CONSTRUCTION OF LARGE-PERIOD SYMPLECTIC MAPS BY INTERPOLATIVE METHODS *

Robert L. Warnock † and Yunhai Cai‡
SLAC National Accelerator Laboratory, Stanford University, Menlo Park, CA 94025, USA
James A. Ellison§
Dept. of Mathematics and Statistics, University of New Mexico, Albuquerque, NM 87131, USA

Abstract

The goal is to construct a symplectic evolution map for a large section of an accelerator, say a full turn of a large ring or a long wiggler. We start with an accurate tracking algorithm for single particles, which is allowed to be slightly non-symplectic. By tracking many particles for a distance S one acquires sufficient data to construct the mixed-variable generator of a symplectic map for evolution over S, given in terms of interpolatory functions. Two ways to find the generator are considered: (i) Find its gradient from tracking data, then the generator itself as a line integral. (ii) Compute the action integral on many orbits. A test of method (i) has been made in a difficult example: a full turn map for an electron ring with strong nonlinearity near the dynamic aperture. The method succeeds at fairly large amplitudes, but there are technical difficulties near the dynamic aperture due to oddly shaped interpolation domains. For a generally applicable algorithm we propose method (ii), realized with meshless interpolation methods.

1. INTRODUCTION

The method of differential algebra, giving automatic differentiation of functions defined by complex algorithms, allows the construction of the truncated Taylor series of a map defined by a tracking code. After this method was implemented by Martin Berz [1], the option of producing a Taylor map eventually became a feature of several tracking codes. The Taylor map is not symplectic, but some codes use the Taylor coefficients to produce the mixed-variable generator of a symplectic map, itself represented as a truncated power series [2]. Another way is to use the Taylor coefficients to form the symplectic "jolt factorization" of Irwin, Abell, and Dragt [3]. The Taylor map, symplectified or not, is good at small phase space amplitudes, but has a range of usefulness at larger amplitudes that varies with the type of accelerator lattice considered. It appears to be fairly useful for hadron rings, but can fail badly for electron rings with stronger nonlinearity near the dynamic aperture. In this paper we choose such a lattice for a demanding test of mapping methods.

* Work supported in part by U.S. Department of Energy contracts DE-AC02-76SF00515 and DE-FG-99ER41104.
† warnock@slac.stanford.edu; also affiliated with LBNL and UNM.
‡ yunhai@slac.stanford.edu
§ ellison@math.unm.edu

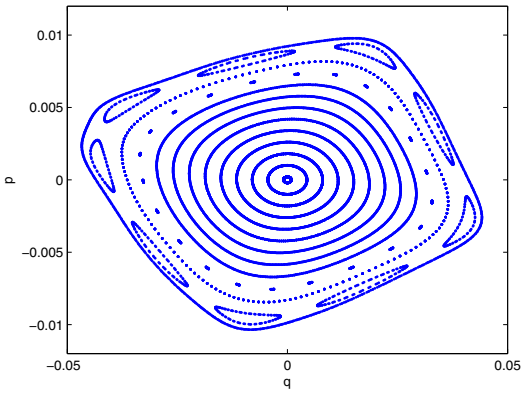

Figure 1: Phase plot from element-by-element tracking, 1000 turns, $\nu_x = 16.23$.

For our example the Taylor map fails at large amplitudes. For a striking illustration we choose a tune $\nu_x = 16.23$. Element-by-element tracking gives the plot of Fig.1 on a Poincaré section at a fixed position in the ring; p is dimensionless and q is in meters. The corresponding plot from iteration of the 10th order Taylor map is shown in Fig.2. The prominent 9th order island chain is only vaguely visible, and there is spurious stochasticity. The result is not improved by going to 13th order. The symplectified Taylor map [2] shows islands and gets rid of the stochasticity, but the shape of phase contours is all wrong. Changing to a better tune of $\nu_x = 15.81$, for which the lattice has a much larger dynamic aperture, we find that the Taylor map still breaks down at about the same amplitude.

One can easily see, however, that producing a more successful map when the Taylor series fails is not out of the question. A spline fit to one-turn tracking data on a grid of initial conditions gives a map which produces the plot of Fig.3 in 1000 iterations. To graphical accuracy it agrees with the tracking map of Fig.1, but the symplectic condition is badly violated at large amplitudes: the determinant of the map's Jacobian differs from 1 at some points by as much as 0.004. Nevertheless it does not do badly over 10^5 iterations, as is seen in Fig.4. The main features of the phase plot persist correctly, but fuzziness appears near ends of the islands. By 10^6 turns there is a clear failure, with spurious damping, whereas phase curves including the islands are sharply defined in tracking for 10^6 turns. The spline is a tensor product B-spline interpolating tracking

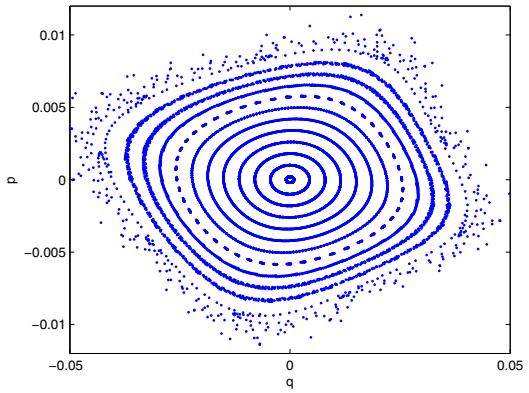

Figure 2: Phase plot from 10th order Taylor map, 1000 turns, $\nu_x = 16.23$.

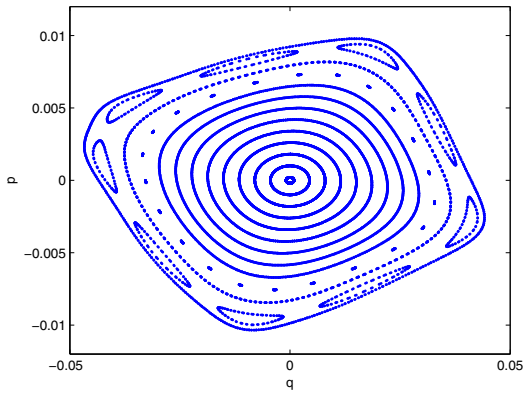

Figure 3: Phase plot from Spline Map, 1000 turns, $\nu_x = 16.23$.

data on a 40×40 uniform mesh; the spline coefficients were determined in 0.6 sec, and the time for 10^5 iterations was 0.14 sec. Computation times are for a single 2.66 GHz processor.

This example supports our belief that interpolative methods can produce maps that are both accurate and fast, even in cases where power series are not useful. Actually, the promise of interpolative map construction was evident long ago, in the work of Refs.[4, 5, 6, 7]. That work resulted in the generator of a fast symplectic map represented in polar coordinates in a hybrid Fourier-spline basis. It was applied successfully to an early LHC lattice, but in a slightly restricted region of phase space owing to a coordinate singularity arising from the polar coordinates. In work going back to 1984, G. Wüstefeld and collaborators have applied generating functions to maps for complicated magnetic fields, sometimes using interpolative methods [8].

In 1999 two of the authors proposed a method to find the map generator in Cartesian coordinates, with splines in all variables [9]. (Earlier, Berz had described a formally similar construction of the generator, but to be realized through power series rather than splines [10].) The

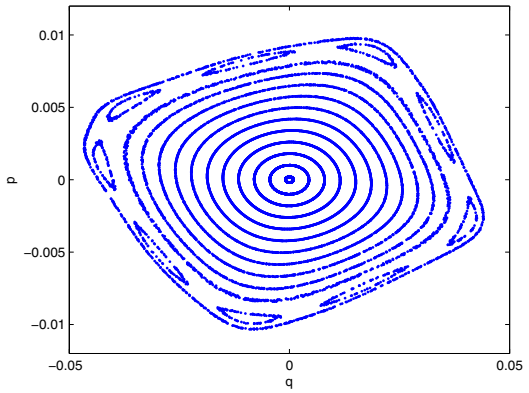

Figure 4: Phase plot from Spline Map, 10^5 turns, $\nu_x = 16.23$.

present paper gives the first numerical implementation of the scheme of Ref.[9]. Another idea, which came to light when one of the authors was preparing an article on the Hamilton-Jacobi equation [11], is to compute the generator directly as Hamilton's Principal Function, which is the integral of the Lagrangian regarded as a function of initial and final spatial coordinates.

2. RELATION OF THE GENERATOR TO THE TRACKING MAP

We denote the map defined through an element-by-element tracking code as follows:

$$q = Q(q_0, p_0) \,, \tag{1}$$
$$p = P(q_0, p_0) \,, \tag{2}$$

where $z = (q, p) = (z_1, \cdots, z_{2n})$ and $z_0 = (q_0, p_0)$ are final and initial phase space points for a system with n degrees of freedom. The map can refer to an arbitrary period in a circular or linear machine, but in the present application it is for one turn of a circular machine. In the theory of canonical transformations [11] the map is expressed implicitly in terms of a generating function or generator, which we take to be a "Type 1" generator of the form $F(q, q_0)$. The implicit map is defined by the equations

$$p = F_q(q, q_0) \,, \tag{3}$$
$$p_0 = -F_{q_0}(q, q_0) \,, \tag{4}$$

where subscripts denote partial derivatives: $F_q = (\partial F/\partial q_1, \cdots, \partial F/\partial q_n)$. We suppose that for q and q_0 in the region of interest, the $n \times n$ Hessian matrix F_{qq_0} is nonsingular:

$$\det F_{qq_0}(q, q_0) \neq 0 \,. \tag{5}$$

Then one can solve (4) for $q = q(q_0, p_0)$ (at least locally) and substitute the result in (3) to obtain also $p(q_0, p_0) = F_q(q(q_0, p_0), q_0)$ [12]. We wish to determine F so that the functions $q(q_0, p_0), p(q_0, p_0)$ can be identified with the map components $Q(q_0, p_0), P(q_0, p_0)$.

One can show that *any* $F(q, q_0)$ that has continuous second derivatives and satisfies (5) defines a symplectic transformation $(q(q_0, p_0), p(q_0, p_0))$. This fact is important for our map construction since it shows that symplecticity can be ensured even if F does not precisely reproduce the map (1),(2).

To derive the relation of F to the map (Q, P), let us suppose that in the region of interest

$$\det Q_{p_0}(q_0, p_0) \neq 0 \,, \tag{6}$$

in which case we can solve (1) for $p_0(q, q_0)$, provided that q is in the range of Q. Then, if F actually generates the map, Eqs. (3),(4), (2) show that

$$F_q(q, q_0) = P(q_0, p_0(q, q_0)) =: \gamma_1(q, q_0) \,, \tag{7}$$
$$F_{q_0}(q, q_0) = -p_0(q, q_0) =: \gamma_2(q, q_0). \tag{8}$$

Symplecticity of the map guarantees that the vector $\gamma = (\gamma 1, \gamma 2)$ is actually a gradient; i.e., that it has zero curl. The proof, which is not obvious, is given in Ref.[13]. Thus we have $\nabla F = \gamma$ from which we can obtain F itself as a path-independent line integral,

$$F(\zeta) = \int_{\zeta_0}^{\zeta} \gamma(\zeta') \cdot d\zeta' \,, \quad \zeta = (q, q_0) \,. \tag{9}$$

3. APPROXIMATION OF THE GENERATOR AS A SPLINE

Our first realization of the scheme of the previous section is based on interpolation of data by spline functions, with the help of the B-spline basis [14]. Let $s(x)$ be any spline function specified by a knot sequence $[t_i]_{i=1}^{n+k}$ where k is the order (degree $+1$) of the local polynomials that are joined to make up the spline. The B-splines $B_i(x)$ determined by that knot sequence form a basis, so that for some λ_i

$$s(x) = \sum_{i=1}^{n} \lambda_i B_i(x) \,. \tag{10}$$

The λ_i are fixed so that $s(x)$ interpolates data at distinct sites $x_j,\ j = 1, \cdots, n$. This is done efficiently thanks to the banded nature of the interpolation matrix. The banded structure arises because at any x only k of the B-splines are non-zero. At a given x all of the non-zero B_i are computed at once by de Boor's stable recursive method; their derivatives can be obtained similarly. This gives a fast evaluation of (10) or its derivatives in a time that increases only mildly with n; the increase is due only to a higher cost of searching for the knot interval in which x lies. Thus when the B-spline representation of a function is refined the evaluation time hardly changes, in marked contrast to a representation by Taylor series. For all B-spline operations we use standard Fortran software available at netlib.org [15].

For multidimensional interpolation the simplest approach is through a tensor product of B-splines. In 2D this is

$$s(x_1, x_2) = \sum_{i,j} \lambda_{ij} B_i^{(1)}(x_1) B_j^{(2)}(x_2) \,, \tag{11}$$

a simple iteration of the 1D interpolation. The superscripts indicate possibly different knot sequences for the two dimensions. The tensor product interpolation requires data on a Cartesian grid, which can be used in the present study only at fairly small phase space amplitudes. At large amplitudes we have oddly shaped interpolation domains, which we handle by completing the array of real data with reasonable but arbitrary values. An item on our agenda is to look at more local interpolation methods that not only can handle general domains but also are more efficient in high-dimensions, for instance radial basis functions [16] or Shepard interpolation [17]

Our construction of F proceeds in the following steps:

1. Make a spline $\tilde{Q}(q_0, p_0)$ of $Q(q_0, p_0)$ from values on a Cartesian mesh $\{q_{0i}, p_{0j}\}$.
2. Solve (1) for $p_0(q, q_0)$ on a similar mesh $\{q_i, q_{0j}\}$. This is done by Newton's method, taking a first guess for p_0 from the linear part of the map. The required Jacobian is approximated by $\tilde{Q}_{p_0}$.
3. Using this solution, evaluate γ_1 in (7) on the same mesh, and then make a spline of (γ_1, γ_2).
4. Integrate this spline on some convenient path to find $F(q_i, q_{0j})$ through (9).
5. Finally, make a spline to represent F from the values $F(q_i, q_{0j})$. This spline must be of order $k \geq 4$ (cubic or higher degree), to ensure that $F \in C^2$.

The map defined implicitly by F comes from (3),(4) and is evaluated as follows:

a. For any initial point (q_0, p_0) solve (4) for q by Newton's method, taking the spline value $\tilde{Q}(q_0, p_0)$ as a first guess.

b. Substitute the solution from (a) in (3) to obtain p as well.

In all the above steps the derivatives and integrals of splines are expressed analytically then evaluated numerically by means of BSPLVD and FPINTB [15]. Note that for a cubic spline the function $F_{q_0}(q, q_0)$ is C^2 in q, which makes it suitable for Newton's method in step (a).

4. APPLICATION TO AN ELECTRON RING LATTICE

We have carried out the map construction just described for horizontal motion in a lattice for an electron ring that was part of a study for an ILC damping ring. For a full description see [18]. It has racetrack form and entails 64 cells, primarily 90° FODO cells. The energy is 5 GeV, the length 960m, and the x-emittance is 47nm. A scaling to include more cells gives a lower emittance and a candidate for the damping ring. Tracking is done by the code LEGO, which integrates equations of motion based on the Hamiltonian in the local frame of each lattice component [19].

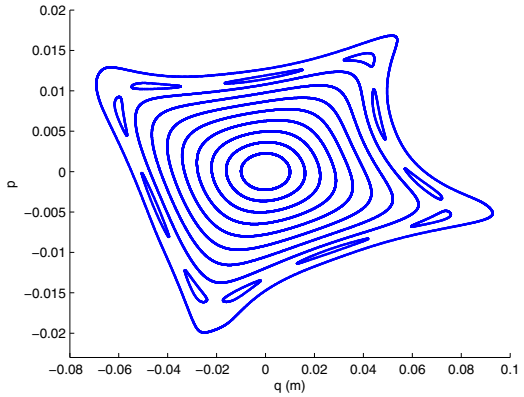

Figure 5: Phase plot from tracking, $\nu_x = 15.81$. Dynamic aperture (2000 turns) just beyond outer curve.

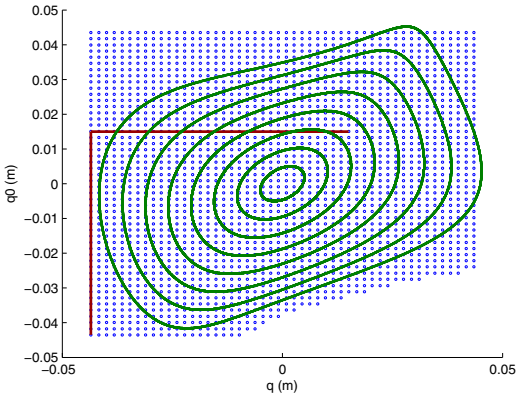

Figure 6: Points at which $p_0(q, q_0)$ exists, in blue. The curves represent values of (q, q_0) on the orbits of Fig.7. The path of integration for Eq.(9) is shown in red.

The motion of this example becomes very nonlinear as the dynamic aperture is approached. This makes it a challenging example for map construction. A hadron machine such as the LHC has weaker sextupoles and weaker nonlinearity out to the dynamic aperture, notwithstanding important effects of random higher order multipoles which contribute to long-term diffusion. On the basis of earlier work [5, 6, 7] we expect that the present algorithm will be useful for the LHC over a bigger range of amplitudes than it is in the present example. Nevertheless, the difficulties of this example have been very informative, and the lessons learned will certainly be relevant to other cases.

We now choose a tune, $\nu_x = 15.81$, for which the short term dynamic aperture is large, near the orbit with $(q_0, p_0) = (7\text{cm}, 0)$. There are no easily found resonances except for an 11th order one very close to the dynamic aperture. A phase plot is shown in Fig.5.

We try to construct a map on a rectangular region $|q_0| \leq r_1$, $|p_0| \leq r_2$, and expect that some sub-region will be mapped into itself by the constructed map. We therefore seek to construct $F(q, q_0)$ in a square domain, $|q|, |q_0| \leq r_1$. For $r_1 = 2.5$cm everything goes according to the plan of the previous section. This is in accord with the mathematical analysis of Ref.[9] which showed that the scheme should work at small amplitude. Passing to $r_1 = 4.5$cm, we encounter a new situation. The solution for $p_0(q, q_0)$ does not exist in one corner of the domain, as is seen in Fig.(6). Let us call this triangular region $\mathcal{B}$, for Bermuda Triangle. The Newton iteration, which converges beautifully elsewhere, diverges in $\mathcal{B}$ in a mode such that iterates get larger and larger. To generate the plot of successful solutions in Fig.6 the Newton iteration was declared a failure when iterates $p_0^{(n)}$ got bigger than a small multiple of r_2, or when it failed to converge to desired accuracy within a set maximum number of iterations. As expected, the boundary of $\mathcal{B}$ corresponds closely to the curve along which the condition of Eq.(6) first fails.

In order to make a tensor product spline of $\gamma(q, q_0)$ in spite of the missing values, we have to fill in the array

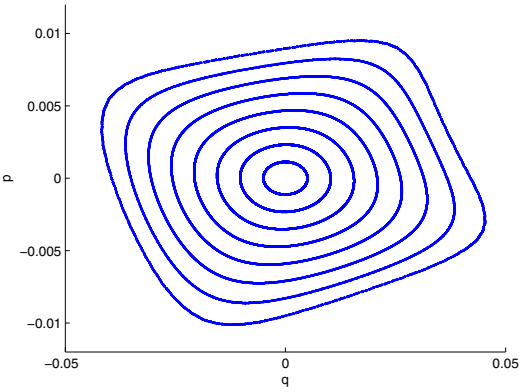

Figure 7: Symplectic map from generator, 10^7 turns. Iteration time 20-25 sec. for 10^7 turns, per orbit.

in some reasonable manner. We choose to put in values close to those on the boundary of $\mathcal{B}$, line-by-line in the q-direction. Of course, the path in the line integral must avoid the region $\mathcal{B}$. We choose the path shown in Fig.6.

This procedure with a 50×50 mesh leads to a symplectic map that gives the phase plot of Fig.7, where each orbit is followed for 10^7 turns in 20-25 seconds. We have also followed the outer orbit for 10^9 turns, finding no change to graphical accuracy from the result of 10^7 turns. Mapping time per turn is 20-200 times faster than the underlying tracking code (depending on the tracking integrator chosen), but this is not a fair comparison since our tracking code is not optimal for 2D tracking; it routinely computes the time of flight which is not needed in 2D. Nevertheless, the timing suggests a good outlook for speed in higher dimensions. The Newton iteration to solve (4) converged to machine precision in two or three steps.

The plot of Fig.7 is graphically indistinguishable from the corresponding plot from tracking over 10^5 turns. The quantitative agreement is not spectacular, however. At the

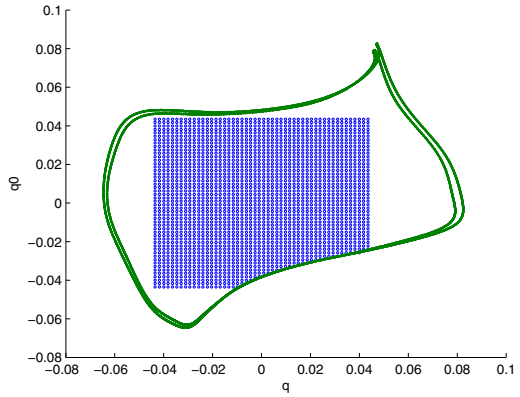

Figure 8: Two orbits in (q, q_0) plane close to the boundary of the region in which $F(q, q_0)$ exists. The dotted points are the same as those of Fig.6.

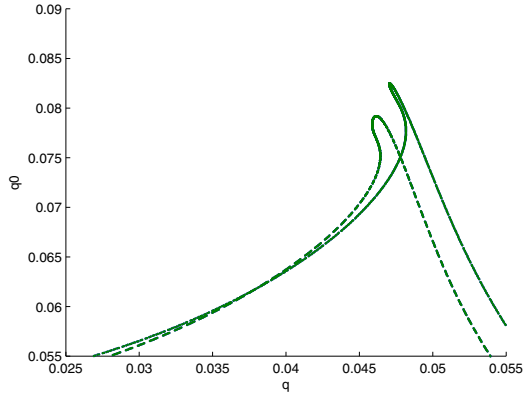

Figure 9: An enlargement of the top of Fig.8 showing intersecting orbits in (q, q_0) plane.

one turn level we have agreement to 5 digits at small amplitudes, declining to 4 digits at the outermost curve. As the map is iterated a lot of phase error quickly builds up, even though the iterates adhere to a clearly defined invariant curve. This is in accord with much earlier experience in approximating maps. For a quantitative test in spite of phase error one has to make an accurate representation of an invariant curve from tracking, then see how well the map follows it; this can be done with the method of [20].

When we try to go beyond the outermost curve of Fig.7, the method abruptly breaks down. The reason can be understood by using the tracking code to plot orbits in the $(q, q0)$ coordinates for increasingly large amplitudes. At some point two curves for neighboring initial conditions cross, as is seen in the upper right corner of Fig.8. There are two crossings, as seen in the enlargement in Fig.9. After this happens, the pair (q, q_0) does not determine a unique orbit, hence $p_0(q, q_0)$ is not a single-valued function and the construction of $F(q, q_0)$ must fail. The curve with smaller initial condition ($q_0 = 0.0604\mathrm{m},\ p_0 = 0$) roughly defines the boundary of existence of F. As seen in Fig.8, the divergence of the Newton method defines part of the boundary quite accurately.

We should be able with sufficient care to construct the generator in its full region of existence. The reason that our attempted construction failed abruptly with increasing amplitude is seen by comparing Fig.6 and Fig.8, both of which show the points at which $p_0(q, q_0)$ was determined by Newton's method. The orbits of higher amplitude that we seek go into the cusp-like region in the upper right corner of Fig.8, but the path of integration used in the code can reach such points only by penetrating the boundary of the allowed region, thus giving nonsensical F. This could be avoided by a better choice of path, but that would be difficult if not impossible to generalize to higher dimensions. Even with a better path, we have to face the problem of interpolation, which becomes increasingly awkward for the tensor product spline.

One might ask whether a generator $F_2(q, p_0)$ exists when $F(q, q0) = F_1(q, q_0)$ does not. Plotting the same two orbits of Fig.8 in (q, p_0) coordinates, we find again two intersections but at negative $q \approx -.0605, -.025$, rather than the positive $q \approx .0375, .0475$ of the intersections in (q, q_0) coordinates. Thus when (q, q_0) fails to specify an orbit (q, p_0) succeeds, and vice versa. Each of the generators fails to exist at some points on each of the two orbits. The symplectic condition in 2D phase space is $Q_{q_0} P_{p_0} - Q_{p_0} P_{q_0} = 1$ which shows that Q_{p_0} and Q_{q_0} cannot vanish simultaneously, hence either F_1 or F_2 must exist *locally*. A theorem in [21] generalizes this statement to higher dimensions.

5. GENERATOR CONSTRUCTION BY ACTION INTEGRAL

We have shown one way to construct the generator, through a partial map inversion to give $p_0(q, q_0)$ and a line integral. Another method goes back to Hamilton's original work of 1830-1832 [11]. He obtained the generator through the integral of the Lagrangian. Namely,

$$S(q_0, p_0, t) = \int_0^t \left[p(\tau, z_0) \cdot \dot{q}(\tau, z_0) - H(z(\tau, z_0), \tau)\right] d\tau \ , \tag{12}$$

where H is the Hamiltonian and the second argument of the orbit $z(\tau, z_0)$ indicates its initial condition, $z_0 = (q_0, p_0)$. The time-like variable t would normally be path length s in accelerator physics. Hamilton's essential idea was to regard the action as a function of (q, q_0) rather than (q_0, p_0). Then the generator, which is a solution of the Hamilton-Jacobi equation called Hamilton's Principal Function, is

$$F(q, q_0, t) = S(q_0, p_0(q, q_0, t), t) \ . \tag{13}$$

This is identical to the function of the previous sections when $t = T$ is the period of the map.

The advantage of this formula for a practical construction is that it avoids the line integral (9), which could be almost impossible to deal with in higher dimensions with excluded regions. A disadvantage is that the tracking code must be augmented to calculate the action integral in the course of tracking. Fortunately, LEGO is able to furnish the Hamiltonian needed in the integrand.

6. LOCAL INTERPOLATION AND THE USE OF SCATTERED DATA

Tensor product splines may be adequate in many problems but in general we need a more local sort of interpolation or approximation that can be used in non-rectangular domains. One possibility is a generalized Shepard method [17] based on the formula

$$F(\zeta) = \sum_i P_i(\zeta) \frac{w_i(\zeta)}{\sum_j w_j(\zeta)} ,$$
$$w_i(\zeta) = c(\zeta - \zeta_i) \|\zeta - \zeta_i\|^{-n} , \qquad (14)$$

where n is a positive integer such as 4 or 6. Here $P_i(\zeta)$ is a polynomial that interpolates or approximates values of F at ζ_i and a few nearby sites. The factor $c(\zeta - \zeta_i)$ is a smooth cutoff that restricts the sum at any evaluation. For ζ close to ζ_i this behaves like $P_i(\zeta)$ with small corrections from other terms in the sum. The formula is globally smooth, with the degree of smoothness controlled by that of c.

This formula works when the sites ζ_i lie on a mesh or even when they are scattered. In the latter case one would normally use a least squares fit to determine the coefficients of P_i, rather than strict interpolation, since the interpolation matrix could be nearly singular for some dispositions of sites. Interpolation could be done by replacing the polynomial by a radial basis function [16].

As suggested by Fasshauer (private communication) it may be advantageous to use scattered sites from a quasi-random (low discrepancy) sequence. This might enjoy the advantage that quasi - Monte Carlo quadrature has over mesh based quadrature in high dimensions [22]. Such a scheme was tried in Ref.[23], in connection with solution of a Vlasov equation. In 2D it was possible to reduce the number of data sites by a factor of 8 in comparison to mesh based interpolation. A much bigger advantage is expected in higher dimensions, and this augurs well for construction of maps in 4D or 6D phase space using quasi-random data.

7. CONCLUSION

After analysis of a difficult example in 2D phase space, we have a plan for building fast symplectic maps in higher dimensions. The method described in Sec.3 will probably succeed in many cases, but in general we shall need a more powerful approach using local interpolation and Hamilton's Principal Function. The use of scattered interpolation sites from quasi-random sequences may increase the efficiency of interpolation in high dimensions.

ACKNOWLEDGMENT

We thank J. Scott Berg and Ronald Ruth for helpful remarks and encouragement.

REFERENCES

[1] M. Berz, Particle Accelerators **24**, 109 (1989).

[2] D. Douglas, É. Forest, and R. V. Servranckx, IEEE Trans. Nuc. Sci., **NS-32**, 2279 (1985); Y. Yan, P. Channell, and M. Syphers, SSCL-Preprint-157 (1992).

[3] D. T. Abell, E. McIntosh, and F. Schmidt, Phys. Rev. ST Accel. Beams **6**, 064001 (2003) and references therein.

[4] J. S. Berg and R. L. Warnock, Proc. IEEE 1991 Part. Accel. Conf., San Francisco, p.1654.

[5] J. S. Berg, R. L. Warnock, R. D. Ruth, and É. Forest, Phys. Rev. E **49**, 722-739 (1994).

[6] R. L. Warnock and J. S. Berg, Particle Accelerators **54**, 213-222 (1996).

[7] R. L. Warnock and J. S. Berg, AIP Conf. Proc. **395**, 423-445 (1997).

[8] G. Wüstefeld, "A Generating Function for Particle Tracking", note obtained from wuestefeld@bessy.de; M. Scheer, Ph.D. Thesis, Humboldt-Universität, Berlin (2008).

[9] R. L. Warnock and J. A. Ellison, Applied Numerical Math. **29**, 89-98 (1999).

[10] M. Berz, in *Nonlinear Problems in Future Accelerators*, (World Scientific, Singapore, 1991), pp. 288-296.

[11] R. L. Warnock, "The Hamilton-Jacobi Equation", under Dynamical Systems at scholarpedia.com.

[12] Although a local solution does not imply a single-valued global solution, we can proceed assuming (5) and try to verify a global solution.

[13] R. L. Warnock and J. A. Ellison, AIP Conf. Proc. **405**, 41-45 (1997).

[14] C. de Boor, "A Practical Guide to Splines", Revised Edition, (Springer, New York, 2001)

[15] See libraries pppack and dierckx at netlib.org. In pppack find SPLINT for interpolation, BSPLVD with INTERV for evaluation and differentiation. In dierckx find FPINTB for integration .

[16] G. Fasshauer, "Meshless Approximation Methods with Matlab", (World Scientific, Singapore, 2007).

[17] R. Farwig, Math. of Computation **46**, 577-590 (1986).

[18] Y. Cai, SLAC-PUB-11084 (2005).

[19] Y. Cai, SLAC-PUB-8011 (1998).

[20] R. L. Warnock and R. D. Ruth, Physica D **56**, 188-215 (1992).

[21] V. I. Arnol'd, "Mathematical Methods of Classical Mechanics", (Springer, New York, 1978), Section 48-B.

[22] H. Niederreiter, "Random Number Generation and Quasi - Monte Carlo Methods", (SIAM, Philadelphia,1992).

[23] R. L. Warnock, J. A. Ellison, K. Heinemann, and G. Q. Zhang, Proc. EPAC'08, Genova, paper TUPP109.