NEW ALGORITHMS IN ZGOUBI[*]

Dan Tyler Abell[†], RadiaSoft LLC, Boulder, CO, USA
François Méot, Brookhaven National Lab, Upton, NY, USA

Abstract

The particle tracking code Zgoubi [1,2] is used for a broad array of accelerator design studies, including FFAGs [3] and EICs [4,5]. In this paper, we describe recent work aimed at improving Zgoubi's speed and flexibility. In particular, we describe a new implementation of the Zgoubi tracking algorithm that requires significantly less memory and arithmetic. And we describe a new algorithm that performs symplectic tracking through field maps. In addition, we describe the current efforts to parallelize Zgoubi.

ZGOUBI'S PARTICLE UPDATE

Zgoubi was originally developed in the 1970s as a spectrometer code. This heritage explains it's focus on, and capabilities for, detailed particle integration in spatially-varying magnetic fields. Zgoubi's particle update algorithm integrates the Lorentz force equation, $\mathrm{d}\vec{p}/\mathrm{d}t = q(\vec{E} + \vec{v}\times\vec{B})$, for a charged particle in electric field $\vec{E}$ and magnetic field $\vec{B}$. For the independent variable, however, it uses *distance s along the particle trajectory*. Using a prime ($'$) to denote differentiation with respect to s, defining the *normalized velocity* $\vec{u} = \vec{v}/v$, and expressing the particle momentum as

$$\vec{p} = m\gamma\vec{v} = q(B\rho)\vec{u}, \tag{1}$$

where $(B\rho)$ denotes the usual magnetic rigidity, Zgoubi writes the Lorentz force Law in the form

$$\frac{\mathrm{d}}{\mathrm{d}s}(B\rho)\vec{u} = (B\rho)'\vec{u} + (B\rho)\vec{u}' = \frac{1}{v}\vec{E} + \vec{u}\times\vec{B}. \tag{2}$$

Using this equation together with derivatives of the known electric and magnetic fields, Zgoubi can construct the sequence of derivatives $(B\rho)'$, $\vec{u}'$, $(B\rho)''$, $\vec{u}''$, *etc.* Zgoubi then uses these derivatives to update both position $\vec{r}$, and velocity $\vec{u}$ according to the Taylor series approximation

$$\vec{r}^{f} \approx \vec{r} + \Delta s\,\vec{u} + \frac{\Delta s^2}{2!}\vec{u}' + \cdots + \frac{\Delta s^6}{6!}\vec{u}^{(5)}, \tag{3a}$$

$$\vec{u}^{f} \approx \vec{u} + \Delta s\,\vec{u}' + \frac{\Delta s^2}{2!}\vec{u}'' + \cdots + \frac{\Delta s^5}{5!}\vec{u}^{(5)}. \tag{3b}$$

The hierarchy of computations leading to the derivatives $(B\rho)^{(n)}$ and $\vec{u}^{(n)}$ is very much like that used in the computations in truncated power series algebra (TPSA) [6, ch.39]. As a consequence, we can reimplement Zgoubi's particle update algorithm in a manner that reduces its memory footprint by a factor of about three, and the arithmetic involved in the particle update by a similar factor. In addition, a TPSA-like update makes it easy to remove deeply nested conditionals that interfere with performance. We illustrate the idea for motion in a purely magnetic field.

In the absence of an electric field, (2) reduces to the particularly simple form

$$\vec{u}' = \vec{u}\times\vec{b}, \tag{4}$$

where $\vec{b}$ denotes the magnetic field scaled by the magnetic rigidity:

$$\vec{b} = \frac{1}{B\rho}\vec{B}. \tag{5}$$

Successive s derivatives of $\vec{u}$ are computed as

$$\vec{u}'' = \vec{u}'\times\vec{b} + \vec{u}\times\vec{b}', \tag{6a}$$

$$\vec{u}''' = \vec{u}''\times\vec{b} + 2\vec{u}'\times\vec{b}' + \vec{u}\times\vec{b}'', \tag{6b}$$

etc., and successive s derivatives of $\vec{b}$ are computed as

$$\vec{b}' = \sum_i \frac{\partial\vec{b}}{\partial x_i}u_i, \tag{7a}$$

$$\vec{b}'' = \sum_{ij}\frac{\partial^2\vec{b}}{\partial x_i\partial x_j}u_i u_j + \sum_i \frac{\partial\vec{b}}{\partial x_i}u_i', \tag{7b}$$

etc. Zgoubi then computes the $\vec{b}^{(n)}$ exactly as indicated: For example, $\partial^2\vec{b}/\partial x_i\partial x_j$ is stored as three two-dimensional arrays; $\partial^4\vec{b}/\partial x_i\partial x_j\partial x_k\partial x_l$ is stored as three four-dimensional arrays; and so on. This approach greatly simplifies checking the implementation: it is an easy matter to render the math directly in code. But the many symmetries in those multidimensional arrays means they are wasteful of both space and computational effort. Zgoubi takes advantage of those symmetries when populating the arrays, but not when computing the expressions for the $\vec{b}$ derivatives in (7). Moreover, the sum over elements in a four-dimensional array involves a deep nesting of loops.

One can instead store the collection of multivariate $\vec{b}$ derivatives as a linear array using Giorgilli indexing, exactly as is done in standard TPSA algorithms [6, ch.39]. In addition, the terms u_i', $u_i u_j$, and the like can be stored in a similarly indexed linear array, so that as a consequence the computations in (7) become easily optimized dot products between pairs of vectors. This is similar to the evaluating a truncated power series as a dot product between a vector of coefficients and a vector of monomials. The only difference is that here what corresponds to entries in the vector of monomials—*e.g.* the $\{u_i', u_i u_j\}$—must be constructed to include appropriate integer factors that account for the symmetries in the coefficients—*e.g.* the $\{\partial\vec{b}/\partial x_i, \partial^2\vec{b}/\partial x_i\partial x_j\}$.

If electric, or both electric and magnetic, fields are present, the computation of the $u^{(n)}$ is more complicated, but the conclusion remains that refactoring the computation in TPSA-like form yields significant computational benefits.

[*] This work is supported in part by the US Department of Energy, Office of Science, Office of Nuclear Physics, including grant No. DE-SC0017181
[†] dabell@radiasoft.net

PARALLELIZE ZGOUBI

The most obvious problem of Zgoubi is that it is a serial code with little or no data parallelism. Developing a parallel code is critical for exploiting modern architectures. Fortunately, the recent Fortran standards have paved the way for us with the introduction of Coarray Fortran (CAF) [7]. First developed as an extension to Fortran 95, and now part of the Fortran 2008 standard, CAF uses the Single Program Multiple Data (SPMD) model of parallelization. A CAF program launches a fixed (determined at run-time) number of copies, called *images*, that run asynchronously. When declared in the code, a *coarray* is a data structure whose data may be shared between images. One declares a coarray simply by appending square brackets to define the array's *codimension*:

```
    real, allocatable :: x(:,:,:)[:]
```

declares an allocatable array with three regular dimensions and a single codimension. Access to data on a remote image is then gained by reference to that image's codimension.

In the context of single-particle dynamics, the use of coarrays is required only in those parts of Zgoubi that require access to data on remote images. These are the data reduction operations—so, in the case of Zgoubi, the computations of beam emittance, beam polarization, and the like. For such computations, the CAF standard provides functions to ensure data synchronization, analogous to barriers in MPI.

Case studies of modernizing and parallelizing Fortran 77 with CAF have shown that the resulting code exhibits good scalability after optimization [8]. CAF has also proven to be generally easy to write and maintain as compared to other parallelization options [9].

SYMPLECTIC TRACKING

Of course another approach to improving Zgoubi's performance involves replacing Zgoubi's particle update with traditional symplectic integrators such as those described in Forest's 2006 review paper [10]. Those integrators have been generalized to include spin [11]. We plan to implement those integrators as *options* for the standard accelerator elements in Zgoubi. Retaining Zgoubi's traditional particle update algorithm (albeit in a new, TPSA-like form) means that users will still have access to Zgoubi's detailed fringe-field models, but they can use the optional fast integrators where and when occasion warrants. Moreover, this approach will make it easy to perform comparisons that test the impact of the different integrators.

SYMPLECTIC FIELD MAPS

One can derive symplectic tracking algorithms from both Hamiltonian and Lagrangian perspectives. Because Zgoubi tracks the (normalized) velocity, we choose the Lagrangian perspective. Marsden and West [12] give a useful (if dense) introduction to this approach. The basic idea is to identify the Lagrangian, write a discrete form of the action integral, and then apply a discretized form of the Euler-Lagrange equations. The result of this process is a set of coupled difference equations that describe the particle update. Moreover, the resultant particle update is automatically symplectic. In very simple cases, one can obtain an explicit algorithm, but in most cases the result is implicit.

For a relativistic charged particle in a general static magnetic field, the Lagrangian (here with time as the independent variable) has the form

$$L = -mc^2\sqrt{1-\vec{v}^{\,2}/c^2} + q\,\vec{v}\cdot\vec{A}(\vec{r}). \tag{8}$$

A corresponding discrete form for the action integral covering a time step of size h from $\vec{r}^{\,0}$ to $\vec{r}^{\,1}$ is given by

$$\begin{aligned}\mathcal{A}_d[\vec{r}^{\,0},\vec{r}^{\,1};h] = -h\,mc^2\sqrt{1-\frac{1}{c^2}\left(\frac{\vec{r}^{\,1}-\vec{r}^{\,0}}{h}\right)^2}\\ + q\left(\frac{\vec{r}^{\,1}-\vec{r}^{\,0}}{h}\right)\cdot\frac{h}{2}\left(\vec{A}(\vec{r}^{\,0})+\vec{A}(\vec{r}^{\,1})\right).\end{aligned} \tag{9}$$

The *discrete Euler-Lagrange* (dEL) equations can be written in the form

$$-D_1\mathcal{A}_d[\vec{r}^{\,k},\vec{r}^{\,k+1};h] = D_2\mathcal{A}_d[\vec{r}^{\,k-1},\vec{r}^{\,k};h], \tag{10}$$

where D_1 denote differentiation with respect to the *first* argument of $\mathcal{A}_d$, and D_2 denote differentiation with respect to the *second* argument. The result of applying (10) to (9) is a set of implicit equations that one can solve for $\vec{r}^{\,k+1}$ in terms of $\vec{r}^{\,k-1}$ and $\vec{r}^{\,k}$. This defines a symplectic particle update.

We applied our new symplectic particle update to the Störmer problem—the motion of a charged particle in Earth's magnetic dipole field [13–15]. This problem has significant nonlinearity, and the vector potential is known analytically, making it a useful test case. In Figure 1 we show a three-dimensional plot of the trajectory formed by a single 10 MeV proton over the course of about 1600 cyclotron periods. In Figure 2 we show the error in the computed value of the relativistic γ factor. The small oscillatory variation is typical of symplectic integrators. Unfortunately, this approach does not work well at higher energies. In the case of

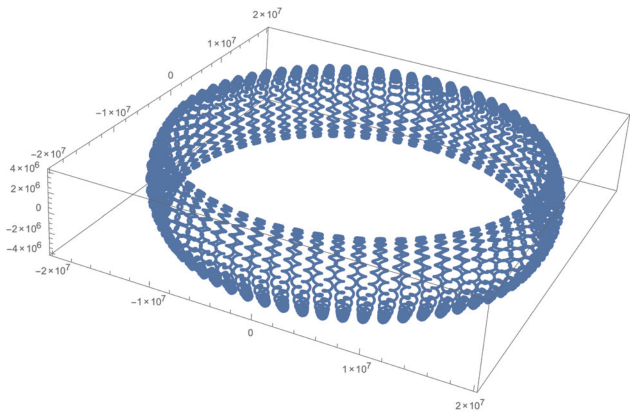

Figure 1: Trajectory of a 10 MeV proton launched about three Earth radii from the center with an initial velocity pitched upwards about 24° out of the equatorial plane.

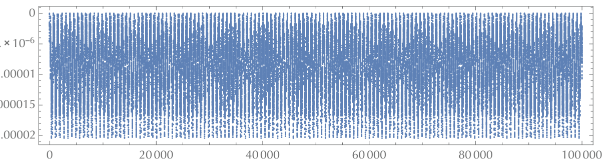

Figure 2: Variation in the relativistic γ for the 10 MeV proton simulation shown in Figure 1.

static magnetic fields, the cure for this difficulty turns out to be rather simple: Use the "non-relativistic" Lagrangian

$$L = \frac{1}{2} m^* \vec{v}^2 + q\,\vec{v} \cdot \vec{A}(\vec{r}), \tag{11}$$

with $m^* \equiv m\gamma$. This Lagrangian also yields the correct relativistic equation of motion. The corresponding symplectic particle update derived from the discrete Euler-Lagrange equation is much better behaved. In Figure 3, we compare this new version of the dEL equations with the older version.

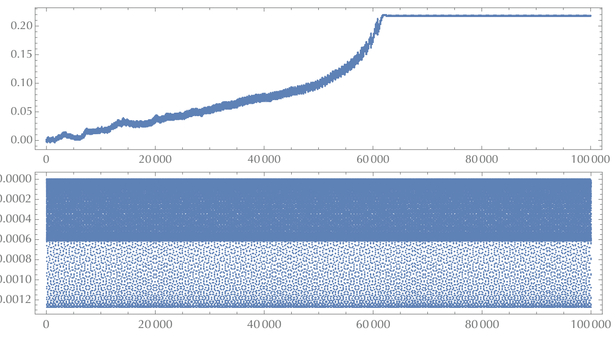

Figure 3: Error in the relativistic γ for a 400 MeV proton gyrating in the Earth's magnetic dipole field. The upper graphic shows results obtained using the dEL equations derived from the relativistic Lagrangian (8). The lower graphic used dEL equations derived from the "non-relativistic" Lagrangian (11).

To fully implement symplectic tracking for field maps, we must extract the vector potential from magnetic field data. The Helmholtz decomposition theorem together with the fact that a static magnetic field is both solenoidal and irrotational allows us to write $\vec{A}$ as the sum of a single pair of surface integrals [6, ch.22]:

$$\vec{A}^t(\vec{r}) = \frac{1}{4\pi} \oint_S \vec{G}(\vec{r},\vec{r}\,')\,[\vec{B}(\vec{r}\,') \times \mathrm{d}\vec{a}\,'], \tag{12}$$

$$\vec{A}^n(\vec{r}) = \frac{1}{4\pi} \oint_S \vec{K}(\vec{r},\vec{r}\,';\vec{n}\,')\,[\vec{B}(\vec{r}\,') \cdot \mathrm{d}\vec{a}\,']. \tag{13}$$

Here the kernels $\vec{G}$ and $\vec{K}$ are given by the relations

$$\vec{G}(\vec{r},\vec{r}\,') = \frac{1}{|\vec{r}-\vec{r}\,'|}, \tag{14a}$$

$$\vec{K}(\vec{r},\vec{r}\,';\vec{n}\,') = \frac{\vec{n} \times (\vec{r}-\vec{r}\,')}{|\vec{r}-\vec{r}\,'|\,[|\vec{r}-\vec{r}\,'| - \vec{n}\,' \cdot (\vec{r}-\vec{r}\,')]}, \tag{14b}$$

and $\vec{n}\,'$ denotes an outward pointing unit vector at the point $\vec{r}\,'$ on the surface S. Because $\vec{K}$ denotes the vector potential of a Dirac monopole, and because integration is linear, the integral $\vec{A}^n$ automatically obeys Maxwell no matter how well or poorly we evaluate the integral. This property, however, is not true for the term $\vec{A}^t$ as written in the form (12). But one may rewrite $\vec{A}^t$ in the alternate form [6, ch.22]

$$\vec{A}^t(\vec{r}) = \frac{1}{4\pi} \oint_S \frac{\vec{n}\,' \times (\vec{r}-\vec{r}\,')}{|\vec{r}-\vec{r}\,'|^3}\,\psi(\vec{r}\,'), \tag{15}$$

where ψ denotes the corresponding magnetic scalar potential. It can be shown that this form for $\vec{A}^t$ obeys Maxwell independent of how well or poorly we evaluate the integral (15).

The last piece piece of this puzzle, then, is that we must gain access to the magnetic scalar potential ψ over the surface S. Because the accuracy with which we know the scalar potential does not affect the conclusion that our numerically computed vector potential yields a Maxwellian magnetic field, it can suffice to compute reasonable numerical values for ψ on the basis of a multipole expansion that is fitted to the known surface fields. Of course it is far too costly to evaluate the actual integrals (13) and (15) at the location of each particle. Because the symplectic tracking algorithm also requires the matrix of derivatives $\partial_i A_j$—and needs that information in a form that is sufficiently smooth—the best approach will be to compute the information (13) plus (15) on an appropriate lattice within the range of interest, and then use cubic splines, or similar, to compute the values and derivatives of $\vec{A}(\vec{r})$ that are required during tracking.

The computation of the integrals (13) plus (15), and the subsequent spline interpolation needs to be done just once for any given magnet. It can therefore be done "offline"—*i.e.* outside a given simulation—and the results stored for later use.

REFERENCES

[1] F. Méot, "Zgoubi." `http://sourceforge.net/projects/zgoubi/`

[2] F. Méot, "Zgoubi users' guide," Tech. Rep. FERMILAB-TM-2010, Fermi National Accelerator Laboratory, Batavia, IL, Oct. 1997.

[3] F. Lemuet and F. Méot, "Developments in the ray-tracing code zgoubi for 6-D multiturn tracking in FFAG rings," *Nucl. Instrum. Methods Phys. Res., Sect. A*, vol. 547, pp. 638–651, Aug. 2005, `doi:10.1016/j.nima.2005.03.156`

[4] F. Méot, S. Brooks, V. Ptitsyn, D. Trbojevic, and N. Tsoupas, "Tracking studies in eRHIC energy recovery recirculator," Tech. Rep. BNL-108286-2015-IR (eRHIC/45), Brookhaven National Laboratory, Upton, NY, July 2015.

[5] F. Lin, Ya. S. Derbenev, V. S. Morozov, Y. Zhang, and D. P. Barber, "Study of electron polarization dynamics in the JLEIC at JLab," in *Proc. IPAC'15*, Richmond, Virginia, USA, May 2015, pp. 3218–3220.

[6] A. J. Dragt, "Lie methods for nonlinear dynamics with applications to accelerator physics", Latest version available at `http://www.physics.umd.edu/dsat/`, Mar. 2018.

[7] J. Reid, "Coarrays in the next Fortran standard", Eprint `https://wg5-fortran.org/N1801-N1850/N1824.pdf`, Apr. 2010.

[8] H. Radhakrishnan, D. W. I. Rouson, K. Morris, S. Shende, and S. C. Kassinos, "Using coarrays to parallelize legacy Fortran applications: Strategy and case study," *Scientific Programming*, Jan. 2015, doi:10.1155/2015/904983

[9] S. Garain, D. S. Balsara, and J. Reid, "Comparing coarray fortran (CAF) with MPI for several structured mesh PDE applications," *J. Comput. Phys.*, vol. 297, pp. 237–253, 2015, doi:10.1016/j.jcp.2015.05.020

[10] É. Forest, "Geometric integration for particle accelerators," *J. Phys. A: Math. Gen.*, vol. 39, pp. 5321–5377, May 2006, doi:10.1088/0305-4470/39/19/S03

[11] D. T. Abell, D. Meiser, V. H. Ranjbar, and D. P. Barber, "Accurate and efficient spin integration for particle accelerators," *Phys. Rev. ST Accel. Beams*, vol. 18, p. 024001, Feb. 2015, doi:10.1103/PhysRevSTAB.18.024001

[12] J. E. Marsden and M. West, "Discrete mechanics and variational integrators," *Acta Numer.*, vol. 10, pp. 357–514, May 2001, doi:10.1017/S096249290100006X

[13] A. J. Dragt, "Trapped orbits in a magnetic dipole field," *Rev. Geophys.*, vol. 3, no. 2, pp. 255–298, May 1965, doi:10.1029/RG003i002p00255

[14] A. J. Dragt, "Correction to 'Trapped orbits in a magnetic dipole field'," *Rev. Geophys.*, vol. 4, no. 1, p. 112, Feb. 1966, doi:10.1029/RG004i001p00112

[15] A. J. Dragt and J. M. Finn, "Insolubility of trapped particle motion in a magnetic dipole field," *J. Geophys. Res.*, vol. 81, no. 13, pp. 2327–2340, May 1976, doi:10.1029/JA081i013p02327