Nuclear Science and Techniques (2026) 37:6
https://doi.org/10.1007/s41365-025-01814-0

# CUDA-based GPU-only computation for efficient tracking simulation of single and multi-bunch collective effects

**Keon Hee Kim**[1] · **Eun-San Kim**[1]

Received: 26 November 2024 / Revised: 24 March 2025 / Accepted: 22 April 2025 / Published online: 6 December 2025


**Abstract**
Beam-tracking simulations have been extensively utilized in the study of collective beam instabilities in circular accelerators. Traditionally, many simulation codes have relied on central processing unit (CPU)-based methods, tracking on a single CPU core, or parallelizing the computation across multiple cores via the message passing interface (MPI). Although these approaches work well for single-bunch tracking, scaling them to multiple bunches significantly increases the computational load, which often necessitates the use of a dedicated multi-CPU cluster. To address this challenge, alternative methods leveraging General-Purpose computing on Graphics Processing Units (GPGPU) have been proposed, enabling tracking studies on a standalone desktop personal computer (PC). However, frequent CPU-GPU interactions, including data transfers and synchronization operations during tracking, can introduce communication overheads, potentially reducing the overall effectiveness of GPU-based computations. In this study, we propose a novel approach that eliminates this overhead by performing the entire tracking simulation process exclusively on the GPU, thereby enabling the simultaneous processing of all bunches and their macro-particles. Specifically, we introduce MBTRACK2-CUDA, a Compute Unified Device Architecture (CUDA) ported version of MBTRACK2, which facilitates efficient tracking of single- and multi-bunch collective effects by leveraging the full GPU-resident computation.

**Keywords** Code development · GPU computing · Collective effects

## 1 Introduction

The rapid development and deployment of fourth-generation light sources (4GLS) around the world has led to an increasing focus on studying collective beam instabilities in storage and booster rings. These advanced facilities, operating at the forefront of accelerator technology, demand a thorough understanding and mitigation of instabilities to maintain the beam quality and achieve the desired performance. As a result, beam-tracking studies aimed at analyzing these instabilities are essential for supporting the design and operation of such high-performance synchrotron light sources.

To facilitate these studies, various computational tools have been developed, including MBTRACK2 [1–5] and a multi-bunch macro-particle tracking code developed at Synchrotron SOLEIL. Originally written in C as MBTRACK [6], the code was later redesigned in Python with an Object-Oriented Programming (OOP) structure–enhancing usability and extending functionality–and renamed MBTRACK2. MBTRACK2 retains the capability to study both single and multi-bunch collective beam instabilities, with numerous new tracking options added to accommodate the evolving needs of modern accelerator research.

MBTRACK2 utilizes CPU parallel processing with MPI to manage the computational demands of tracking multiple bunches. However, as the number of bunches increases, this approach requires a large number of CPU cores to handle the computational load. The need for such a large number of cores introduces significant challenges, including high costs and complexity of maintaining the necessary hardware

This work was supported by the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (MSIT) (No. RS-2022-00143178) and the Ministry of Education (MOE) (Nos. 2022R1A6A3A13053896 and 2022R1F1A1074616), Republic of Korea.

✉ Eun-San Kim
eskim1@korea.ac.kr

[1] Department of Accelerator Science, Korea University, Sejong 30019, Republic of Korea

infrastructure. These challenges present practical difficulties for several research facilities.

GPU acceleration has emerged as a viable alternative to address these challenges. GPUs designed for high levels of parallelism can significantly enhance the performance of large-scale simulations. A key advantage of GPU acceleration is that it can often be implemented on a single PC, simplifying infrastructure requirements compared with the extensive hardware setups needed for large-scale CPU parallel processing. Various GPU-accelerated tracking simulation programs have been developed to leverage this capability, including MBTRACK-CUDA [7], a GPU-accelerated version of MBTRACK; STABLE [8], a MATLAB-based code with GPU acceleration features; APES [9], which combines GPU and CPU processing with MPI for hybrid parallel acceleration; and GPU implementations for ELEGANT [10], PyHEADTAIL [11, 12], Xsuite [13] and SixTrackLib [14].

However, typical GPU acceleration methods can experience inefficiencies when handling extremely large-scale simulations, primarily because of frequent CPU-GPU data transfers that introduce both latency and synchronization delays. In most GPU-accelerated beam-tracking codes, statistical calculations and tracking-related operations rely on such data movement, leading to communication overhead and hindering the continuous execution of GPU kernels, which ultimately reduces computational efficiency.

This paper presents MBTRACK2-CUDA [15], which performs all the necessary computations entirely on GPU CUDA cores, eliminating the need for frequent data transfers and minimizing synchronization delays. By maintaining both statistical calculations and tracking operations on the GPU, this design avoids unnecessary CPU-GPU interactions, thereby reducing communication overhead and enhancing computational efficiency. As a result, MBTRACK2-CUDA is well suited for large-scale tracking simulations, where minimizing data movement and maintaining continuous GPU execution are critical for performance.

Rapid advancements in GPU technology have played a crucial role in enabling such improvements. Since the mid-2010 s, GPU-accelerated beam-tracking codes have been widely adopted, leading to substantial developments. As of the mid-2020 s, GPUs have undergone significant advancements, with Dual-ported Video Random-Access Memory (VRAM) capacities reaching tens of gigabytes per GPU, and a dramatic increase in the number of processing cores. For instance, the NVIDIA GeForce RTX 4090, which we used in this research, features 16,384 CUDA cores and 24 gigabytes (GB) of Graphics Double Data Rate 6X (GDDR6X) VRAM, showcasing the immense progress in GPU capabilities that support our enhanced simulation methods. While such advanced capabilities were not feasible in the mid-2010 s, the current generation of GPUs now provides the necessary power to implement our novel approach, which involves executing all stages of the tracking simulation directly on the GPU without requiring data transfer between the CPU and GPU.

The remainder of this paper is organized as follows. In Sect. 2, we describe the specific code architecture and the tracking model of MBTRACK2-CUDA. In Sect. 3, we present the validation of MBTRACK2-CUDA and the benchmarking results against MBTRACK2 to demonstrate the code's performance. We also analyze tracking simulations and results for single and multi-bunch cases, using the parameters of the High Energy Photon Source (HEPS) storage ring to study resistive-wall (RW) instability as an example of collective effects. This approach illustrates how MBTRACK2-CUDA can be utilized to investigate collective effects while noting that the resistive-wall study is based on assumptions regarding the beam pipe specifications, which may differ from those of the actual HEPS storage ring setup. Finally, Sect. 4 provides summary and outlook.

## 2 Code architecture and tracking model in MBTRACK2-CUDA

### 2.1 Code architecture

MBTRACK2-CUDA is an independent CUDA-ported version of MBTRACK2, designed to harness the power of GPU computing. It provides a comprehensive set of tracking simulation functionalities capable of handling a large number of bunches and macro-particles efficiently on a stand-alone desktop computer. Although MBTRACK2-CUDA is a partial port of MBTRACK2 and does not include all the features of the original code, it effectively supports the core tracking processes needed for studying collective effects.

MBTRACK2 itself is an evolving codebase with ongoing development and the continuous addition of new features. Similarly, MBTRACK2-CUDA is designed to evolve alongside MBTRACK2 the aim of incorporating new advancements and functionalities over time.

Figure 1 illustrates three different computational methods for tracking simulations, each utilizing distinct approaches to parallel processing. Figure 1a shows the traditional CPU parallel processing, where each bunch is assigned to a separate CPU core, making it highly efficient for multi-bunch calculations. However, because each core is responsible for processing all macro-particles within its assigned bunch, handling a large number of macro-particles places a significant computational load on all allocated cores. Additionally, the number of CPU cores must be sufficient to cover all bunches, which can be a limiting factor. Depending on the code, if the CPU cores cannot cover all the bunches, the calculations may either

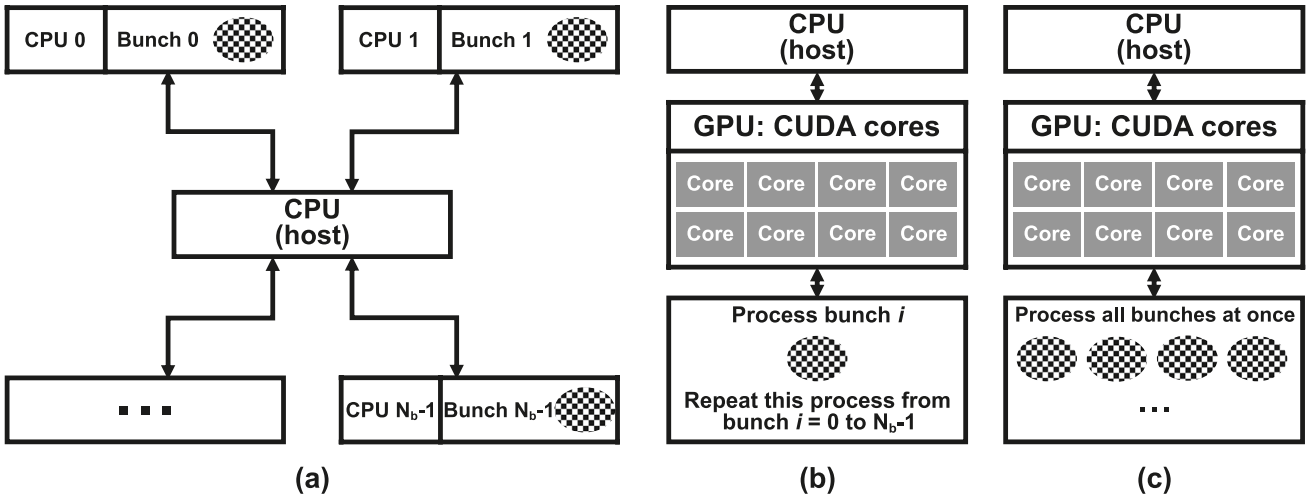


**Fig. 1** Comparisons of parallel computing methods: **a** Traditional CPU-based parallel processing using MPI. **b** GPU-accelerated processing by allocating one bunch at a time, as used in MBTRACK-CUDA. **c** GPU-accelerated processing by allocating all bunches simultaneously, as used in MBTRACK2-CUDA

not proceed or the remaining bunches will have to wait until the current bunches are processed, leading to potential inefficiencies.

Depending on the code, the GPU-accelerated method can be implemented in one of two ways. In the first approach, the GPU acceleration method used in MBTRACK-CUDA is shown in Fig. 1b, which assigns all bunches to the GPU memory but processes only one bunch at a time, utilizing multiple GPU cores to handle macro-particles in parallel within each bunch. Although this method allows for rapid computation of individual bunches, it requires sequential processing for multiple bunches, which may limit the overall performance in large-scale multi-bunch simulations.

In contrast, the approach shown in Fig. 1c, which was adopted in MBTRACK2-CUDA, enables fully parallel computations by processing all bunches and their respective macro-particles simultaneously on the GPU. This method allows for a more efficient utilization of GPU resources, significantly improving the simulation performance, particularly in large-scale scenarios with many bunches and macro-particles.

A flowchart of MBTRACK2-CUDA is presented in Fig. 2, which outlines the simulation process. The process begins with the CPU, similar to MBTRACK2, to generate bunches based on the machine parameters. However, once the bunch data were prepared, the entire tracking process was handed over to the GPU. From this point onward, the GPU handles all tracking computations, eliminating the need for further CPU-GPU data transfers and minimizing the CPU-GPU interactions. This design avoids the communication overhead and synchronization delays associated with frequent data exchanges, ensuring a more efficient simulation process.

When using MBTRACK2-CUDA for tracking simulations, it is important to ensure that the VRAM capacity of the GPU is not exceeded. If the GPU's VRAM is surpassed, the code will resort to using CPU RAM, which can significantly degrade the performance. Therefore, it is advisable

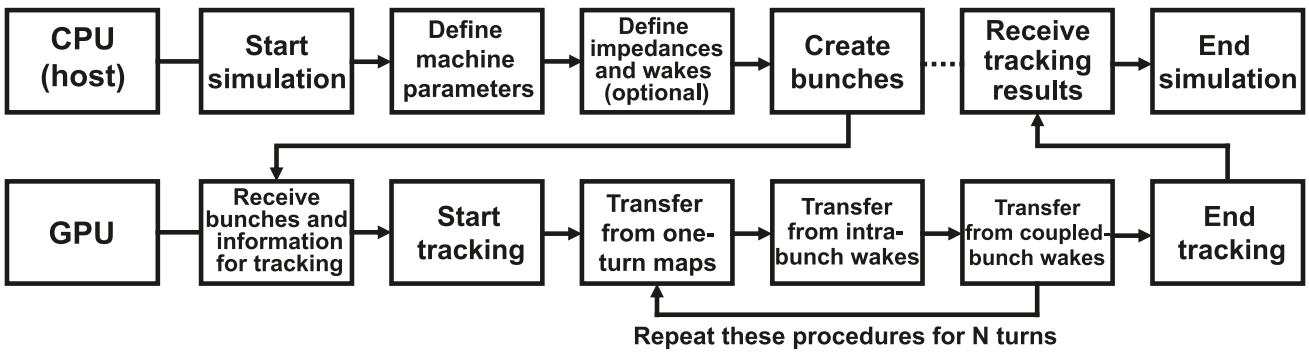


**Fig. 2** Flowchart of MBTRACK2-CUDA

to select a model with sufficient GPU memory for tracking simulations.

Moreover, MBTRACK2-CUDA was designed to minimize overhead by performing all computations on a single GPU, and thus did not support multi-GPU setups. However, as of the mid-2020 s, high-performance GPUs with ample memory for tracking simulations and a dramatically increased number of CUDA cores per GPU were released. With continued advancements in GPU technology, this limitation is becoming less of a concern as performance improvements continue to grow.

To implement the CUDA-based computation in MBTRACK2-CUDA, we utilized the Numba-CUDA library [16], which supports CUDA programming within the Python environment. Numba-CUDA is a powerful tool that enables high-performance computations by compiling Python code for GPU execution. It simplifies CUDA programming by allowing developers to write CUDA kernels directly in Python, thereby eliminating the need to switch to CUDA C/C++. This seamless integration with the Python ecosystem supports efficient GPU acceleration in Python-based workflow. In addition, Numba-CUDA manages memory allocation and optimization, offering fine-grained control over CUDA kernels, which enhances the efficiency of GPU-accelerated applications.

In MBTRACK2-CUDA, the entire tracking simulation process is implemented using multiple CUDA kernels. Each kernel handles specific aspects of the tracking simulation, such as transforming basic one-turn maps, computing collective effects, and managing macro-particles interactions. These kernels are executed sequentially in a turn-by-turn manner through a single CUDA stream, ensuring a coherent tracking progression.

## 2.2 Beam-tracking model

### 2.2.1 Basic one-turn maps for 6D phase space transformations

The bunches we aim to track in MBTRACK2-CUDA are composed of macro-particles, each represented in a 6-dimensional (6D) phase space by the coordinates:

$$\left[x_n, x_n', y_n, y_n', \tau_n, \delta_n\right]_{i,j},$$

where $n$ denotes the turn index, $i$ represents the bunch index, and $j$ is the index of each macro-particle within the $i$-th bunch. In this coordinate system, $x$ and $y$ correspond to the horizontal and vertical positions, respectively, while $\tau$ represents the longitudinal temporal position. The coordinates $x'$ and $y'$ indicate the transverse momenta and $\delta$ is the relative energy deviation.

The basic one-turn maps consist of a longitudinal map, transverse map, radio frequency (RF) cavity, and radiation damping with quantum excitation. These maps are calculated turn-by-turn (once per turn), and we present the formulae describing the coordinate transformations of the $j$-th single particle within the $i$-th bunch at each turn for the ultra-relativistic case.

We begin with a longitudinal map, where the longitudinal coordinates are updated as follows:

$$\tau_{n+1} = \tau_n + \alpha_c T_0 \delta_n, \tag{1}$$

$$\delta_{n+1} = \delta_n - \frac{U_0}{E_0}, \tag{2}$$

where $\alpha_c$ denotes the momentum compaction factor, $T_0$ denotes the revolution time, $U_0$ denotes the average energy loss per turn, and $E_0$ denotes the reference energy.

In the transverse map, we can write the coordinate transport in matrix form as follows:

$$\begin{bmatrix} u \\ u' \\ \delta \end{bmatrix}_{n+1} = \begin{bmatrix} \cos\psi_u + \alpha_u^{\mathrm{loc}} \sin\psi_u & \beta_u^{\mathrm{loc}} \sin\psi_u & D_u \\ -\gamma_u^{\mathrm{loc}} \sin\psi_u & \cos\psi_u - \alpha_u^{\mathrm{loc}} \sin\psi_u & D_{u'} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ u' \\ \delta \end{bmatrix}_n \tag{3}$$

with $u$ being either $x$ or $y$, $\psi_u$ representing the phase advance, $D_{u,u'}$ denoting the dispersion, and $\alpha_u^{\mathrm{loc}}$, $\beta_u^{\mathrm{loc}}$, $\gamma_u^{\mathrm{loc}}$ being the Courant–Snyder parameters at the tracking location. If we ignore the amplitude-dependent tune shift, the phase advance $\psi_u$ can be expressed as

$$\psi_u = 2\pi\left(\nu_u + \xi_u \delta\right), \tag{4}$$

where $\nu_u$ is the betatron tune and $\xi_u$ is the chromaticity.

In the longitudinal map, the energy gain from the RF cavities is not considered. To compensate for the energy loss caused by synchrotron radiation, we must calculate the energy gain provided by the RF cavities. If only the main RF cavity is installed, the relative energy deviation is updated as

$$\delta_{n+1} = \delta_n + \frac{e}{E_0} V_{\mathrm{RF}} \cos\left(\omega_{\mathrm{RF}} \tau_n + \phi_s\right), \tag{5}$$

where $e$ denotes the elementary charge, $V_{\mathrm{RF}}$ denotes the peak voltage of the main RF cavity, $\omega_{\mathrm{RF}}$ denotes the angular frequency of the main RF cavity, and $\phi_s$ denotes the synchronous phase. Here, the synchronous phase can be obtained as

$$\phi_s = \arccos\left(\frac{U_0}{eV_{\mathrm{RF}}}\right). \tag{6}$$

It is noteworthy that the right-hand side of Eq. (2) was incorporated into the $\delta_n$ term in Eq. (5) when both the energy loss and energy gain are considered. MBTRACK2-CUDA can also account for the effects of a harmonic cavity. In a

double RF system, considering both the main RF cavity and a single active harmonic cavity, the relative energy deviation is updated as

$$\delta_{n+1} = \delta_n + \frac{e}{E_0} V_{\mathrm{RF}} \big[\cos\left(\omega_{\mathrm{RF}}\tau_n + \phi_{\mathrm{s}}\right) + A\cos\left(m\omega_{\mathrm{RF}}\tau_n + \phi_{\mathrm{h}}\right)\big], \tag{7}$$

where $A$ is the amplitude factor of the active harmonic cavity, $m$ is the cavity harmonic number, and $\phi_{\mathrm{h}}$ is the synchronous phase of the harmonic cavity. Note that the beam-loading effect was not considered in this model. For a given value $m$, we can calculate the amplitude factor $A$ and the phases $\phi_{\mathrm{s}}$ and $\phi_{\mathrm{h}}$ by applying flat-potential conditions at the synchronous phase as follows [6, 17]:

$$A = \sqrt{\frac{1}{m^2} - \frac{U_0^2}{e^2 V_{\mathrm{RF}}^2}\frac{1}{m^2-1}}, \tag{8}$$

$$\phi_s = \arccos\left(\frac{m^2}{m^2-1}\frac{U_0}{eV_{\mathrm{RF}}}\right), \tag{9}$$

$$\begin{aligned}\phi_h &= \arctan\left[\frac{\sqrt{\left(m^2-1\right)^2 - \left(m^2\frac{U_0}{eV_{\mathrm{RF}}}\right)^2}}{mU_0/\left(eV_{\mathrm{RF}}\right)}\right] - \pi \\ &= -\arccos\left[-\frac{U_0}{AeV_{\mathrm{RF}}}\frac{1}{m^2-1}\right],\end{aligned} \tag{10}$$

where it is crucial to satisfy the condition $0 < \phi_{\mathrm{s}} < \frac{\pi}{2}$.

As the final step of the basic one-turn maps, the radiation damping with quantum excitation is considered as follows [2]:

$$\tilde{\delta}_{n+1} = \left(1 - \frac{2T_0}{\tau_z}\right)\delta_{n+1} + 2\sigma_\delta\sqrt{\frac{T_0}{\tau_z}}\delta_{\mathrm{rand}}, \tag{11}$$

$$\tilde{u'}_{n+1} = \left(1 - \frac{2T_0}{\tau_u}\right)u'_{n+1} + 2\sigma_{u'}\sqrt{\frac{T_0}{\tau_u}}u'_{\mathrm{rand}}, \tag{12}$$

where $\tau_{u,z}$ is the synchrotron radiation damping time, which should not be confused with the longitudinal temporal position; $\sigma_\delta$ is the energy spread; and $\sigma_{u'}$ is the standard deviation of the transverse momentum. Terms $\delta_{\mathrm{rand}}$ and $u'_{\mathrm{rand}}$ are random numbers generated from a Gaussian distribution with a unit standard deviation. For random number generation (RNG), MBTRACK2-CUDA uses Numba-CUDA's xoroshiro128+ algorithm [18] on the GPU, while MBTRACK2 uses NumPy's Mersenne Twister (MT19937) algorithm [19] on the CPU.

### 2.2.2 Wakefields of resistive-wall

After establishing the basic one-turn maps, we then incorporate the wakefield effects arising from collective interactions. In this study, we explore the effects of resistive-wall as an example, which is one of the most impactful types of wakefields in synchrotrons. To this end, we introduce the simplest model for a resistive wall: the circular beam-pipe model.

The resistive wall impedances per unit length for the longitudinal and transverse planes in a single-layered infinitely thick circular beam pipe of radius $b$ with direct current (DC) conductivity $\sigma_{\mathrm{c}}$ are given for an ultrarelativistic beam as follows [20, 21]:

$$\frac{Z_{\mathrm{RW}}^{\parallel}(\kappa)}{L} = \frac{Z_0 c\tau_0}{2\pi b^2}\left(\frac{1+i}{\sqrt{\kappa}} - i\frac{\kappa}{2}\right)^{-1}, \tag{13}$$

$$\frac{Z_{\mathrm{RW}}^{\perp}(\kappa)}{L} = \frac{Z_0 c^2\tau_0^2}{\pi b^4\kappa}\left(\frac{1+i}{\sqrt{\kappa}} - i\frac{\kappa}{2}\right)^{-1}, \tag{14}$$

where $L$ is the length of the wake component, $Z_0$ is the impedance of free space, $c$ is the speed of light, $\tau_0$ is the characteristic temporal distance express as

$$\tau_0 = \frac{1}{c}\left(\frac{2b^2}{Z_0\sigma_{\mathrm{c}}}\right)^{1/3}, \tag{15}$$

and $\kappa = kc\tau_0$, where $k$ denotes the angular wavenumber. It is noteworthy that only the dipole impedance is considered for the transverse impedance in a circular pipe geometry.

From Eqs. (13) and (14), the resistive wall wake functions for the longitudinal and transverse planes can be obtained by performing inverse Fourier transform [22]. The classic resistive wall wake function formulae include integral terms known as diffusion terms [23], which have traditionally been calculated using numerical integration. However, Ivanyan and Tsakanov [24] solved these integrals using the Faddeeva function. They also derived a series of expansion

formulas. The wake function formulae discussed thus far are as follows:

$$\begin{aligned} W_{\mathrm{RW}}^{\parallel}(\tau) &= \frac{4Z_0cL}{\pi b^2}\left[\frac{1}{3}e^{-\tau/\tau_0}\cos\frac{\sqrt{3}\tau}{\tau_0} - \frac{\sqrt{2}}{\pi}\int_0^\infty \frac{x^2 e^{-x^2\tau/\tau_0}}{x^6+8}\,\mathrm{d}x\right] \\ &= \frac{Z_0cL}{3\pi b^2}\left[4e^{-\tau/\tau_0}\cos\frac{\sqrt{3}\tau}{\tau_0} + w\left(i\sqrt{\frac{2\tau}{\tau_0}}\right) - w\left(e^{i\pi/6}\sqrt{\frac{2\tau}{\tau_0}}\right) - w\left(-e^{-i\pi/6}\sqrt{\frac{2\tau}{\tau_0}}\right)\right] \\ &= \frac{Z_0cL}{\pi b^2}\left[\sum_{k=0}^{\infty}\frac{2^{3k}}{(3k)!}\left(\frac{\tau}{\tau_0}\right)^{3k} - \sqrt{\frac{2}{\pi}}\sum_{k=1}^{\infty}\frac{2^{6k-3}}{(6k-3)!!}\left(\frac{\tau}{\tau_0}\right)^{3k-3/2}\right], \end{aligned} \tag{16}$$

$$\begin{aligned} W_{\mathrm{RW}}^{\perp}(\tau) &= \frac{8Z_0c^2\tau_0L}{\pi b^4}\left\{\frac{1}{12}e^{-\tau/\tau_0}\left[\sqrt{3}\sin\frac{\sqrt{3}\tau}{\tau_0} - \cos\frac{\sqrt{3}\tau}{\tau_0}\right] + \frac{\sqrt{2}}{\pi}\int_0^\infty \frac{e^{-x^2\tau/\tau_0}}{x^6+8}\,\mathrm{d}x\right\} \\ &= \frac{Z_0c^2\tau_0L}{3\pi b^4}\left\{2e^{-\tau/\tau_0}\left[\sqrt{3}\sin\frac{\sqrt{3}\tau}{\tau_0} - \cos\frac{\sqrt{3}\tau}{\tau_0}\right] + w\left(i\sqrt{\frac{2\tau}{\tau_0}}\right)\right. \\ &\quad \left. + e^{-i\pi/3}w\left(e^{i\pi/6}\sqrt{\frac{2\tau}{\tau_0}}\right) + e^{i\pi/3}w\left(-e^{-i\pi/6}\sqrt{\frac{2\tau}{\tau_0}}\right)\right\} \\ &= \frac{Z_0c^2\tau_0L}{\pi b^4}\left[\sum_{k=0}^{\infty}\frac{2^{3k+1}}{(3k+1)!}\left(\frac{\tau}{\tau_0}\right)^{3k+1} - \sqrt{\frac{2}{\pi}}\sum_{k=1}^{\infty}\frac{2^{6k-1}}{(6k-1)!!}\left(\frac{\tau}{\tau_0}\right)^{3k-1/2}\right] \end{aligned} \tag{17}$$

with $w$ being the Faddeeva function defined as

$$w(z) := e^{-z^2}\,\mathrm{erfc}\,(-iz). \tag{18}$$

For the long-range regime where $\tau \gg \tau_0$, Eqs. (16) and (17) can be reduced to asymptotic formulae. From an impedance perspective, this involves focusing on low-frequency behavior while neglecting the high-frequency term. Consequently, the second term in Eqs. (13) and (14) can be omitted. By applying an inverse Fourier transform to these low-frequency expressions with regularization [25, 26], we obtain the asymptotic wake function formula as follows:

$$W_{\mathrm{RW}}^{\parallel}(\tau) \approx -\frac{L}{4\pi b}\sqrt{\frac{Z_0}{\pi c\sigma_{\mathrm{c}}}}\frac{1}{\sqrt{\tau^3}}, \tag{19}$$

$$W_{\mathrm{RW}}^{\perp}(\tau) \approx \frac{L}{\pi b^3}\sqrt{\frac{Z_0c}{\pi\sigma_{\mathrm{c}}}}\frac{1}{\sqrt{\tau}}. \tag{20}$$

### 2.2.3 Calculating the influence of intra-bunch wakefields

A wake function is a normalized potential defined within a two-particle model, which represents the wake kick exerted by a source charge on a target charge. To compute the instabilities within a single bunch using wake functions, the model must be extended to account for a given bunch distribution. The normalized potential in the presence of a given bunch distribution is referred to as wake potential. The wake potentials in the longitudinal and transverse planes are defined as the convolutions of the longitudinal bunch distribution with wake functions, as follows [27]:

$$W_{\mathrm{p}}^{\parallel}(\tau) = \int_0^\infty \lambda(\tau-\tau')W^{\parallel}(\tau')\,\mathrm{d}\tau', \tag{21}$$

$$W_{\mathrm{p}}^{\perp}(\tau) = \int_0^\infty \langle u_{bin}\rangle(\tau-\tau')\lambda(\tau-\tau')W^{\perp}(\tau')\,\mathrm{d}\tau', \tag{22}$$

where the longitudinal bunch distribution $\lambda$ is normalized as

$$\int_{-\infty}^{\infty}\lambda(\tau)\,\mathrm{d}\tau = 1, \tag{23}$$

and $\langle u_{\mathrm{bin}}\rangle$ represents the average transverse displacement, known as the dipole moment, of macro-particles in an infinitesimal slice. The lower bound of each integral in Eqs. (21) and (22) was set to zero because the wake function of an ultra-relativistic particle vanishes ahead of it owing to the principle of causality [20, 27]. Note that we considered only the dipole wake and neglected the quadrupole wake for the transverse plane in this context.

In practical computations, continuous convolution must be converted into discrete convolution. To perform these discrete calculations, we sliced the given bunch

distribution into $M$ bins, each with a width of $\Delta\tau$ in the $\tau$ domain. For the $m$th bin, let $\tau_{\min}$ and $\tau_{\max}$ denote the minimum and maximum $\tau$ values, respectively. The average $\tau$ value for the $m$th bin is given by

$$\tau_{\text{bin}}[m] = \frac{\tau_{\min}[m] + \tau_{\max}[m]}{2}. \tag{24}$$

Because the theoretical equations are treated as discrete calculations in the actual simulations, the discrete convolutions for Eqs. (21) and (22) become:

$$W_{\text{p}}^{\parallel}\left(\tau_{\text{bin}}[m]\right) = \sum_{m'=0}^{M-1} \lambda\left[m-m'\right] W^{\parallel}\left[m'\right] \Delta\tau, \tag{25}$$

$$\begin{aligned} W_{\text{p}}^{\perp}\left(\tau_{\text{bin}}[m]\right) = \sum_{m'=0}^{M-1} &\langle u_{\text{bin}}\rangle \left[m-m'\right] \lambda\left[m-m'\right] \\ &\times W^{\perp}\left[m'\right] \Delta\tau, \end{aligned} \tag{26}$$

where for $m - m' < 0$, zero-padding is applied to ensure that $\lambda\left[m-m'\right]$ and $\langle u_{\text{bin}}\rangle \left[m-m'\right]$ are set to zero, as there are no macro-particles in this region.

Both MBTRACK2 and MBTRACK2-CUDA implemented a dynamic binning scheme that adjusted the bin width according to the bunch length while maintaining a fixed total number of bins. To obtain a reliable longitudinal bunch distribution, both codes perform a first binning with approximately 100 bins, where the distribution is determined by counting the number of macro-particles in each bin. However, because wake functions exhibit nonlinear characteristics on a smaller scale, the resolution provided by the first binning alone is not sufficient for accurate convolution. Therefore, a second binning process was carried out through linear interpolation applied to the longitudinal bunch distribution, thereby increasing the total number of bins.

In this way, the final bin width resolution is determined by the number of interpolated bins rather than the initial bin count. This interpolation ensured that the wake functions could be approximated as linear within each bin. Thus, the number of interpolated bins should be determined by the bunch length and wake function characteristics. In general, MBTRACK2 requires $10^4 \sim 10^5$ interpolated bins, which can lead to heavy convolution calculations. Because convolution is inherently a series operation, and such heavy computations are not feasible on a GPU, an alternative method was adopted in MBTRACK2-CUDA to reduce the computational load. This method first calculates the mean of the wake functions within each interpolated bin as follows:

$$\begin{aligned} &\bar{W}^{\parallel}\left(\tau_{\text{bin}}[m]\right) \\ &= \frac{\int_{\tau_{\min}[m]}^{\tau_{\max}[m]} W^{\parallel}\, \mathrm{d}\tau}{\tau_{\max}[m] - \tau_{\min}[m]}, \end{aligned} \tag{27}$$

$$\begin{aligned} &\bar{W}^{\perp}\left(\tau_{\text{bin}}[m]\right) \\ &= \frac{\int_{\tau_{\min}[m]}^{\tau_{\max}[m]} W^{\perp}\, \mathrm{d}\tau}{\tau_{\max}[m] - \tau_{\min}[m]}. \end{aligned} \tag{28}$$

Next, we substitute Eqs. (27) and (28) instead of the wake functions on the right-hand side of Eqs. (25) and (26). Although this reduction technique also assumes that the wake function and bunch distribution are approximately linear within each interpolated bin, it can still yield more accurate results than directly sampling the function at bin centers when some degree of nonlinearity exists. With this method, even in cases where MBTRACK2 requires convolution calculations of the order of $10^4 \sim 10^5$ interpolated bins, MBTRACK2-CUDA can achieve equivalent results with only an order of $10^3$ interpolated bins. This reduced computational load makes it feasible to quickly perform convolution operations on a GPU. Additionally, the fundamental theorem of beam loading [27] was automatically considered when calculating the averages.

Finally, from the wake potentials, we calculated the effects of the longitudinal and transverse kicks on single-bunch instabilities. An extra interpolation step during the kick calculation further refines this process by assigning a unique kick to each macro-particles. The changes in the relative energy deviation and transverse momenta due to these kicks are given as

$$\Delta\delta = -\frac{QeW_{\text{p}}^{\parallel}(\tau)}{E_0}, \tag{29}$$

$$\Delta u' = \frac{QeW_{\text{p}}^{\perp}(\tau)}{E_0} \frac{\beta_u^{\text{avg}}}{\beta_u^{\text{loc}}}, \tag{30}$$

where $Q$ is the total charge of a given bunch and $\beta_u^{\text{avg}}$ is the average value of the beta functions from each wake component. In practice, $W_{\text{p}}^{\parallel}(\tau)$ and $W_{\text{p}}^{\perp}(\tau)$ can be obtained by interpolating Eqs. (25) and (26). The negative sign on the right-hand side of Eq. (29) arises because a particle traversing a passive structure cannot gain energy from its field [27]. Because we define the wake function such that $W^{\parallel}(+0) > 0$, this negative sign must be applied accordingly.

For resistive wall wakefields, the means of the wake functions within each interpolated bin can be expressed analytically. These are derived by substituting Eqs. (16) and (17) into Eq. (27) and (28). The resulting expressions are as follows.

$$
\begin{aligned}
&\bar{W}^{\parallel}_{\mathrm{RW}}\left(\tau_{\mathrm{bin}}[m]\right) \\
&= \frac{Z_0 c \tau_0 L}{6\pi b^2(\tau_{\max}[m]-\tau_{\min}[m])}\left\{2e^{-\tau/\tau_0}\left[\sqrt{3}\sin\frac{\sqrt{3}\tau}{\tau_0}-\cos\frac{\sqrt{3}\tau}{\tau_0}\right]+w\left(i\sqrt{\frac{2\tau}{\tau_0}}\right)\right. \\
&\quad \left.+e^{-i\pi/3}w\left(e^{i\pi/6}\sqrt{\frac{2\tau}{\tau_0}}\right)+e^{i\pi/3}w\left(-e^{-i\pi/6}\sqrt{\frac{2\tau}{\tau_0}}\right)\right\}_{\tau=\tau_{\min}[m]}^{\tau=\tau_{\max}[m]} \\
&= \frac{Z_0 c \tau_0 L}{\pi b^2(\tau_{\max}[m]-\tau_{\min}[m])}\left[\sum_{k=0}^{\infty}\frac{2^{3k}}{(3k+1)!}\left(\frac{\tau}{\tau_0}\right)^{3k+1}-\sqrt{\frac{2}{\pi}}\sum_{k=1}^{\infty}\frac{2^{6k-2}}{(6k-1)!!}\left(\frac{\tau}{\tau_0}\right)^{3k-1/2}\right]_{\tau=\tau_{\min}[m]}^{\tau=\tau_{\max}[m]},
\end{aligned}
\tag{31}
$$

$$
\begin{aligned}
\bar{W}^{\perp}_{\mathrm{RW}}\left(\tau_{\mathrm{bin}}[m]\right) &= \frac{Z_0 c^2 \tau_0^2 L}{6\pi b^4(\tau_{\max}[m]-\tau_{\min}[m])}\left\{-2e^{-\tau/\tau_0}\left[\sqrt{3}\sin\frac{\sqrt{3}\tau}{\tau_0}+\cos\frac{\sqrt{3}\tau}{\tau_0}\right]+w\left(i\sqrt{\frac{2\tau}{\tau_0}}\right)\right. \\
&\quad \left.-e^{-i2\pi/3}w\left(e^{i\pi/6}\sqrt{\frac{2\tau}{\tau_0}}\right)-e^{i2\pi/3}w\left(-e^{-i\pi/6}\sqrt{\frac{2\tau}{\tau_0}}\right)+6\sqrt{\frac{2\tau}{\pi\tau_0}}\right\}_{\tau=\tau_{\min}[m]}^{\tau=\tau_{\max}[m]} \\
&= \frac{Z_0 c^2 \tau_0^2 L}{\pi b^4(\tau_{\max}[m]-\tau_{\min}[m])}\left[\sum_{k=0}^{\infty}\frac{2^{3k+1}}{(3k+2)!}\left(\frac{\tau}{\tau_0}\right)^{3k+2}-\sqrt{\frac{2}{\pi}}\sum_{k=1}^{\infty}\frac{2^{6k}}{(6k+1)!!}\left(\frac{\tau}{\tau_0}\right)^{3k+1/2}\right]_{\tau=\tau_{\min}[m]}^{\tau=\tau_{\max}[m]}.
\end{aligned}
\tag{32}
$$

Both expressions using the Faddeeva function $w$ and those based on series expansion are implemented as built-in functions in MBTRACK2-CUDA.

Special functions such as the Faddeeva function are challenging to compute within a GPU kernel. To handle this, we ported the CERN library's Fortran90 (F90) implementation, which had already been adapted to CUDA C in the CUDA version of PyHEADTAIL [28], to CUDA Python in MBTRACK2-CUDA for efficient GPU computation.

The series expansion expressions are straightforward to implement within a CUDA kernel and offer a performance advantage, running few times faster than using the Faddeeva function. For double-precision calculations (float64), we found that the optimal number of terms in the series expansion was $k = 25$ for Eq. (31) and up to $k = 24$ for Eq. (32), thereby ensuring an accuracy of up to $\tau = 11.7\tau_0$. For values of $\tau$ greater than $11.7\tau_0$, the long-range expressions for the means of the wake functions within each interpolated bin were used as follows:

$$
\begin{aligned}
\bar{W}^{\parallel}_{\mathrm{RW}}\left(\tau_{\mathrm{bin}}[m]\right) &\approx \frac{L}{2\pi b}\sqrt{\frac{Z_0}{\pi c \sigma_c}}\frac{1}{\tau_{\max}[m]-\tau_{\min}[m]} \\
&\quad \times \frac{1}{\sqrt{\tau}}\Bigg|_{\tau=\tau_{\min}[m]}^{\tau=\tau_{\max}[m]},
\end{aligned}
\tag{33}
$$

$$
\begin{aligned}
\bar{W}^{\perp}_{\mathrm{RW}}\left(\tau_{\mathrm{bin}}[m]\right) &\approx \frac{2L}{\pi b^3}\sqrt{\frac{Z_0 c}{\pi \sigma_c}}\frac{1}{\tau_{\max}[m]-\tau_{\min}[m]} \\
&\quad \times \sqrt{\tau}\Bigg|_{\tau=\tau_{\min}[m]}^{\tau=\tau_{\max}[m]}.
\end{aligned}
\tag{34}
$$

It is worth noting that the Faddeeva-function expressions are valid over the entire range; therefore, there is no need to use the long-range equations separately when calculating intra-bunch resistive-wall wakefields. Furthermore, when using built-in functions, such as the Faddeeva-function expressions or series expansion expressions, the user does not need to manually set the total length of the wake functions in MBTRACK2-CUDA. The code automatically adjusts to accommodate wake functions based on bunch length, ensuring accurate calculations.

MBTRACK2-CUDA is also capable of performing convolution calculations when wake functions are provided as numerical data rather than built-in functions, allowing for arbitrary wake functions to be used. In such cases, the wake functions must be input as arrays, and these arrays should represent the cumulative sum of the wake functions, meaning that they need to be pre-integrated. Additionally, when using this approach, MBTRACK2-CUDA requires the user to specify the total length of the wake functions because this length is not automatically adjusted by the code.

**Table 1** Machine parameters of the HEPS storage ring used for resistive-wall study

| Parameters | Values |
|---|---|
| Circumference, $L$ (m) | 1360.4 |
| Beam energy, $E_0$ (GeV) | 6 |
| Bunch length, $\sigma_{z,0}$ (mm) | 5.02 |
| Energy spread, $\sigma_{\delta,0}$ | $1.02 \times 10^{-3}$ |
| Natural emittance, $\epsilon_{x,0}/\epsilon_{y,0}$ (pm·rad) | 34.82/3.48 |
| Betatron tune, $\nu_x/\nu_y$ | 115.15/104.29 |
| Corrected chromaticity, $\xi_x/\xi_y$ | 4.95/5.01 |
| Energy loss per turn, $U_0$ (MeV) | 2.64 |
| Damping time, $\tau_x/\tau_y/\tau_z$ (ms) | 10.86/20.62/18.71 |
| Momentum compaction, $\alpha_c$ | $1.83 \times 10^{-5}$ |
| Harmonic number, $h$ | 756 |
| Main RF frequency, $f_{RF}$ (MHz) | 166.6 |
| Average beta, $\beta_x^{avg}/\beta_y^{avg}$ (m) | 4.24/5.98 |
| Beam pipe conductivity, $\sigma_c$ (S/m) | $1.3 \times 10^6$ |
| Beam pipe radius, $b$ (mm) | 11 |

#### 2.2.4 Calculating the influence of coupled-bunch wakefields

To consider the effects of coupled-bunch wakefields in a multi-bunch system, we model each target bunch using a point charge representation, while the source bunch remains represented by macro-particles. This approach is valid when the dominant frequency of the long-range wake is sufficiently low, such that its wavelength is much larger than the bunch length, ensuring that the wake interacts with the bunch as a whole, rather than being sensitive to its internal structure. For instance, this assumption holds true for resistive-wall wakefields. In this model, all macro-particles within the source bunch impart the same kick to the target bunch. Additionally, because wakefields generated by a bunch do not dissipate when the bunch circulates around the synchrotron ring multiple times, the influence of residual wakefields from previous turns, which is often referred to as wake memory or wake history [2, 6], must also be considered.

Let the source bunch index be $i$, target bunch index be $i'$, and total number of bunches be $N_b$. To track the wake memory from the present back to the past, we introduce the turn memory index $l$, where $l = 0$ represents the current turn, and $l = N_t$ represents the oldest turn considered in the calculations. When calculating up to the turn $l = N_t$, the kick factors that all macro-particles in $i'$-th target bunch receive uniformly at the current turn are given by the following:

$$K^{\parallel}(\tau_{i'0}) = \sum_{i=0}^{i'-1} W^{\parallel}(\tau_{i'0} - \tau_{i0}) Q_{i0} + \sum_{l=1}^{N_t} \sum_{i=0}^{N_b-1} W^{\parallel}(\tau_{i'0} - \tau_{il}) Q_{il}, \tag{35}$$

$$K_u^{\perp}(\tau_{i'0}) = \sum_{i=0}^{i'-1} W_u^{\perp}(\tau_{i'0} - \tau_{i0}) Q_{i0} \langle u \rangle_{i0} + \sum_{l=1}^{N_t} \sum_{i=0}^{N_b-1} W_u^{\perp}(\tau_{i'0} - \tau_{il}) Q_{il} \langle u \rangle_{il}, \tag{36}$$

where $Q_{il}$ is the total charge of the source bunch and $\langle u \rangle_{il}$ is the transverse center of mass (CM) position of the source bunch. Note that the intra-bunch wake effects are ignored in Eqs. (35) and (36) because they are already considered in the kicks on the intra-bunch wakefields. To consider the coupled-bunch effects for resistive-wall wakefields, one can

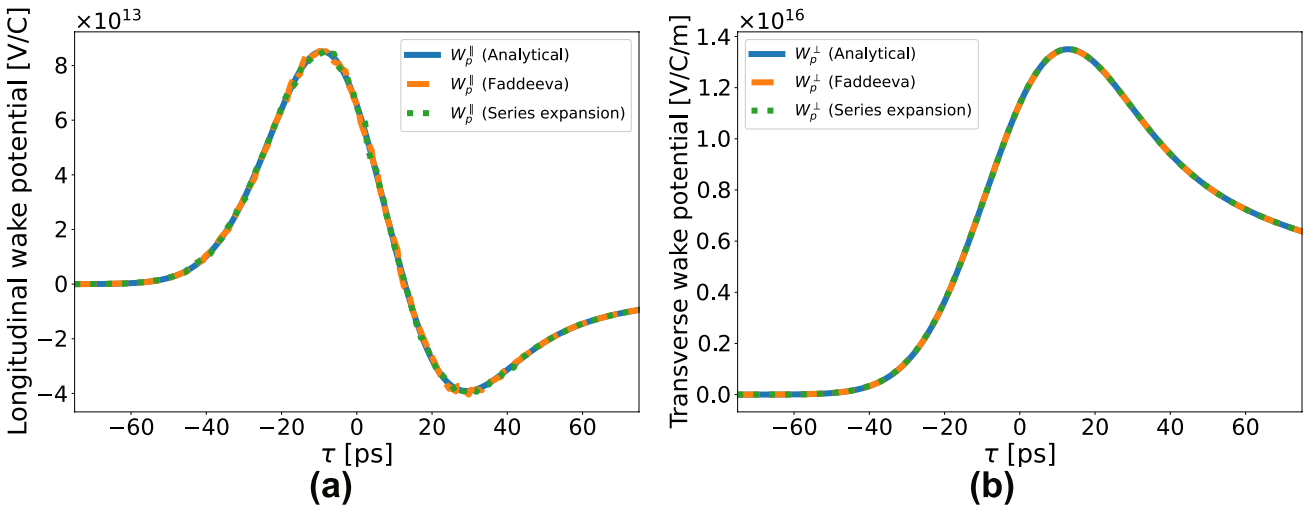


**Fig. 3** Comparisons of wake potentials of resistive-wall for an ideal Gaussian bunch at a single-bunch current of 5 mA from theory and simulations: **a** Longitudinal plane. **b** Transverse plane with omitting dipole moment

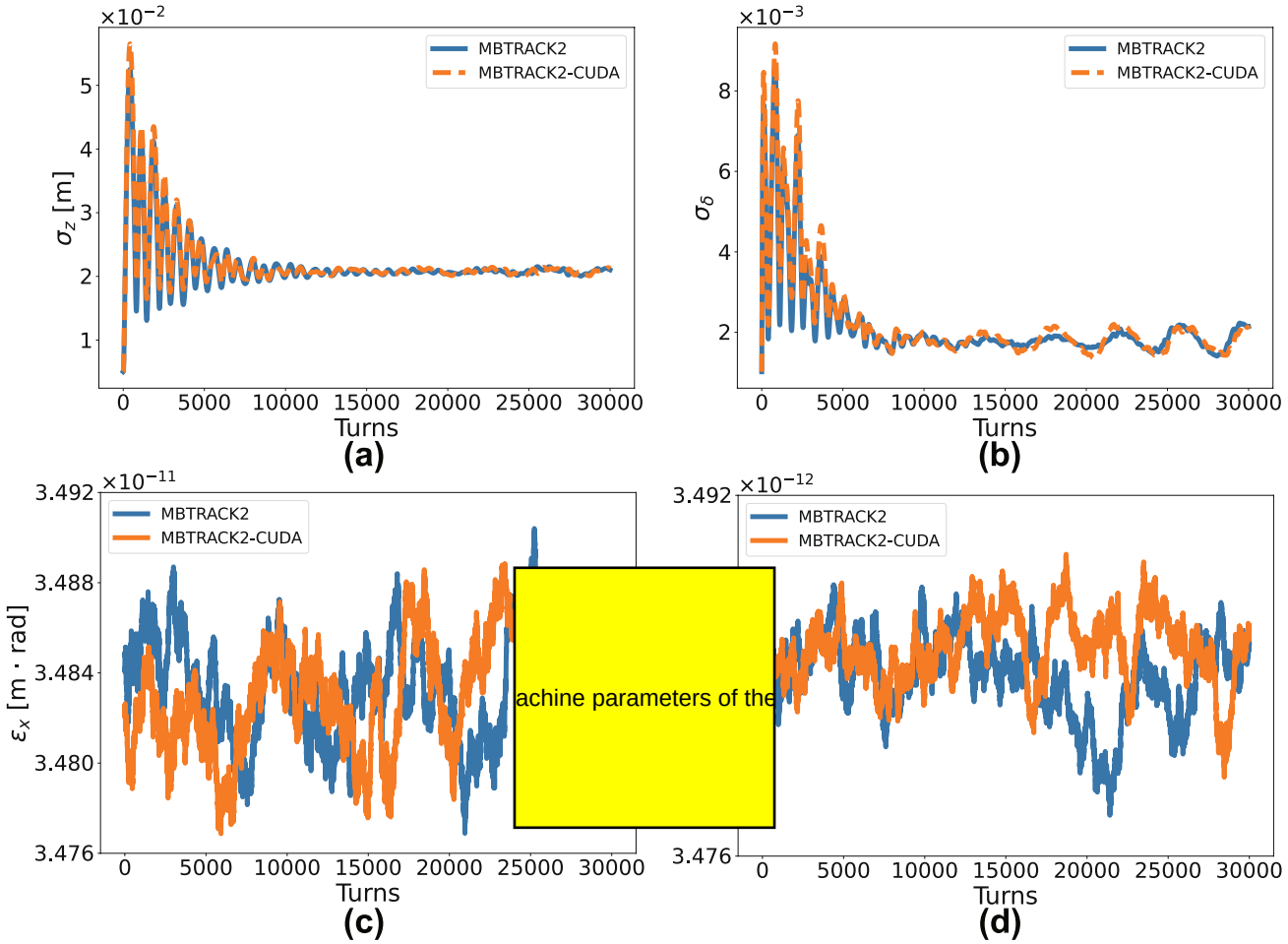


**Fig. 4** Comparisons of MBTRACK2 and MBTRACK2-CUDA tracking simulation results in the presence of the intra-bunch resistive-wall wake-fields at a single-bunch current of 5 mA: **a** Bunch length. [illegible] emittance. **d** Vertical emittance

simply substitute the wake functions on the right [illegible] of these kick factors with asymptotic wake functi[illegible] lae for the resistive-wall, as given in Eqs. (19) an[illegible]

In conclusion, in a coupled-bunch scenario, the [illegible] the relative energy deviation and transverse mom[illegible] macro-particles within the $i'$-th target bunch can be calculated as follows:

$$\Delta\delta_{i'} = -\frac{eK^{\parallel}(\tau_{i'0})}{E_0}, \tag{37}$$

$$\Delta u'_{i'} = \frac{eK_u^{\perp}(\tau_{i'0})}{E_0}\frac{\beta_u^{\mathrm{avg}}}{\beta_u^{\mathrm{loc}}}. \tag{38}$$

achine parameters of the storage ring used for resis[illegible]

[illegible]mparison of the computation times between MBTRACK2 [illegible]e CPU and MBTRACK2-CUDA for single and multi-[illegible]ive-wall simulations

| | MBTRACK2 | MBTRACK2-CUDA |
|---|---|---|
| 1 | 4,079 s | 959 s |
| 2 | 6,920 s | 1,578 s |
| 5 | 7,885 s | 2,041 s |
| 10 | 10,767 s | 2,117 s |
| 100 | – | 6,658 s |
| 340 | – | 19,346 s |
| 680 | – | 36,442 s |

## 3 Beam-tracking simulations of resistive-wall instability using the HEPS storage ring parameters

### 3.1 Validation and benchmarking of MBTRACK2-CUDA

Before evaluating the effects of resistive-wall instability using the machine parameters of the HEPS storage ring with MBTRACK2-CUDA, we first validated the code and conducted a benchmark against MBTRACK2. The machine parameters used in this study for the HEPS storage ring are presented in Table 1 [29, 30]. For the resistive-wall instability, we used a simplified model in which the beam pipes were assumed to have a circular geometry and were made entirely of 316LN stainless steel. The analytical expressions of the intra-bunch resistive-wall wake potentials for an ideal Gaussian bunch can be obtained by substituting a Gaussian distribution into the longitudinal bunch distribution of Eqs. (21) and (22) as follows [31].

$$\left[W_{\mathrm{p}}^{\parallel}(\tau)\right]_{\mathrm{RW}}^{G} = \frac{Z_0 cL}{16\pi b^2}\left[\frac{c^2\tau_0|\tau|}{\sigma_z^2}\right]^{3/2} e^{-\zeta}\left[-I_{1/4}(\zeta) + I_{-3/4}(\zeta) - \mathrm{sgn}(\tau)I_{-1/4}(\zeta) + \mathrm{sgn}(\tau)I_{3/4}(\zeta)\right], \tag{39}$$

$$\left[W_{\mathrm{p}}^{\perp}(\tau)\right]_{\mathrm{RW}}^{G}\Big|_{\mathrm{omit}\,\langle u_{\mathrm{bin}}\rangle} = \frac{Z_0 c^2\tau_0 L}{4\pi b^4}\sqrt{\frac{c^2\tau_0|\tau|}{\sigma_z^2}}e^{-\zeta}\left[I_{-1/4}(\zeta) + \mathrm{sgn}(\tau)I_{1/4}(\zeta)\right], \tag{40}$$

where $\zeta = \left(\frac{c\tau}{2\sigma_z}\right)^2$, $\sigma_z$ is the bunch length of an ideal Gaussian bunch, and $I_\alpha$ is the modified Bessel function for the order $\alpha$. It is noteworthy that $\langle u_{\mathrm{bin}}\rangle$ is omitted from the derivation of Eq. (40) for the validation. In addition, when $\tau$ is zero, Eqs. (39) and (40) are not available; therefore, in this case, a limit value near $\tau = 0$ should be applied.

Figure 3 compares the wake potentials calculated by MBTRACK2-CUDA using both the Faddeeva expressions and the series expansion expressions with the analytical results from Eqs. (39) and (40) for an ideal Gaussian bunch. In these simulations, MBTRACK2-CUDA employed two million macro-particles to represent the Gaussian bunch, utilizing 100 initial bins and 6,000 interpolated bins for the wake potential calculations. These results show almost identical agreement, validating the accuracy of

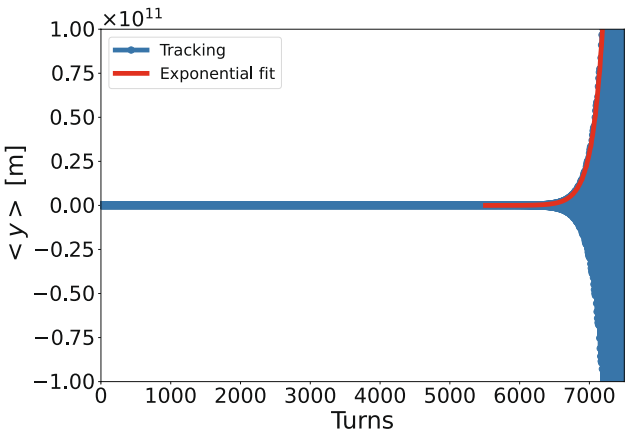


**Fig. 5** Exponential fit of the vertical center of mass for determining the growth rate of the transverse coupled-bunch instability caused by resistive-wall impedance at a ring current of 200 mA

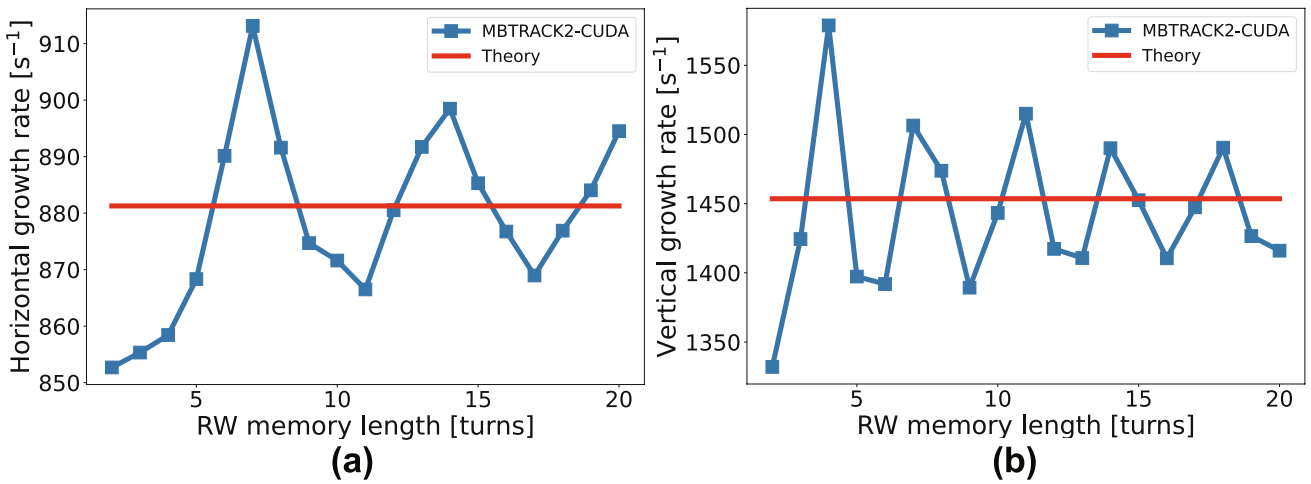


**Fig. 6** Dependences of growth rates on resistive-wall memory length and comparisons with theoretical predictions for a ring current of 200 mA: **a** Horizontal plane. **b** Vertical plane

MBTRACK2-CUDA in calculating the intra-bunch resistive-wall wakefields.

We benchmarked MBTRACK2-CUDA against MBTRACK2 for a single bunch with a current of 5 mA in the presence of intra-bunch resistive-wall wakefields above the microwave instability threshold in a system with a single main RF cavity with a peak voltage of 3.176 MV. The simulations were performed for over 30,000 turns, and the results are shown in Fig. 4. Both codes used 2 million macro-particles and 100 initial bins. However, MBTRACK2 utilizes $10^5$ interpolated bins, whereas MBTRACK2-CUDA uses $6 \times 10^3$ interpolated bins. For the intra-bunch resistive-wall wake functions, MBTRACK2 used the Faddeeva function formulae in Eqs. (16) and (17), whereas MBTRACK2-CUDA employed the series expansion formulae in Eqs. (31) and (32). The simulations showed good agreement in terms of the bunch length and energy spread. However, a discrepancy in transverse emittance was observed between the two codes. This difference arises from the use of distinct RNG algorithms for quantum excitation in each code. Despite this variation, the amplitude of the quantum noise was consistent between the simulations, ensuring that the discrepancy did not significantly impact the overall results.

The system used for the simulations consisted of an Intel Xeon w5-3423 CPU and an NVIDIA GeForce RTX 4090 GPU with 24 GB of GDDR6X VRAM, featuring an error correction code (ECC) enabled. For the simulation of a single bunch with $2 \times 10^6$ macro-particles undergoing an intra-bunch resistive-wall wakefield, MBTRACK2 took 17 h, 38 min, and 25 s (63,505 s), respectively. In contrast, MBTRACK2-CUDA, following the same procedure, completed the simulation in only 35 min and 13 s (2,113 s), demonstrating a remarkable reduction in the computation time. Following this, tracking simulations in the presence of resistive-wall instability for both single- and multi-bunch cases with $10^5$ macro-particles were performed, and the computation times were compared, as shown in Table 2.

The default data type for beam-tracking computations in MBTRACK2-CUDA was double-precision (float64). However, users can choose to use single-precision (float32) to store the 6D phase-space coordinates and RNG. When the single-precision option is selected, the tracking computations are still performed with double-precision for accuracy, and the 6D phase-space coordinates are updated in double-precision before being converted back to single-precision for storage. This approach helps reduce the VRAM consumption and speed up the process, although it may slightly increase the uncertainty in the results.

To validate the code for simulating the transverse coupled-bunch instability caused by resistive-wall impedance in a multi-bunch environment, we disabled the beam loss caused by the beam pipe aperture and intra-bunch resistive-wall wakefields. In addition, we set the chromaticity to zero and applied a uniform filling pattern for the beam-tracking simulation. To calculate the growth rate in this environment, an exponential fit was applied to the center of mass from the tracking result in the transverse direction, as shown in Fig. 5. The growth rate was determined using the following equation:

$$\langle u \rangle (t + \Delta t) = \langle u \rangle (t) \exp\left(\frac{\Delta t}{\tau_{\mathrm{tot}}}\right), \tag{41}$$

where $t$ is the initial temporal position of the fit, $\Delta t$ is the temporal length of the fit, and $\tau_{\mathrm{tot}}^{-1}$ is the total growth rate, defined by substracting the damping rate $\tau_u^{-1}$ from the growth

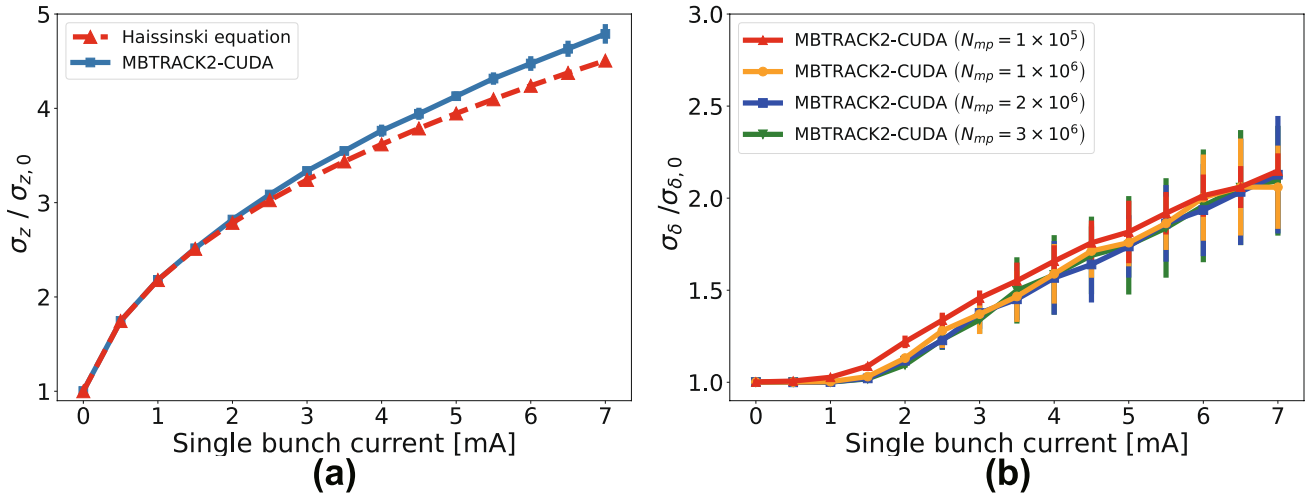


**Fig. 7** Variations of normalized bunch length and normalized energy spread with changing single-bunch current in the presence of intra-bunch resistive-wall wakefields: **a** Comparison of normalized bunch length from tracking simulations and solving the Haïssinski equation. **b** Comparison of normalized energy spread for $1 \times 10^5$, $1 \times 10^6$, $2 \times 10^6$, and $3 \times 10^6$ macro-particles

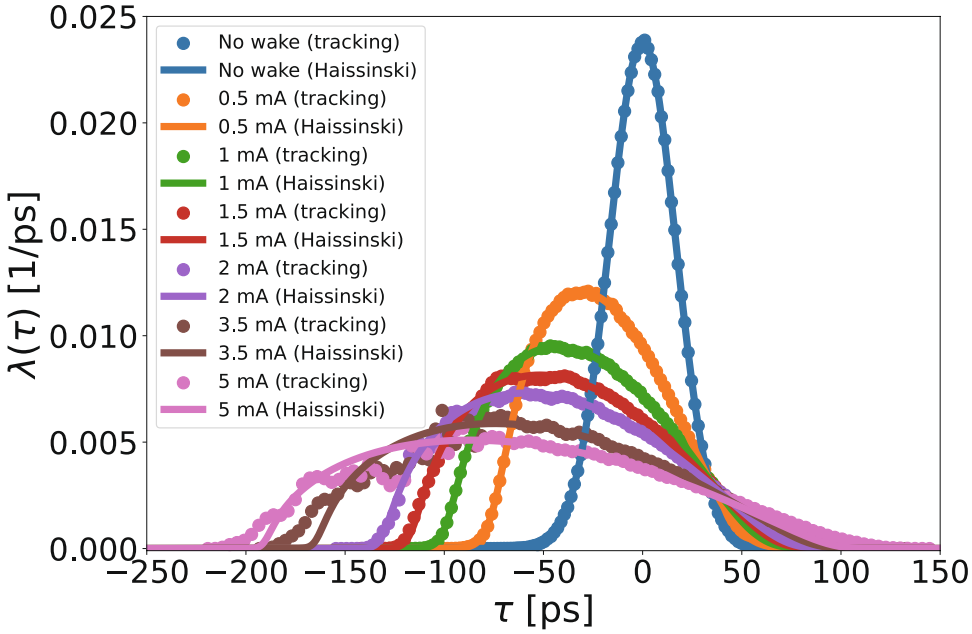


**Fig. 8** Comparisons of longitudinal bunch distributions obtained from tracking and the Haïssinski equation for different bunch currents in the presence of intra-bunch resistive-wall wakefields

rate of the resistive-wall $\tau_{\mathrm{RW}}^{-1}$. The theoretical prediction of the growth rate of the coupled-bunch resistive-wall for the case of zero chromaticity, uniform filling, and in the absence of intra-bunch wakefields is given by the following equation [32]:

$$\left[\tau_{\mathrm{RW}}^{-1}\right]_{\xi_u=0} = \frac{\beta_u^{\mathrm{avg}}\omega_0 e I_{\mathrm{beam}} L}{8\pi^2 E_0 b_{\mathrm{eff}}^3}\sqrt{\frac{2cZ_0}{\left[1-\mathrm{frac}\left(\nu_u\right)\right]\omega_0\sigma_{\mathrm{c}}}}, \tag{42}$$

where $\omega_0$ denotes the angular revolution frequency, $I_{\mathrm{beam}}$ denotes the total beam (ring) current, $\mathrm{frac}\left(\nu_u\right)$ denotes the fractional part of the betatron tune, and $b_{\mathrm{eff}}$ denotes the effective radius of the beam pipe. Detailed discussions of $b_{\mathrm{eff}}$ can be found in Ref. [2, 6]. In the analysis presented here, however, a single pipe configuration was assumed; therefore, $b_{\mathrm{eff}}$ was simply considered as $b$ in Table 1.

Because the growth rate obtained from the simulation was affected by the resistive-wall memory length, the simulation was repeated while varying the number of memory turns. The results are presented in Fig. 6 show that as the number of memory turns increases, the simulated growth rate approaches the theoretical value obtained by Eq. (42) for both the horizontal and vertical planes. Although the results exhibit oscillations rather than perfect convergence, the amplitude of these oscillations remains of the order of 10 $\mathrm{s}^{-1}$, allowing for a reliable determination of the growth rate. A similar discussion on this behavior can be found in Ref. [6].

### 3.2 Single-bunch tracking results of intra-bunch resistive-wall wakefields

We investigated the impact of intra-bunch resistive-wall wakefields on a single bunch using the machine parameters of the HEPS storage ring through tracking simulations conducted in a system featuring a single main RF cavity. To study the collective effects at high single-bunch currents, simulations often require millions of macro-particles [33, 34]. This is especially true when investigating single-bunch currents from low to high, particularly above the microwave instability threshold, where the simulations become highly sensitive to numerical noise.

To determine the appropriate number of macro-particles, we conducted tracking simulations to compute the energy spread as a function of single-bunch current for different numbers of macro-particles. From this, we identify the convergence point, as shown in Fig. 7b, where the energy spread converges around 1 million macro-particles. As mentioned earlier, for the purpose of the single-bunch instability study in this study, we utilized two million macro-particles. The mean value and standard deviation over the last 10,000 turns were used to generate the graphs.

We obtained the equilibrium longitudinal bunch distributions at different single-bunch currents through tracking simulations in the presence of potential-well distortion (PWD). The theoretical equilibrium longitudinal bunch distributions affected by the intra-bunch resistive-wall wakefield can be calculated using the Haïssinski equation [35]:

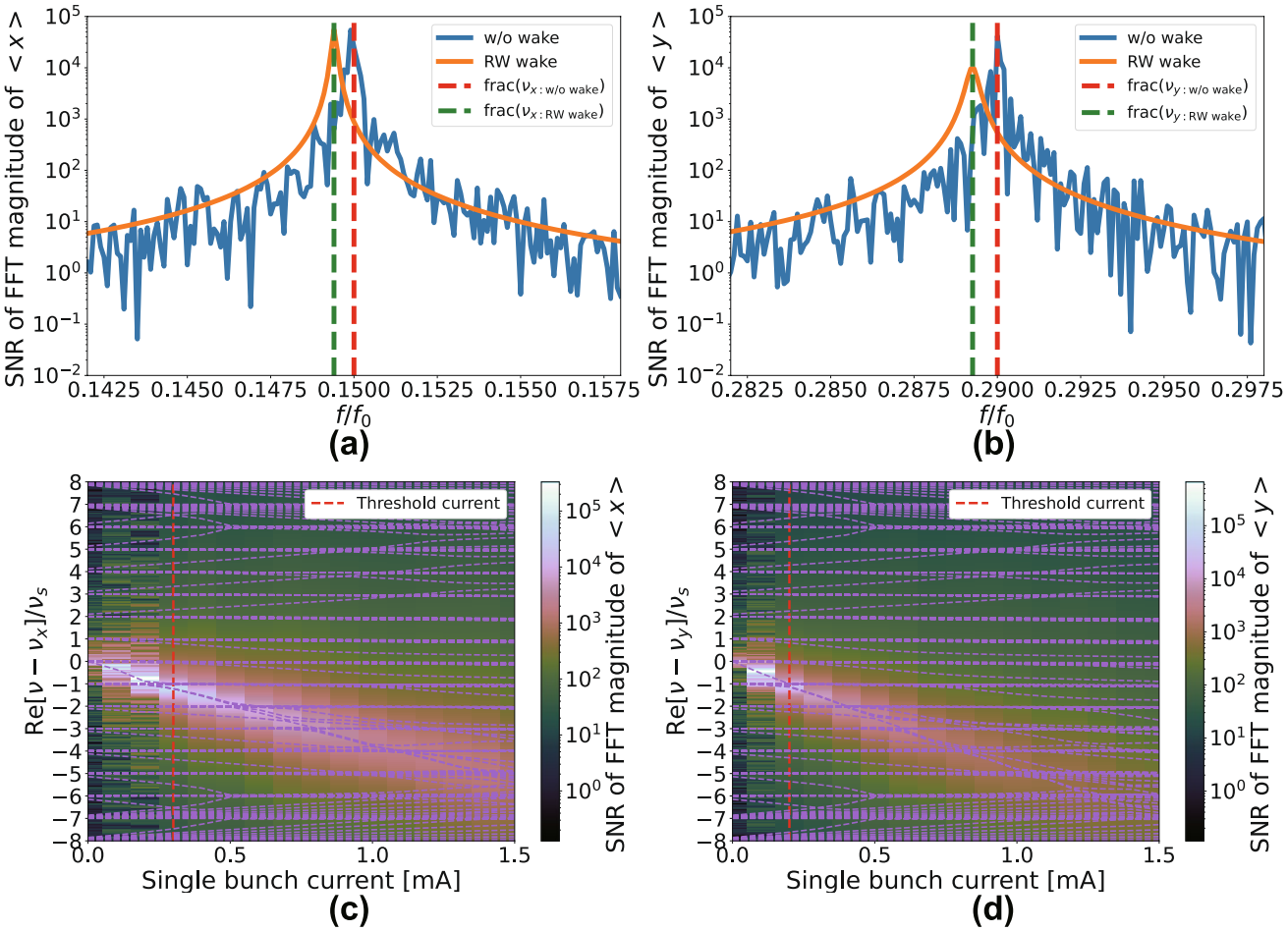


**Fig. 9** (Color online) TMCI analyses: **a**, **b** Comparisons of the FFT magnitude spectrums obtained from MBTRACK2-CUDA are presented for the cases without wakefields and with intra-bunch resistive-wall wakefields, each evaluated at 0.3 mA under zero chromaticity in the horizontal and vertical planes, respectively. **c**, **d** Betatron mode shifts tracked by MBTRACK2-CUDA and analytically computed by pycolleff with zero chromaticity in the horizontal and vertical planes, respectively

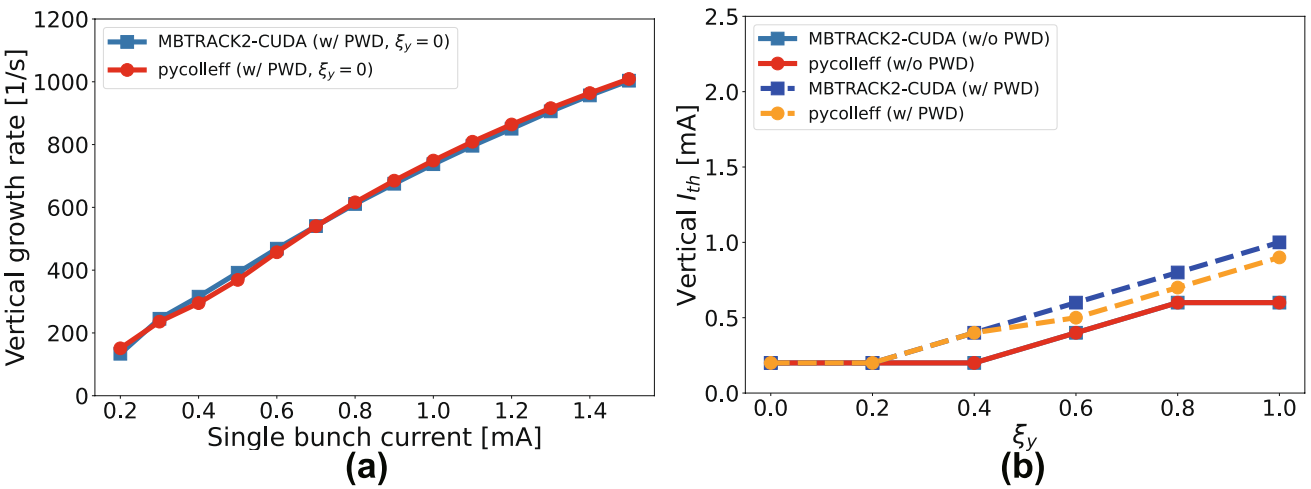


**Fig. 10** Comparisons of transverse instability calculation results between MBTRACK2-CUDA and pycolleff in the vertical plane: **a** Comparison of TMCI growth rate for varying single-bunch current with PWD and zero chromaticity. **b** Comparison of single-bunch transverse instability threshold current for varying chromaticity with and without PWD

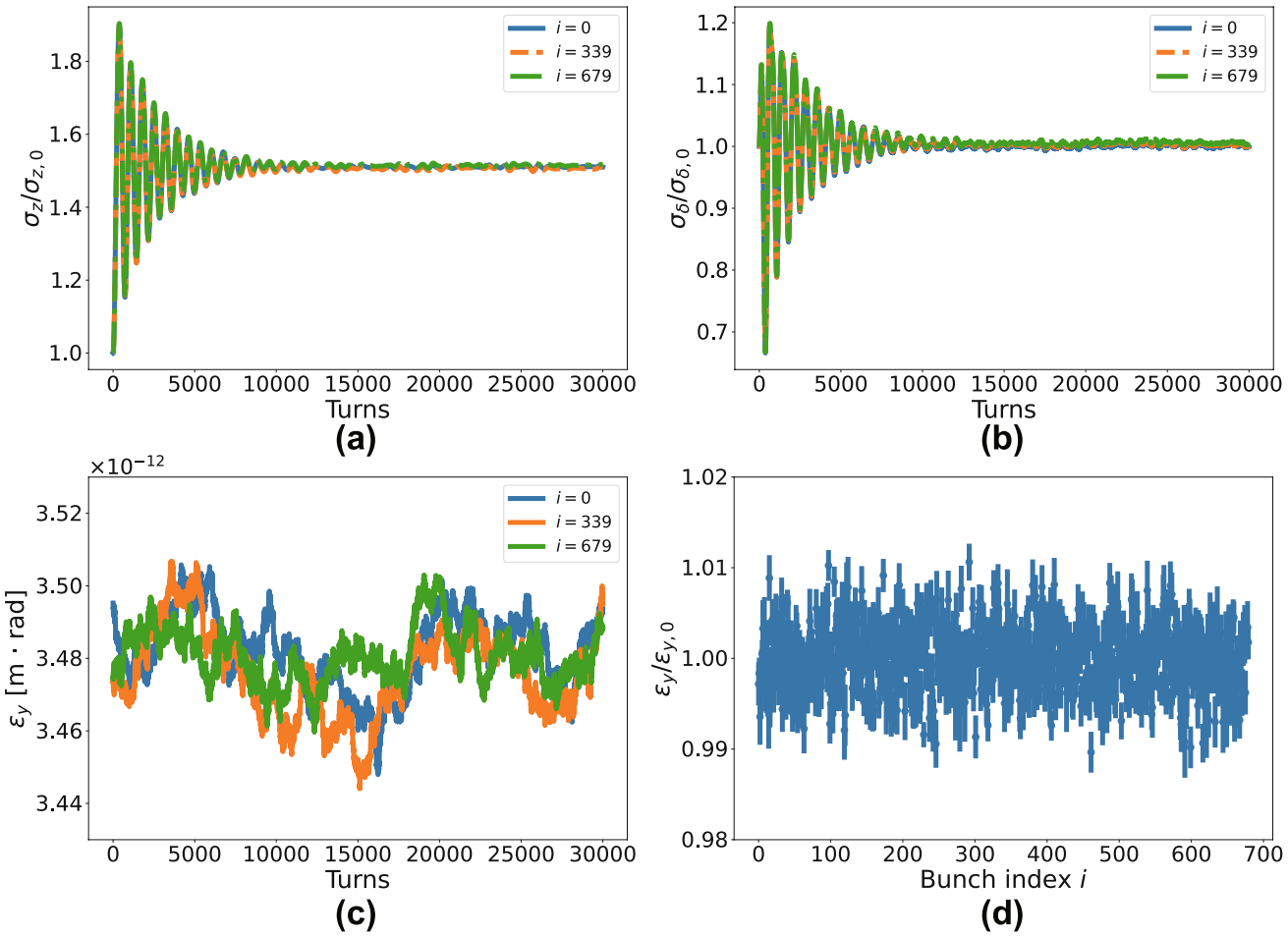


**Fig. 11** (Color online) Multi-bunch tracking results with intra-bunch and coupled-bunch resistive-wall wakefields for the high-brightness mode configuration without harmonic cavity: **a** Normalized bunch lengths for the first, middle, and last bunches. **b** Normalized energy spreads for the first, middle, and last bunches. **c** Vertical emittance for the first, middle, and last bunches. **d** Normalized vertical emittances of all bunches throughout the entire tracking process

$$\lambda(\tau) = \lambda_0 \exp\left[-\frac{\Phi(\tau)}{\alpha_c \sigma_{\delta,0}^2}\right], \tag{43}$$

where $\lambda_0$ is a normalization constant for the equilibrium longitudinal bunch distribution, $\sigma_{\delta,0}$ is the natural energy spread, and

$$\Phi(\tau) = -\frac{c}{E_0 L}\int_0^\tau \Big[eV_{\mathrm{RF}}(\tau') - eQ\int_0^\infty \lambda(\tau'-\tau'')W^{\parallel}(\tau'')\,\mathrm{d}\tau'' - U_0\Big]\,\mathrm{d}\tau'. \tag{44}$$

To solve the Haïssinski equation, we used the pycolleff [36–38], which includes modules for impedance analysis and wakefield-induced instability evaluation. We input the longitudinal resistivity-wall impedance, given by Eq. (13) in the code to obtain the equilibrium longitudinal bunch distributions for different single-bunch currents.

Figure 8 shows a comparison of the equilibrium longitudinal bunch distributions between the results of the tracking simulations and those obtained from the Haïssinski equation. To provide further insight into the bunch dynamics, the variation in bunch length and energy spread for different single-bunch currents is presented in Fig. 7. From Fig. 7b, it can be observed that the microwave instability (MWI) threshold occurs at approximately 1.5 mA. Below this threshold, as illustrated in Fig. 8 shows that bunch lengthening is predominantly caused by PWD owing to the intra-bunch resistive-wall wakefield. The equilibrium longitudinal bunch distributions obtained by tracking with MBTRACK2-CUDA and solving the Haïssinski equation using pycolleff demonstrate excellent agreement under these conditions.

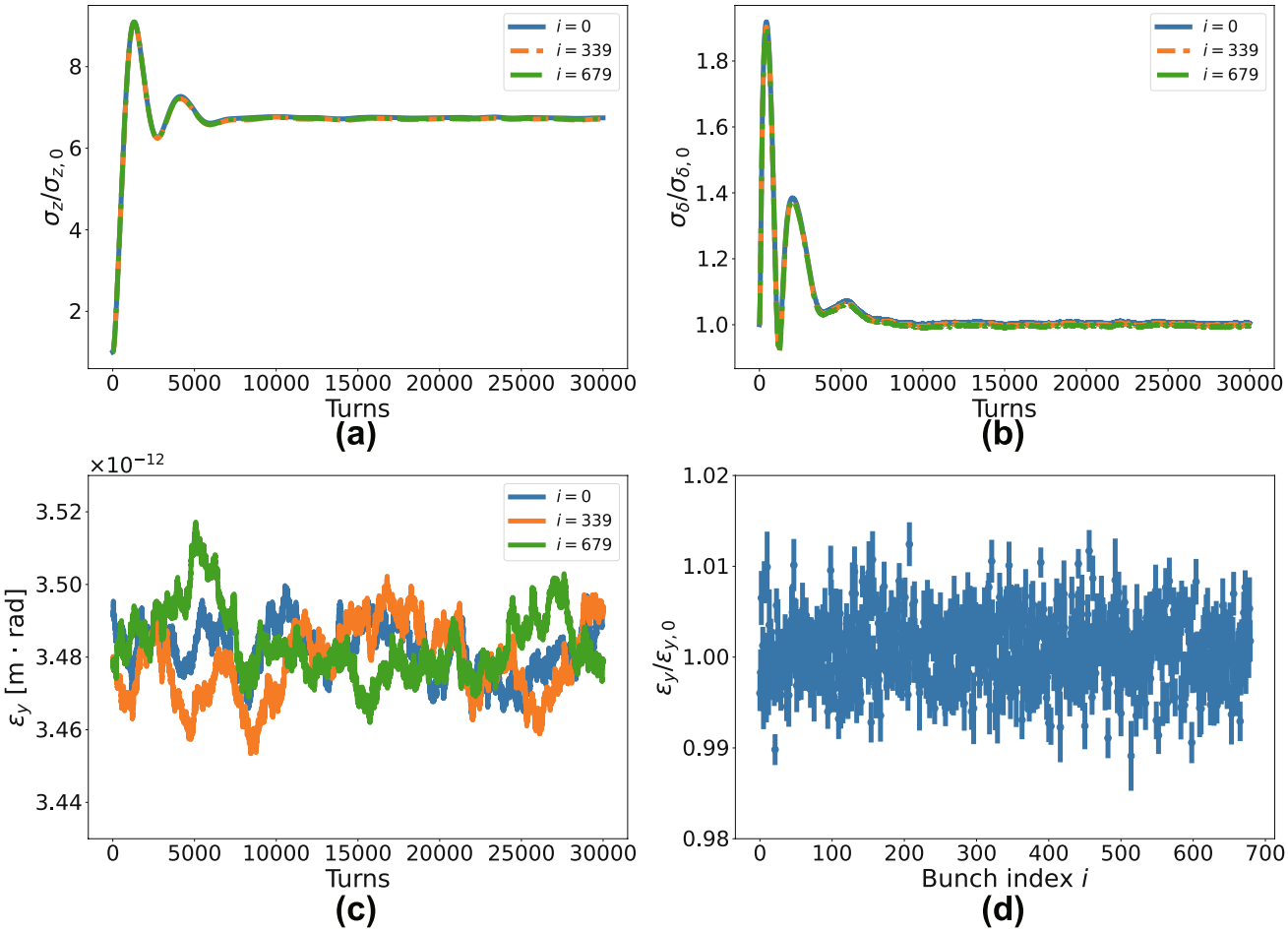


**Fig. 12** (Color online) Multi-bunch tracking results with intra-bunch and coupled-bunch resistive-wall wakefields for the high-brightness mode configuration with harmonic cavity: **a** Normalized bunch lengths for the first, middle, and last bunches. **b** Normalized energy spreads for the first, middle, and last bunches. **c** Vertical emittances for the first, middle, and last bunches. **d** Normalized vertical emittances of all bunches throughout the entire tracking process

However, at currents exceeding the MWI threshold, a difference began to emerge. The tracking simulations revealed a distorted bunch structure in the head region owing to MWI. Additionally, the MWI causes an overall increase in the bunch length when compared to the Haïssinski equation results. This discrepancy in bunch length is shown in Fig. 7a, because the Haïssinski equation does not account for the effects of the MWI.

To study the transverse mode-coupling instability (TMCI), we first set the chromaticity to zero and tracked the center of mass while varying the single-bunch current. We identified the threshold single-bunch current where growth occurred and analyzed the turn-by-turn data of the center of mass using fast Fourier transform (FFT). By calculating the signal-to-noise ratios (SNR) of the FFT magnitudes, we observed betatron tuning shifts, as shown in Figs. 9a and b. The corresponding shifts were especially noticeable as the single-bunch current increased in the presence of resistive-wall impedance. However, no changes in the fractional tune were observed in the absence of resistive-wall impedance.

We normalized these betatron tune shifts by the synchrotron tune, which accounts for the incoherent synchrotron tune shift caused by PWD, and plotted the betatron mode shift spectrograms, presented in Figs. 9c and d. Using the changing bunch length information derived from the Haïssinski equation, we performed a mode-shift analysis with pycolleff and overlaid the results on the same graphs. The mode-shift results obtained from the two approaches matched well in the region below the MWI threshold.

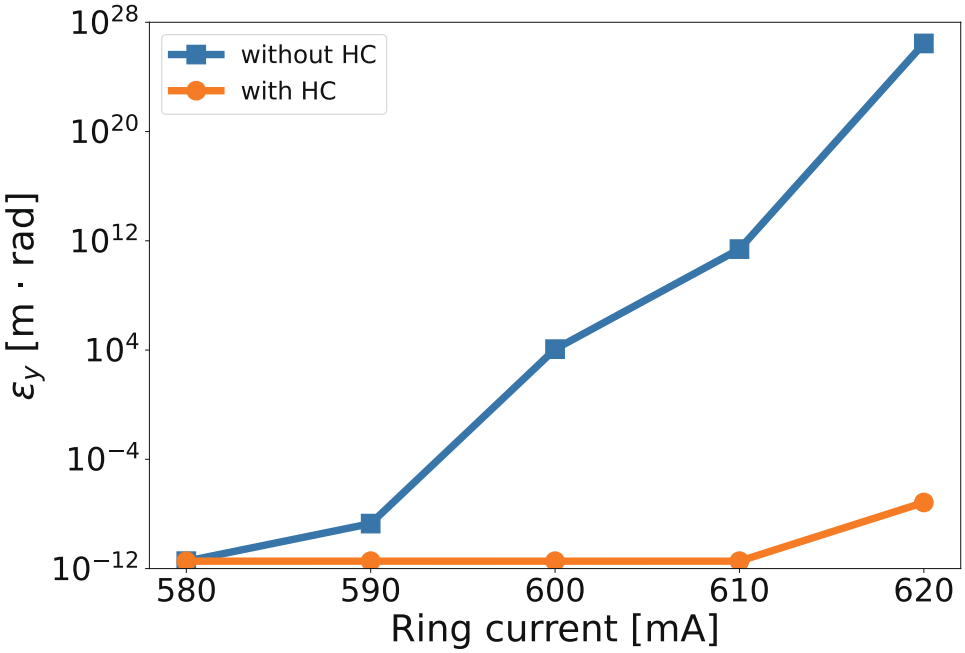


**Fig. 13** Variation in the vertical emittance of the last bunch at 30,000 turns as a function of ring current in the presence of intra-bunch and coupled-bunch resistive-wall wakefields, with and without an active harmonic cavity

In addition, to conduct in-depth analyses of the effects of transverse instability, the transverse growth rate and single-bunch threshold current were computed using MBTRACK2-CUDA and pycolleff, respectively. The results were in excellent agreement with the vertical growth rate presented in Fig. 10a for varying single-bunch current with PWD and zero chromaticity. The single-bunch vertical-threshold current calculations shown in Fig. 10b for varying chromaticity, both with and without PWD, also yielded consistent results.

### 3.3 Simultaneous tracking results of intra-bunch and coupled-bunch resistive-wall wakefields in a multi-bunch system

Tracking simulations were conducted to evaluate the effects of intra-bunch and coupled-bunch resistive-wall wakefields on multi-bunches using the machine parameters of the HEPS storage ring under the high-brightness mode configuration. This configuration consists of a filling pattern with a 680-bunch train and 76 empty bucket gaps out of a total of 756 buckets [29, 39]. In the high-brightness mode configuration, the current per bunch was approximately 0.29 mA. As shown in Fig. 7b, for $10^5$ macro-particles per bunch, the energy spread converged sufficiently in the region where the current per bunch was below 1 mA. Based on this observation, we set the number of macro-particles per bunch to $10^5$ in the tracking simulation. In addition, 100 initial bins and $6 \times 10^3$ interpolated bins were used to calculate the effects of the intra-bunch wakefields. The horizontal and vertical chromaticities were configured to 4.95 and 5.01, respectively.

With the main RF cavity set to a peak voltage of 3.176 MV in the presence of both intra-bunch and coupled-bunch resistive-wall wake fields, the multi-bunch tracking simulation results are shown in Fig. 11. In equilibrium, the bunch lengths increased by a factor of approximately 1.5, while the transverse emittances for all bunches remained at their initial values, indicating the feasibility of stable operation.

Subsequently, we conducted additional tracking simulations that included the active third harmonic cavity without considering the beam-loading effect, with the main RF cavity set to a peak voltage of 3.393 MV and the active third harmonic cavity set to a peak voltage of 0.638 MV. These simulations were performed with and without resistive-wall effects, and the results for the case with resistive-wall effects are presented in Fig. 12.

In terms of bunch length, the simulation without resistive-wall instability showed that all bunches increased in length from the initial value of 5.02 mm to 29.72 mm, approximately 5.92 times longer due to the effect of the active third harmonic cavity, which closely matched the result of Ref. [29].

When both intra-bunch and coupled-bunch resistive-wall wakefields were present, with the active harmonic cavity in operation, as shown in Fig. 12a, the bunch lengths of all bunches increased to 33.99 mm, approximately 6.77 times longer. Figure 12b shows that the energy spreads remained stable in the equilibrium state, and Fig. 12c, d demonstrates that the transverse emittances were consistently maintained throughout the tracking process, indicating that the HEPS storage ring with the active harmonic

cavity in operation could be stably operated under resistive-wall wakefield conditions.

To investigate the impact of transverse instability when both intra-bunch and coupled-bunch resistive-wall wakefields were present in multi-bunch configurations, we increased the ring current while disabling the beam loss caused by the beam pipe aperture. We then examined the vertical emittance near the threshold at which it started to increase, both with and without an active harmonic cavity, as shown in Fig. 13. The results indicate that the ring current needs to be increased to approximately three times its usual value to observe vertical instability. Furthermore, the presence of an active harmonic cavity slightly increased the instability threshold.

## 4 Summary and outlook

In this study, we developed MBTRACK2-CUDA, a novel approach that fully parallelizes the entire tracking computation process solely on a GPU, building upon the existing MBTRACK2 framework. The performance of the code was remarkable, demonstrating its effectiveness and practicality through tracking simulations addressing resistive-wall impedance by utilizing the machine parameters of the HEPS storage ring.

As GPU architectures continue to evolve, with an increased number of CUDA cores, improved core performance, and expanded memory capacities, fully leveraging these advancements will be beneficial for GPGPU computations. In this context, we believe that executing the entire tracking computation in a fully parallel manner on the GPU is an efficient and future-oriented approach that maximizes GPGPU performance. This approach will pave the way for faster solutions in complex simulations involving large datasets by reducing data transfer overhead.

**Acknowledgements** The authors would like to thank Haisheng Xu and Uldis Locans for helpful discussions regarding MBTRACK-CUDA. We appreciate Haisheng Xu's valuable advice on collective effects studies for the HEPS storage ring. We would also like to express our gratitude to Sungjin Kim and Dae-Ho Kwon for their suggestions regarding CUDA programming using Numba-CUDA. We are also grateful to Alexis Gamelin, Watanyu Foosang, and Vadim Gubaidulin for their invaluable discussions and advice on the usage of MBTRACK2 and tracking simulations for collective effects. Finally, we thank Fernando H. de Sá for providing pycolleff and for fruitful discussions on its proper usage and collective effects.

**Author contributions** All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Keon Hee Kim and Eun-San Kim. The first draft of the manuscript was written by Keon Hee Kim and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

## Declarations

**Conflict of interest** The authors declare that they have no conflict of interest.

## References

1. A. Gamelin, W. Foosang, R. Nagaoka, MBTRACK2, a collective effect library in Python. Proceedings of IPAC2021, Campinas, SP, Brazil (2021). https://doi.org/10.18429/JACoW-IPAC2021-MOPAB070
2. W. Foosang, *Study of transverse beam instabilities in the SOLEIL II synchrotron booster* (Paris-Saclay University, Thesis, 2023)
3. X. Gu, A. Gamelin. MBTRACK2: A longitudinal RF beam loading tracking code. BNL-224788-2023-TECH (2023)
4. X. Gu, A. Blednykh, M. Blaskiewicz et al., MBTRACK2 - Application on EIC 5 GeV electron ring reverse phase configuration. BNL-225173-2024-TECH (2024)
5. A. Gamelin, W. Foosang, N. Yamamoto et al., MBTRACK2. https://doi.org/10.5281/zenodo.12749990
6. G. Skripka, R. Nagaoka, M. Klein et al., Simultaneous computation of intrabunch and interbunch collective beam motions in storage rings. Nucl. Instrum. Methods Phys. Res. Sect. A **806**, 221–230 (2016). https://doi.org/10.1016/j.nima.2015.10.029
7. H.S. Xu, U. Locans, A. Adelmann et al., Calculation of longitudinal collective instabilities with mbtrack-cuda. Nucl. Instrum. Methods Phys. Res. Sect. A **922**, 345–351 (2019). https://doi.org/10.1016/j.nima.2019.01.041
8. T. He, Z. Bai, Graphics-processing-unit-accelerated simulation for longitudinal beam dynamics of arbitrary bunch trains in electron storage rings. Phys. Rev. Accel. Beams **24**, 104401 (2021). https://doi.org/10.1103/PhysRevAccelBeams.24.104401
9. Z. Li, Y. Zhang, K. Ohmi et al., Strong-strong beam-beam simulations with lattices of circular e+e- colliders. Nucl. Instrum. Methods Phys. Res., Sect. A **1064**, 169386 (2024). https://doi.org/10.1016/j.nima.2024.169386
10. K. Amyx, I. Pogorelov, J. King et al., Current status of the GPU-accelerated ELEGANT. Proceedings of IPAC2014, Dresden, Germany (2014). https://doi.org/10.18429/JACoW-IPAC2014-MOPME035
11. A. Oeftiger, Parallelisation of PyHEADTAIL, a collective beam dynamics code for particle accelerator physics. arXiv:1610.05801 (2016). https://doi.org/10.48550/arXiv.1610.05801
12. S.E. Hegglin, *Simulating collective effects on GPUs* (Thesis, ETH Zurich, 2016)
13. G. Iadarola, A. Abramov, P. Belanger et al., Xsuite: An Integrated Beam Physics Simulation Framework. Proceedings of HB2023, Geneva, Switzerland (2023). https://doi.org/10.18429/JACoW-HB2023-TUA2I1
14. M. Schwinzerl, H. Bartosik, R. De Maria et al., Optimising and Extending a Single-Particle Tracking Library for High Parallel Performance. Proceedings of IPAC2021, Campinas, SP, Brazil (2021). https://doi.org/10.18429/JACoW-IPAC2021-THPAB190
15. K.H. Kim, MBTRACK2-CUDA. https://doi.org/10.5281/zenodo.14802810
16. S.K. Lam, A. Pitrou, S. Seibert, Numba: A LLVM-based Python JIT compiler. LLVM '15: Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC (2015). https://doi.org/10.1145/2833157.2833162
17. R.A. Bosch, K.J. Kleman, J.J. Bisognano, Robinson instabilities with a higher-harmonic cavity. Phys. Rev. ST Accel. Beams **4**, 074401 (2001). https://doi.org/10.1103/PhysRevSTAB.4.074401

18. S. Vigna, Further scramblings of Marsaglia's xorshift generators. J. Comput. Appl. Math. **315**, 175–181 (2017). https://doi.org/10.1016/j.cam.2016.11.006
19. M. Matsumoto, T. Nishimura, Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. ACM Trans. Model. Comput. Simul. **8**, 3–30 (1998). https://doi.org/10.1145/272991.272995
20. A.W. Chao, Physics of collective beam instabilities in high energy accelerators. (Wiley, 1993)
21. K.L.F. Bane, G.V. Stupakov, Using surface impedance for calculating wakefields in flat geometry. Phys. Rev. ST Accel. Beams **18**, 034401 (2015). https://doi.org/10.1103/PhysRevSTAB.18.034401
22. H. Schlarb, *Resistive wall wake fields* (University of Hamburg, Thesis, 1997)
23. K.L.F. Bane, M. Sands, The short-range resistive wall wakefields. SLAC-PUB-7074 (1995)
24. M. Ivanyan, V. Tsakanov, Analytical treatment of resistive wake potentials in round pipes. Nucl. Instrum. Methods Phys. Res. Sect. A **522**, 223–229 (2004). https://doi.org/10.1016/j.nima.2003.12.006
25. G.V. Stupakov, G. Penn, Classical mechanics and electromagnetism in accelerator physics, (Springer, 2018). https://doi.org/10.1007/978-3-319-90188-6
26. G.V. Stupakov, Wake and impedance. SLAC-PUB-8683 (2000)
27. B.W. Zotter, S.A. Kheifets, Impedances and wakes in high-energy particle accelerators. (World Scientific, 1998). https://doi.org/10.1142/3068
28. A. Oeftiger, A. Aviral, R. De Maria et al., Review of CPU and GPU Faddeeva implementations. Proceedings of IPAC2016, Busan, Republic of Korea (2016). https://doi.org/10.18429/JACoW-IPAC2016-WEPOY044
29. H.S. Xu, C. Meng, Y. Peng et al., Equilibrium electron beam parameters of the high energy photon source. Radiat. Detect. Technol. Methods **7**, 279–287 (2023). https://doi.org/10.1007/s41605-022-00374-w
30. P. He, B.L. Deng, D.Z. Guo et al., Preliminary design of HEPS storage ring vacuum chambers and components. Proceedings of FLS2018, Shanghai, China (2018). https://doi.org/10.18429/JACoW-FLS2018-TUP2WD04
31. A. Piwinski, Wake fields and ohmic losses in round vacuum chambers. DESY-HERA-92-11 (1992)
32. R. Nagaoka, K.L.F. Bane, Collective effects in a diffraction-limited storage ring. J. Synchrotron Rad. **21**, 937–960 (2014). https://doi.org/10.1107/S1600577514015215
33. W. Li, T. He, Z. Bai, Terahertz scale microbunching instability driven by high resistivity nonevaporable getter coating resistive-wall impedance. Phys. Rev. Accel. Beams **27**, 034401 (2024). https://doi.org/10.1103/PhysRevAccelBeams.27.034401
34. A. Gamelin, W. Foosang, Influence of the coating resistivity on beam dynamics. Phys. Rev. Accel. Beams **26**, 054401 (2023). https://doi.org/10.1103/PhysRevAccelBeams.26.054401
35. J. Haïssinski, Exact longitudinal equilibrium distribution of stored electrons in the presence of self-fields. Nuov. Cim. B **18**, 72–82 (1973). https://doi.org/10.1007/BF02832640
36. M.B. Alves, F.H. Sá, Equilibrium of longitudinal bunch distributions in electron storage rings with arbitrary impedance sources and generic filling patterns. Phys. Rev. Accel. Beams **26**, 094402 (2023). https://doi.org/10.1103/PhysRevAccelBeams.26.094402
37. F.H. de Sá, M.B. Alves, L. Liu, Measurements of collective effects related to beam coupling impedance at SIRIUS. Proceedings of IPAC2022, Bangkok, Thailand (2022). https://doi.org/10.18429/JACoW-IPAC2022-MOOYSP2
38. F.H. de Sá, M.B. Alves, Pycolleff and cppcolleff: modules for impedance analysis and wake-field induced instabilities evaluation. https://doi.org/10.5281/zenodo.8088076
39. J. He, Y.F. Sui, Y.H. Lu et al., Preliminary study on detection and cleaning of parasitic bunches. Nucl. Sci. Tech. **32**, 114 (2021). https://doi.org/10.1007/s41365-021-00948-1