A GENERIC DATA MODEL FOR HEADTAIL: DESIGN AND IMPLEMENTATION WITH EXAMPLES

Kevin Li, G. Rumolo, CERN, Geneva, Switzerland

Abstract

HEADTAIL has been developed in 2002 for the efficient simulation of instabilities and collective effects in large circular accelerators. Since then, the capabilities of the code have been continuously extended and the output data has become increasingly complex and large-scale ranging from the statistical description of single bunches to the statistical description of all slices within bunches up to the dynamics of the full 6D phase space over several thousands of turns. Processing this data in an effective manner and endowing it with a structure that provides a physical concept calls for new and optimised data formats. To meet state-of-the-art standards, the hierarchical data format (HDF5) has been selected as native output data format together with H5Part and XDMF as native data structures. We describe the implementation of the H5Part and the XDMF data structures into HEADTAIL and show some illustrative examples for data processing.

INTRODUCTION

HEADTAIL [1] has become a widely used numerical tool for the efficient simulation of instabilities and collective effects in large accelerator rings. The code handles a wide range of collective effects including different types of impedances (constant, dipolar, quadrupolar, simple broadband models as well as wake tables from complicated accelerator structures), space charge and electron cloud effects. As the code is growing in complexity it is being reorganised towards an object-oriented design. It has been extended by a number of new classes. One of these classes (HDF5_IO) handles data i/o operations in HDF5 format [2].

As the capabilities of the code increase, complicated data structures arise that want to be analysed. Although, up to a certain extend, ways can be found to save the output data in human readable ASCII format, it is not very practical and moreover very poor in performance with regard to subsequent data analysis. The current state-of-the art standard to save scientific data is the HDF5 data format which is widely supported by third party software. These do not only include open-source projects like Python or NetCDF4 but also commercial programs such as Matlab or Mathematica. Specialized sofware tools such as HDFView [3] are freely available and enable quick and easy data investigation. Paraview [4] allows dynamic 3D visualisation of HDF5 data when provided with an XDMF layer [5]. Furthermore, Paraview's native VTK data format now contains HDF5 to allow easy utilisation of HDF5 storage [6].

The HDF5 data format is highly generic and does not impose any structure on the data. This flexibility is beneficial as, in practise, most applications are so specialised that it does not make sense to enforce any given data structure. For HEADTAIL the data structure has been chosen to roughly follow the fiber bundle concept [7], which is a rather natural choice for Hamiltonian systems, making use of the H5Part and the XDMF data structures [8, 5]. The different output data formats will be described below and highlighted by some examples.

HEADTAIL DATA MODELS

The HEADTAIL output data in HDF5 format has been chosen to roughly follow the fiber bundle concept with the fiber bundle $F \longrightarrow E \xrightarrow{\pi} B$ consisting of the topological spaces (E, B, F), where E is called the total space, B is called the base space and F is called the fiber, and a continuous surjection $\pi : E \rightarrow B$ which satisfies the local triviality condition [7]. For most cases, the time line $\mathcal{T}$ corresponds to the base space and the root group of the H5 file corresponds to the total space. The datasets in each group correspond to the fiber. In this formulation the H5 file itself forms a (trivial) fiber bundle. Currently, HEADTAIL outputs four major data files denoted by `*_prt.h5`, `*_hdtl.h5`, `*_prb.h5` and `*_pic.h5`.

The bunch statistics file

The prt file contains the bunch statistics recorded at the Poincaré section over the full length of the simulation. It has the time line $\mathcal{T}$ mapped to the row index of the "StatisticsData" dataset. Each fiber consists of a set of bunch statistics parameters. The data is stored in a compound array similar to a C structure and is referenced likewise when imported using h5py or Matlab. The time evolution of the bunch statistics can be easily viewed using HDFView. Figure 1 highlights the data layout as viewed in HDFView.

The slice statistics file

The hdtl file contains the slice statistics recorded at each BPM location over the full length of the simulation. It has the time line $\mathcal{T}$ mapped to the time slice groups. Each fiber consists of an array containing the slice statistics parameters of all slices. The data is stored in a 2D array which can be imported directly using h5py or Matlab. The attributes of each column are specified in the dataset attributes under the "Data" entry. They are summarised in tab. 1 for reference.

Copyright © 2012 by IEEE – cc Creative Commons Attribution 3.0 (CC BY 3.0) — cc Creative Commons Attribution 3.0 (CC BY 3.0)

ISBN 978-3-95450-115-1

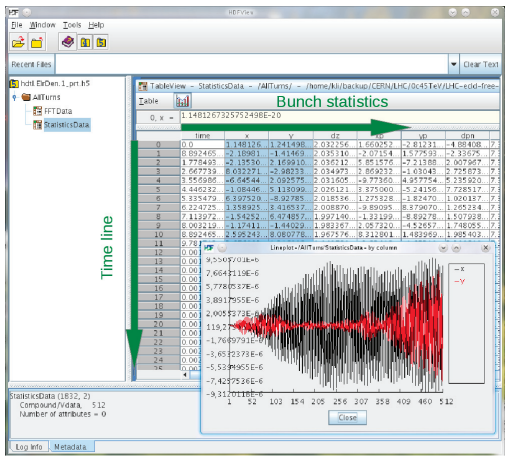

Figure 1: The bunch statistic file (*_prt.h5) as viewed in HDFView. The data is stored in a compound array. HDFView comes with a simple lineplot tool for quick data inspection.

Table 1: The column attributes of the 2D arrays in the slice statistic file (*_hdtl.h5).

Column	Data
1	longitudinal slice position within bunch
2	slice charge
3	derivative slice charge by slice position
4	rms horizontal slice position
5	rms vertical slice position
6	rms horizontal slice size
7	rms vertical slice size

The phase space file

The prb file contains the phase space recorded at each phase space monitor (PSM) location over the full length of the simulation. It has the time line $\mathcal{T}$ mapped to the step number groups as determined by the H5Part data structure. Each fiber consists of a set of arrays that make up the beam phase space Ω. The data is stored in 1D arrays which each needs to be imported individually using h5py or Matlab. By virtue of the H5Part plugin the data can be visualised with Paraview allowing to qualitatively explore the beam dynamics in real space as well as in phase space in order to understand the progression of emittance blow-up or coupling among different planes. Figure 1 highlights the H5Part data structure as viewed in HDFView.

The PIC file

The pic file contains the data from the Particle-In-Cell solver for the interaction of the beam with the electron cloud. This type of data is more complex compared to the data of the previous files since it derives from a field theory rather than from a single particle theory, hence, it contains more degrees of freedom. It still has the time line $\mathcal{T}$ mapped to the time slice groups. In addition, it has

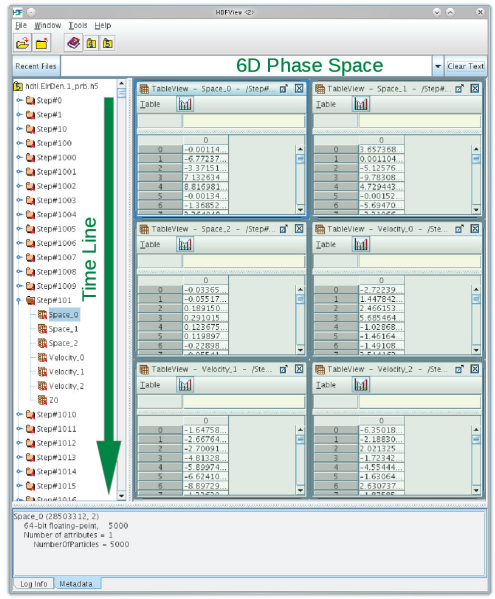

Figure 2: The 6D phase space file (*_prb.h5) as viewed in HDFView. The left frame displays the time line organised in time slice groups. Each time slice group contains the 6D phase space information in 1D arrays displayed in the right frame.

the static group that contains all field data as a 3D representation of the passage of the beam through the electron cloud. Furthermore, space $\mathcal{S}$ is discretised in the space group containing the geometry datasets "2DGeometry" and "3DGeometry" and the total number of points each space is mapped to as attribute. The topology is given implicitly as "2DRectMesh" and "3DRectMesh" for 2D space and 3D data, respectively.

The data is stored in a 2D or 3D arrays and contains the space geometry in the space group as well as the density and potential fields of the beam and the cloud in the time slice groups. An XDMF file output by the class provides the description of the H5 data in XDMF format so that the data can be visualised with Paraview. Fig. 3 highlights the data layout as viewed in HDFView.

EXAMPLES

The following section shows some examples of exploiting some HEADTAIL output data by visualisation with Paraview. The first example in fig. 4 shows the effects of a constant wake field on the transverse phase space. The slice dependent orbit distortion becomes visible and its dynamics can be examined in the 3D representation. Fig. 5 displays a snapshot of the dynamical evolution of an electron cloud pinch. Fig. 6 shows the same electron cloud pinch in a static 3D representation along with a color map of the potential field evolution, where the z-axis is mapped to the slice position which in turn relates to time according to $\sigma = (t - t_0)\beta c$. The cloud density evolution can be examined along any arbitrary cut through the box. In

Copyright © 2012 by IEEE – cc Creative Commons Attribution 3.0 (CC BY 3.0) — cc Creative Commons Attribution 3.0 (CC BY 3.0)

ISBN 978-3-95450-115-1

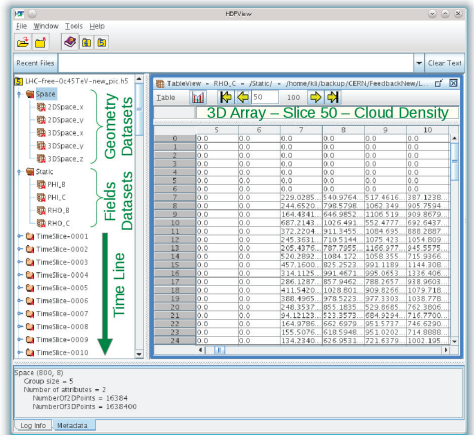

Figure 3: The PIC file (*_pic.h5) as viewed in HDFView. The geometry datasets along with the static group and the time slice groups are displayed in the left frame. The right frame shows the representation of a 3D data array.

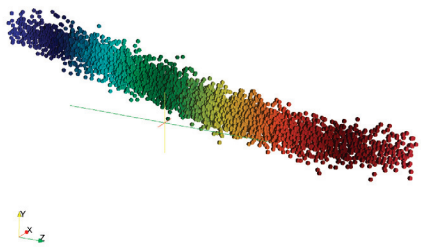

Figure 4: The effects of a constant wakefield on the transverse phase space. The axis mapping is $x \to x$, $y \to x'$ and $z \to \sigma$. The effect is similar to the potential well distortion in the longitudinal plane.

the same way, the evolution of beam particle density and the cloud and beam potentials can be examined during the passage of the beam through an electron cloud.

CONCLUSIONS AND OUTLOOK

HEADTAIL has been extended by a new class to handle data i/o operations in HDF5 format. The HDF5 data format is able to endow complex data with a logical structure so that it can be efficiently analysed with readily available third party software. For HEADTAIL the data structure has been chosen to roughly follow the fiber bundle concept, making the data layout reusable and adaptable for different applications. In this layout it is also easy to add an XDMF layer to enable 3D visualisation in Paraview.

The HEADTAIL output files that support HDF5 are the bunch statistics file (*_prt.h5), the slice statistics file (*_hdtl.h5), the 6D phase space file (*_prt.h5) and a

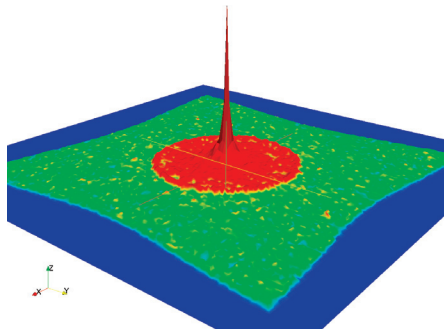

Figure 5: 2D representation of an electron cloud pinch. The z-axis (vertical) and the colouring qualitatively represent the cloud density.

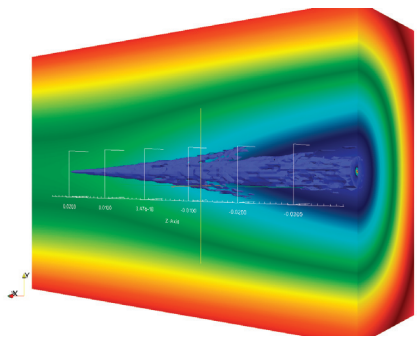

Figure 6: 3D representation of an electron cloud pinch. The z-axis is mapped to the bunch slice positions. The cone shows the cloud density evolution while clip plot illustrates the cloud potential evolution.

file containing the output from the PIC solver (*_pic.h5).

Examples highlight potential uses of the output format. What has not been mentioned are the gains in performance due to the binary data format when reading and analysing large amounts of output files from batch jobs. Furthermore, future versions will include native data compression by using chunked storage. In addition, the H5Part data structure will be dropped in favour of XDMF which provides a higher flexibility as well as conformance among the output files.

REFERENCES

[1] G. Rumolo and F. Zimmermann, *Practical User Guide for HEADTAIL*, CERN-SL-Note-2002-036, 2002.

[2] *The HDF5 Home Page*, http://www.hdfgroup.org/HDF5/.

[3] *HDFView*, http://www.hdfgroup.org/hdf-java-html/hdfview/.

[4] *ParaView*, http://www.paraview.org/.

[5] *XDMF*, http://www.xdmf.org/index.php/Main_Page/.

[6] *Kitware, Inc.*, http://www.kitware.com/news/home/browse/.

[7] Ren Heinzl, *Concepts for Scientific Computing*, Dissertation, Technische Universitt Wien, 2007.

[8] *H5Part: a Portable High Performance Parallel Data Interface to HDF5*, http://vis.lbl.gov/Research/H5Part/.

Copyright © 2012 by IEEE – cc Creative Commons Attribution 3.0 (CC BY 3.0) — cc Creative Commons Attribution 3.0 (CC BY 3.0)

ISBN 978-3-95450-115-1