Symplectified ZMaps in the ZLIB++ Library

Wolfram Fischer

March 7, 1997

Abstract

The ZLIB++ library [1,2] is a C++ library that has been developed as a tool for particle accelerator problems. ZMaps are objects in the ZLIB++ library that may describe phase space transformations with truncated Taylor series. They can be a model for the passage through an accelerator element or a whole ring. But in general, the mapping provided by a truncated Taylor series map is non-symplectic and many repeated iterations, as in dynamic aperture studies, can lead to a significant amplitude error.

Cremona maps are both polynomial and symplectic [3]. Truncated Taylor series maps may be approximated by Cremona maps which can then be used in dynamic aperture studies. This note describes how this is done in the framework of the ZLIB++ library.

1 Introduction

The particle motion in accelerators can be conveniently described in an ortho-normal coordinate system relative to a design path, where the positions are denoted by $\vec{x}$ and the conjugate momenta by $\vec{p}$. The phase space of the motion may have n dimensions. Of practical interest are the cases $n = 4$ (coasting beam) and $n = 6$ (bunched beam). The path length s along the design orbit may serve a as time-like variable.

The action of an accelerator element, like a magnet, can be viewed as a mapping of the initial coordinates $\vec{v}^i = (\vec{x}^i, \vec{p}^i)$ at the entrance of the element s^i onto the final coordinates $\vec{v}^f = (\vec{x}^f, \vec{p}^f)$ at the exit of the element s^f:

$$\vec{v}^f = M\ \vec{v}^i. \tag{1}$$

If the motion is governed by Hamiltonian mechanics, as for hadron machines in the energy range accessible today, the mapping (1) must be symplectic, i.e. the Jacobian matrix

$$J(s^f, s^i) = \frac{\partial \vec{v}(s^f)}{\partial \vec{v}(s^i)} \tag{2}$$

must fulfill the condition

$$J^T S J = S \qquad \text{with} \qquad S = \begin{pmatrix} 0_n & I_n \\ -I_n & 0_n \end{pmatrix} \tag{3}$$

for all pairs of positions (s^f, s^i). J^T denotes the transposed Jacobian matrix and S the symplectic form. 0_n and I_n are the zero and unity matrix of order $n/2$.

The map M may take the special form of a Taylor series representation so that

$$v_j^f = T^0 + \sum_{m=1}^{n} T_{jm}^1 v_m^i + \sum_{m_1=1}^{n} \sum_{m_2=1}^{n} T_{jm_1m_2}^2 v_{m_1}^i v_{m_2}^i + \ \ldots \tag{4}$$

However, for any practical purpose, the series has to be terminated at a certain order k which will lead, in general, to a nonsymplectic coordinate transformation M_k. Truncated Taylor series maps are implemented in the `ZLIB++` library as `ZMaps`. Along with a data structure that holds the coefficients $(T^0, T^1, \ldots, T^k)$ there are methods to manipulate `ZMaps` and track particles through them [2].

But there are truncated Taylor maps of limited order k that are symplectic. Combinations of these maps can be used to approximate a general truncated Taylor map. Such a procedure is described in Ref. [4] and in more general terms in Ref. [3]. The result, a map that is both polynomial and symplectic, is usually called a Cremona map.

In the following the basic idea of Cremona map approximations are presented following Ref. [3]. The class `ZJMap`, that represents Cremona maps in the `ZLIB++` library, is described as well as the program `ZCremona` which transforms a `ZMap` into a `ZJMap`.

2 Cremona Maps

In this section, Lie transformations and Lie operators are defined. A theorem is presented that relates a truncated Taylor series maps to a factored representation of Lie transformations. Jolts are introduced as special generating functions for Lie operators. With jolts Cremona approximations of truncated Taylor maps can be constructed.

2.1 Lie Transformations and Lie Operators

If f and g are functions of the canonical phase space variables $(\vec{x}, \vec{p})$, one can define a Lie operator $: f :$ generated by the function f by its action on g as

$$: f : \ g \ = [f, g] = \frac{\partial f}{\partial \vec{x}} \frac{\partial g}{\partial \vec{p}} - \frac{\partial f}{\partial \vec{p}} \frac{\partial g}{\partial \vec{x}}. \tag{5}$$

A Lie transformation $e^{:f:}$ can then be defined by

$$e^{:f:} g = \sum_{n=0}^{\infty} \frac{1}{n!} : f :^n g, \tag{6}$$

where a power of $: f :$ has to be understood as the successive application of the operator.

Special functions g are the phase space coordinates v_j (e.g. $g(\vec{x}, \vec{p}) = x_1$) and a Lie transformation (6) applied to all coordinates is always a symplectic mapping [5]. However, the series in (6) does, in general, not terminate, so that there is again a practical difficulty in dealing with Lie transformations.

An important property of Lie transformations and Lie operators, which is used later, is [5]

$$: e^{:f:} g : = \ e^{:f:} : g : e^{-:f:}. \tag{7}$$

2.2 Factored Product Representation of Taylor Maps

The following theorem is given without proof (cf. Refs. [3,5,6]): If M_k is the truncated Taylor series representation of a symplectic map, then there is a uniquely determined factored product form

$$N_{k+1} = e^{:f_1:} e^{:f_2^c:} e^{:f_2^a:} e^{:f_3:} \ \ldots \ e^{:f_{k+1}:} \tag{8}$$

the Taylor series expansion of which agrees with M_k up to order k. The f_m denote homogeneous polynomials of degree m in the variables v_j and the quadratic polynomials f_2^c and f_2^a have the special form

$$f_2 = -\frac{1}{2} \sum_{m_1=1}^{n} \sum_{m_2=1}^{n} C_{m_1 m_2} v_{m_1} v_{m_2} \tag{9}$$

where C denotes a symmetric matrix. For f_2^c the matrix C commutes with the matrix S, and for f_2^a C anti-commutes with S. f_1 denotes the constant, f_2 the linear and the f_m with $m \geq 3$ the nonlinear part of the map. Note, that a Taylor map of order k corresponds to a factored product representation with polynomials up to order $k+1$. This follows from the fact the a Lie transformation uses is the derivative operator (5).

2.3 Jolts and Jolt Maps

Following Ref. [3] a jolt is defined as any dynamical function g that satisfies

$$: g :^2 v_j = 0 \qquad j = 1, \ldots n \tag{10}$$

The Lie transformation generated by g is then

$$v_j^f = e^{:g:} v_j^i = v_j^i + : g : v_j^i = v_j^i + [g, v_j^i] \tag{11}$$

which is called a jolt map.

The important point here is that by having the property (10) and using g as a generating function for a Lie transformation, the series in (6) terminates. Therefore, one has an exact representations for a Lie transformation and thereby a symplectic coordinate transformation.

Now note that if g is a function of the coordinates $\vec{x}$ alone or the momenta $\vec{p}$ alone then the functions $g(\vec{x})$ and $g(\vec{p})$ are jolts. Lie operators with generating functions $g(\vec{x})$ only affect the momenta and are called kicks. The associated maps are called kick maps[1] [4].

[1]Lie operators with generating function $g(\vec{p})$ appear to be unphysical since they only affect the positions and thereby lead to discontinuous paths in physical space. However, there may be applications for helical dipole maps since a full helical dipole can be approximated by a like drift plus a shift of the position [7].

Denoting a linear symplectic transformation $e^{:f_2}$ by L and using the property (7) one can see that that any linear transformation of a jolt is also a jolt:

$$: Lg :^2 v_j = (L : g : L^{-1})^2 v_j = L : g : L^{-1} L : g : L^{-1} v_j = L : g :^2 L^{-1} v_j = 0. \tag{12}$$

The corresponding Lie transformation has the form

$$e^{:Lg:} = e^{L:g:L^{-1}} = L e^{:g:} L^{-1}. \tag{13}$$

Kicks and linear transformations will be used to obtain Cremona approximations of truncated Taylor maps.

2.4 Jolt Representation

If one could approximate the nonlinear part of the truncated Taylor series map M_k by an expression of the form

$$J_k = L_1 e^{:g_1:} L_1^{-1} \ldots L_N e^{:g_N:} L_N^{-1} \tag{14}$$

in which all g_m are jolts and the Taylor series expansion of which agrees with M_k through order k, one would have a Cremona approximation of M_k since all factors $e^{:g_m:}$ are symplectic transformations and all factors L_m linear transformations.

In Ref. [3] is shown how to accomplish that task. N and the L_m are chosen in advance and in an order-by-order procedure the kicks g_m are determined. To get an agreement between M_k and J_k up to order k a minimum number of N is required which depends on k. Tab. 1 shows what the number N is that is used by the program `ZCremona` (not necessarily the minimum).

Table 1: Number of jolts used by `ZCremona`.

order k	number of jolts N
3	12
4	12
5	24
6	24
7	32
8	32

3 Software Description

The `ZLIB++` library uses a number of global parameters which have to be set in a program before objects are declared.

ZLIB_DIM	number of phase space dimensions n
ZLIB_ORDER	order k of Taylor series expansion, for the use of ZJMaps this number has to be by one larger than the order of the ZMap from which is was constructed
ZJOLT_DIM	number of jolts N
ZMAP_6DC	only usable for ZLIB_DIM=6, default value is zero, ZMAP_6DC=1 indicates that the 6th phase space variable is constant (increases tracking speed)

The last two parameters have been introduced recently by the author of this report.

3.1 The ZJMap Class

A ZJMap is the representation of a Cremona map in the ZLIB++ library. It is best described by its definition:

```
class ZJMap : public ZMap
{
public:
  ZJMap();
  ZJMap(ZJMap& JM);

  friend ostream& operator<<(ostream& out, ZJMap& JM);
  friend istream& operator>>(istream& in,  ZJMap& JM);
  Particle& operator*(Particle& P);

  ~ZJMap();

protected:
  ZMap     C;      // constant part
  ZMap     A2i;    // linear coordinate transformation
  ZMap     A2;     // inverse coordinate transformation to A2i
  ZMap     R;      // rotation
  ZSeries* g;      // array of jolts
  ZMap*    M;      // array of maps corresponding to jolts
};
```

The ZMap C is of order zero and contains the constant part of the mapping. The ZMap A2i is of order one and is a transformation into a coordinate system in which the linear part of the map is a simple rotation. All the following computations will be carried out in this coordinate system. The ZMap A2 transforms the coordinates back into the original system. The ZMap R is of order two and describes the rotation. The ZSeries' g are the jolts and the ZMaps M are their associated maps.

The elements C, A2i, A2, R and the jolts g can be stored in a file using the << operator. ZJMaps can be read in with the >> operators. Multiplication of ZJMap with an object of the class Particle performs the tracking. In this case, the ZMaps A2i,R,M[1],..., M[ZJOLT_DIM], A2,C are applied successively. The use of the ZMAP_6DC flag is optional.

3.2 ZCremona

The program ZCremona is adapted from a program provided by D. Abell to compute Cremona symplectifications for truncated Taylor series maps. Abell's program uses the LIELIB library [8]. For use with the ZLIB++ library, the input and output format has been adapted to ZLIB++ data format. The input and output files for ZCremona have fixed names:

```
input  file: ZMap.in     map, ZLIB++ format
output file: ZJMap.out   jolt map, ZLIB++ format
output file: LJMap.out   jolt map, LIELIB format
output file: LJMap.test  jolt map test, LIELIB format
```

To compute the Cremona approximation of a ZMap, one has to copy the file containing the ZMap to ZMap.in and run the program ZCremona. The files ZJMap.out and LJMap.out contain the Cremona map approximation of the input map in ZLIB++ and LIELIB format respectively. The file LJMap.test holds test results which are explained in the file itself.

3.3 Example

The modeling of the particle motion in helical magnetic fields may serve as an example. Example computations were carried out and the results can be visited in

```
/rap/PAC++/example
```

To produce a 4th order Taylor map of an ideal helical dipole magnet in 6 variables (see Ref. [9]), which will be stored in the file helix_d6_o4.ZMap run

```
helix -d6 -o4 ideal.in > ZMap_d6_o4.ZMap
```

To produce a Cremona map from this map execute

```
cp helix_d6_o4.ZMap ZMap.in
ZCremona
cp ZJMap.out helix_d6_o4.ZJMap
```

The program Ztest reads a ZJMap and tracks a particle through the map 10 times. Its syntax is

```
Ztest helix_d6_o4.ZJMap
```

3.4 Computing Times

In Tab. 2 the CPU times needed to produce ZMaps and ZJMaps are listed for various orders and numbers or variables. The time needed to track a 1000 times through those maps is also listed.

Table 2: CPU times for various maps on a SUN Ultra Sparc.

number of variables	order	`helix` [hh:mm:ss]	`ZCremona` [hh:mm:ss]	1,000 turns `ZMap` [s]	1,000 turns `ZJMap` [s]
4	3	3:26	3	0.4	2.0
4	4	7:25	7	0.8	4.3
4	5	16:06	30	1.4	7.4
4	6	37:27	1:12	2.3	12.2
4	7	1:09:22	3:24	3.5	19.9
4	8	2:15:45	7:23	5.5	32.0
6	3	7:42	8	1.1	4.1
6	4	24:15	24	2.5	9.0
6	5	1:12:29	1:58	5.4	18.4
6	6	3:24:47	5:21	10.4	35.0
6	7	8:50:07	16:53	19.7	60.7
6	8	21:36:27	40:13	33.8	97.7

4 Summary

The basic ideas of Cremona maps have been presented. They are implemented as `ZJMaps` in the `ZLIB++` library and can be obtained from `ZMaps` with the program `ZCremona`. In principle, `ZJMaps` can be used in the same way as `ZMaps` in computer codes which are part of the `PAC++` package [10]. Of particular interest will be their use in the tracking program `Teapot` [11].

5 Acknowledgment

I am thankful to N. Malitsky for comments on the `ZLIB++` library, to D.T. Abell for providing the symplectification software and to F. Pilat for reading the manuscript.

References

[1] Y. Yan and Ch.-Y. Yan, "ZLIB - numerical library for differential algebra", SSCL-300 (1990).

[2] N. Malitsky and A. Reshetov, "ZLIB++: object-oriented numerical library for differential algebra", SSCL-659 (1994).

[3] D.T. Abell, "Analytic properties and Cremona approximations of transfer maps for Hamiltonian systems", PhD thesis, University of Maryland (1995).

[4] J. Irwin, "A multi-kick factorization algorithm for nonlinear maps", SSC-228 (1989).

[5] A.J. Dragt, "Lectures on nonlinear orbit dynamics", in AIP Conference Proceedings, Vol. 87 (1982).

[6] E. Forest, M. Berz and J. Irwin, "Normal form methods for complicated periodic systems: a complete solution using differential algebra and lie operators", Part. Acc. 24, 91 (1989).

[7] M. Syphers, "Closed orbit errors from helical dipole magnets", BNL AGS/RHIC/SN 16 (1996).

[8] E. Forest, "LIELIB FORTRAN library".

[9] N. Malitsky, "Application of a differential algebra approach to a RHIC helical dipole", BNL RHIC/AP/51 (1994).

[10] N. Malitsky, A. Reshetov and G. Bourianoff, "PAC++: object-oriented platform for accelerator codes", SSCL-675 (1994).

[11] N. Sun et al., "The linkage of Zlib to Teapot for auto-differentiation map extraction and nonlinear analysis", SSCL-Preprint-474 (1993).