mostly InducedVoltageTime class in impedance.py is modified

- Requires additional info for masking ’n_window’, ‘n_sampling’; is activated by passing dictionary compression_dict

- Masks for profile and wake function are introduced in process()

- Method induced_voltage_1turn() is overridden; SciPy fftconvolve is used for speed-up

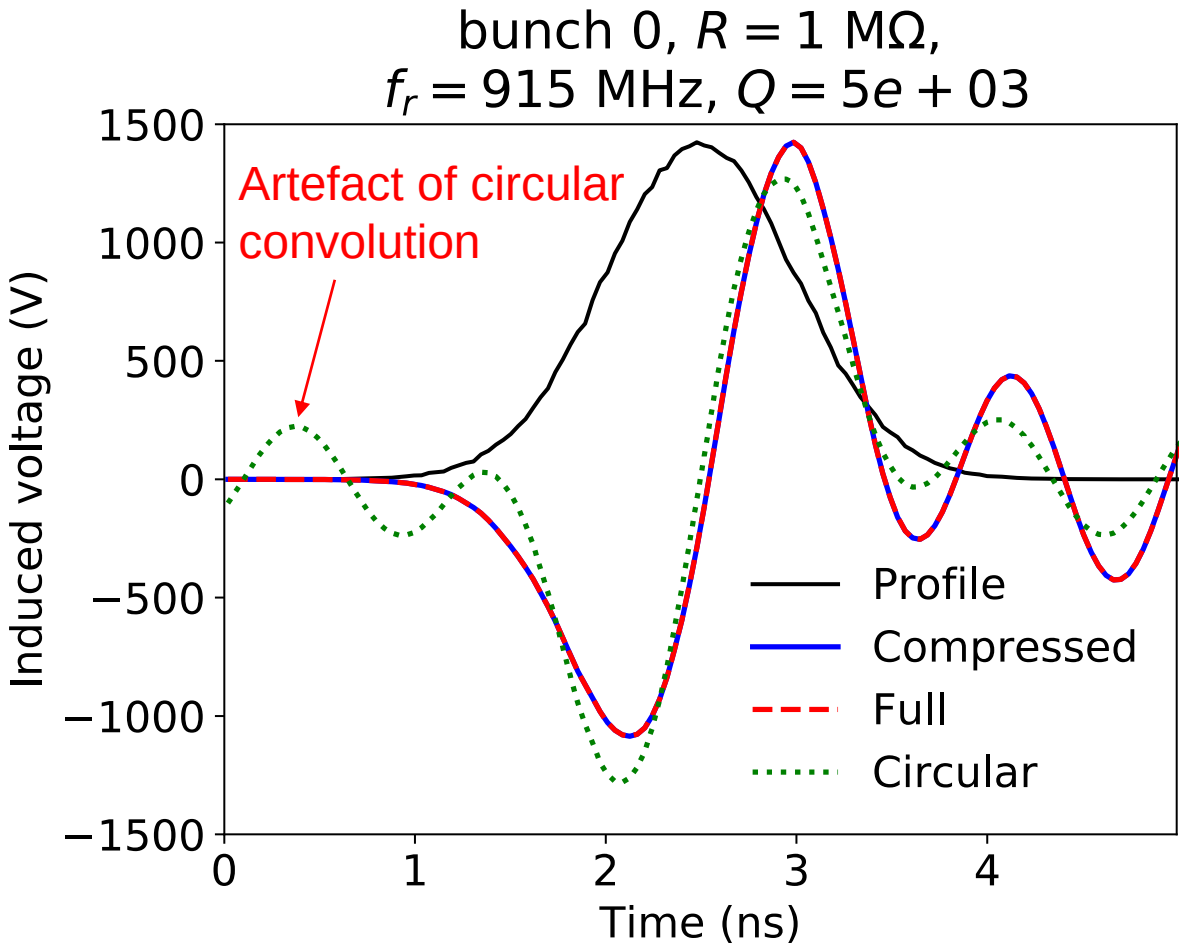
*More details in fork https://github.com/IEKarpov/BLonD

Example EX_23_compressed_wake.py

Simulation set-up:

- SPS flat bottom
- 200 bunches spaced by 100 ns (20 RF buckets)
- Single resonator impedance

Comparison of full, compressed and frequency domain calculations

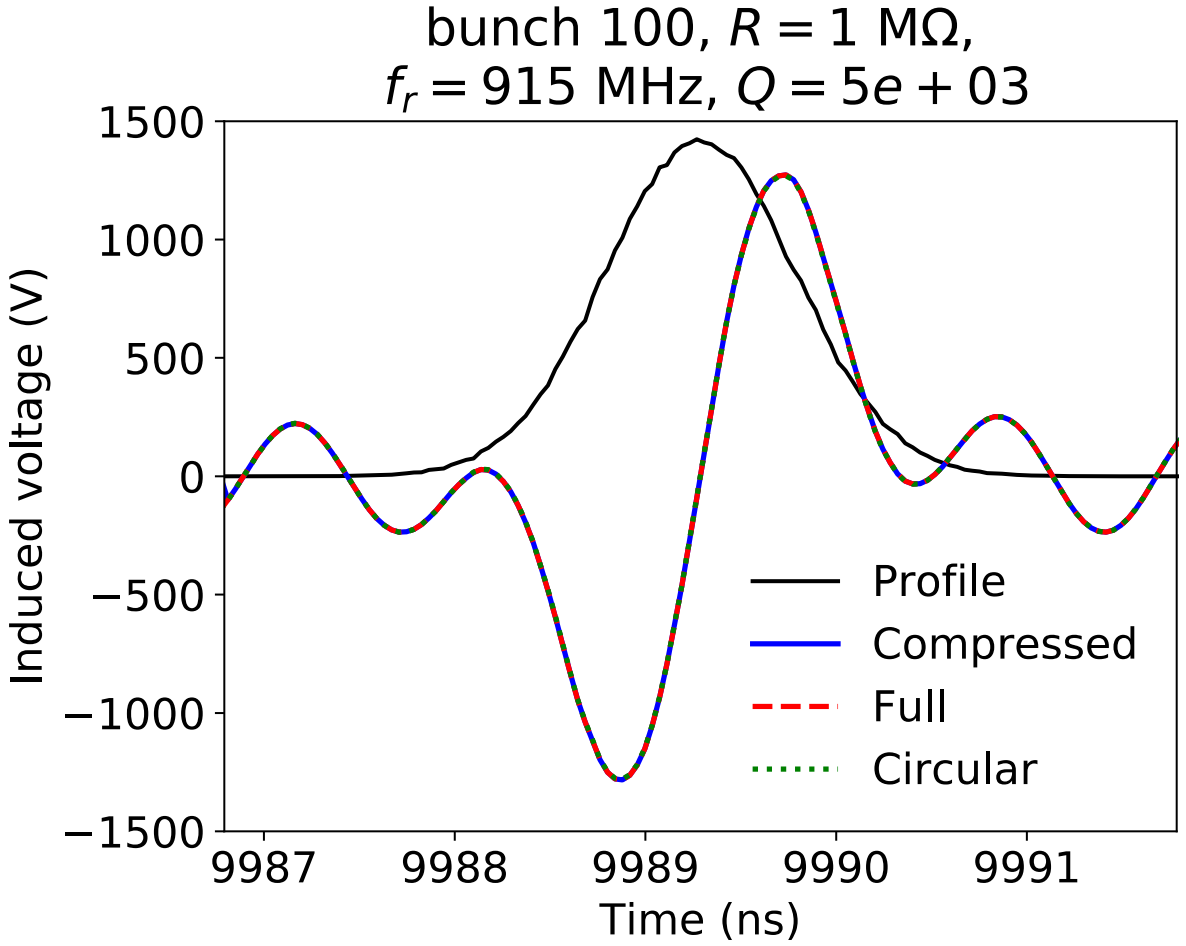


Example EX_23_compressed_wake.py

Simulation set-up:

- SPS flat bottom
- 200 bunches spaced by 100 ns (20 RF buckets)
- Single resonator impedance

Comparison of full, compressed and frequency domain calculations



Small performance test

induced_voltage_sum() is called 50 times
MEAN and STD values are calculated

200 bunches spaced by 100 ns (20 RF buckets), n_window = 2*n_slices_per_bucket

```
Compressed convolution: mean 9.483e-03 s, std 1.843e-03 s
Full convolution: mean 5.931e-02 s, std 3.375e-03 s
Frequency domain: mean 2.751e-02 s, std 1.562e-03 s
Speed-up vs full convolution 6.25
Speed-up vs frequency domain 2.90
```

Small performance test

induced_voltage_sum() is called 50 times
MEAN and STD values are calculated

Can be used for shorter bunches

200 bunches spaced by 100 ns (20 RF buckets), n_window = 1.5*n_slices_per_bucket

```
Compressed convolution: mean 9.103e-03 s, std 1.331e-03 s
Full convolution: mean 1.240e-01 s, std 3.327e-02 s
Frequency domain: mean 5.121e-02 s, std 1.153e-02 s
Speed-up vs full convolution 13.62
Speed-up vs frequency domain 5.62
```

Small performance test

induced_voltage_sum() is called 50 times
MEAN and STD values are calculated

500 bunches spaced by 25 ns (5 RF buckets), n_window = 1.5*n_slices_per_bucket

```
Compressed convolution: mean 1.375e-02 s, std 2.328e-03 s
Full convolution: mean 4.194e-02 s, std 2.280e-03 s
Frequency domain: mean 1.828e-02 s, std 1.129e-03 s
Speed-up vs full convolution 3.05
Speed-up vs frequency domain 1.33
```

Summary and outlook

- First version of compressed FFT convolution is implemented.
- Can be easily activated, but profile object need to be carefully generated.
- Factor of 6-13 speed-up is achieved for example case.
- Further optimization potentially can be done.

Thank you!

Smaller window function

n_window = 1.5*n_slices_per_bucket