

***Numerical Methods for the Simulation of High
Intensity Hadron Synchrotrons***

Alfredo Luccio, Nicholas D'Imperio, Nikolay Malitsky

*Presented at the Coulomb '05 High Intensity Beam Dynamics
Senigallia, ITALY
September 12-16, 2005*

September 2005

Collider – Accelerator Department

Brookhaven National Laboratory

P.O. Box 5000
Upton, NY 11973-5000
www.bnl.gov

Notice: This manuscript has been authored by employees of Brookhaven Science Associates, LLC under Contract No. DE-AC02-98CH10886 with the U.S. Department of Energy. The publisher by accepting the manuscript for publication acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes.

This preprint is intended for publication in a journal or proceedings. Since changes may be made before publication, it may not be cited or reproduced without the author's permission.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or any third party's use or the results of such use of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof or its contractors or subcontractors. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.



Numerical Methods for the Simulation of High Intensity Hadron Synchrotrons ¹

Alfredo U. Luccio, Nicholas D'Imperio, Nikolay Malitsky

Brookhaven National Laboratory, C-AD Department, Upton, NY 11973, USA

Abstract. Numerical algorithms for PIC simulation of beam dynamics in a high intensity synchrotron on a parallel computer are presented. We introduce numerical solvers of the Laplace-Poisson equation in the presence of walls, and algorithms to compute tunes and twiss functions in the presence of space charge forces. The working code for the simulation here presented is *SIMBAD*, that can be run as standalone or as part of the UAL (Unified Accelerator Libraries) package.

PIC TRACKING BASICS IN THE SYNCHROTRON CODE *SIMBAD*

In the discussion of numerical methods we will continuously refer to the PIC synchrotron simulation code *SIMBAD*[1]. Accelerator simulation relies on approximations. PIC itself is an approximation. Other approximations used in the code will be indicated when they occur.

In a set of approximations the code uses a Split Operator technique[2]. Progressively, the coordinates of randomly prepared macro particles are transformed through maps calculated for a bare lattice by some optical code, followed by Space Charge (SC) kicks. Maps must be continuously updated during acceleration, because

- there may be nonlinear effects in the lattice, like hysteresis in the magnets,
- map elements describing the longitudinal motion are a function of particle energy,
- we may want to vary the lattice dynamically, e.g. by changing quadrupole setting and then betatron tunes.

The independent variable may be time, t , or the longitudinal coordinate s . *SIMBAD* uses s during macro transport, but transforms between s and t when SC forces are applied to the macros, that are then all considered at a time that is the same for everyone. Like in *MAD*[3], the canonical phase space vector is

$$\vec{u} = (x, p_x/p_z, p_y/p_z, -c\Delta\phi, \Delta E/pc),$$

x radial, y vertical, and z longitudinal. Δ denotes the deviation of a variable from the corresponding of the

synchronous particle, and δ the sudden variation of a quantity at some location, say, in a thin kick or RF cavity.

To address the SC problem in the presence of walls, we want to solve the electromagnetic problem for a steady beam current flow, using the two partial differential equations (Poisson and Ampère Law)

$$\nabla^2\Phi(P) = -\frac{\rho(Q)}{\epsilon_0}, \quad \nabla^2\vec{A}(P) = -\frac{\vec{j}(Q)}{\mu_0}. \quad (1)$$

Q , source point, P , field point. Beam charge density $\rho(Q)$ is obtained by binning the position of macroparticles on a grid, and current density $\vec{j}(Q)$ by binning momenta. If beam bunches are long, as it is the case with synchrotrons, we may make the approximation that the beam current is locally parallel to the walls. In this case we only integrate the Poisson equation, and represent the partial compensation between space charge repulsion and current attraction with a factor γ^{-2} .

Transverse and longitudinal SC momentum kicks are (\wp , perveance)

$$\frac{\delta p_{\perp}}{p} = \wp \frac{\partial \Phi}{\partial r} L_T, \quad \frac{\delta \Delta E}{E} = \beta^2 \wp \frac{\partial \Phi}{\partial z} L_S, \quad \wp = \frac{4\pi\lambda q h r_0}{H\beta^2 \gamma^3 m_0},$$

with L_T and L_S , length of kicks, λ , longitudinal charge density, h , harmonic number, and H the grid mesh size.

In a synchrotron, beam bunches are long and thin. We treat the longitudinal dimension differently than the transverse, and adopt the approximation to numerically solve the first of Eqs.(1) in only 2 dimensions in different longitudinal segments along the beam. On a transverse grid and in the approximation of a boundary of perfectly conductive walls ², Poisson's assumes the following dis-

¹ Work performed under the auspices of the U.S. Department of Energy

² Finite conductivity will not be discussed here for lack of space.

crete form

$$-4\pi\rho_{ij} = \mathcal{L}_{ij}^{kl}\Phi_{kl}; \quad \Phi_{i,j}(w) = 0, \quad \rho_{i,j}(w) = \rho_{image},$$

with the Laplacian \mathcal{L} a band sparse matrix. The discrete implicit formulation for the potential is

$$\rho_{i,j} = \frac{1}{2H^2}(\Phi_{i-1,j} + \Phi_{i,j+1} - 4\Phi_{i,j} + \Phi_{i+1,j} + \Phi_{i,j-1}).$$

This is a set of linear algebraic equations. With a boundary made of N points, around an area of M points, the number of equations is $M + N$. Known quantities are: charge in the inner M point plus N values of the potential on the boundary. Unknowns are: potential in the inner M points plus image charge on the N boundary points. We solve the system by LU decomposition, and iteration, as described in a later Section.

Note that SC solvers based on the **integral** form of Eqs. (1), e.g. FFT solvers, would require a previous knowledge of the wall image. In a **differential** form the image is part of the solution.

ALGORITHMS BASED ON THE ONE TURN MATRIX

SC induces coupling between betatron and synchrotron modes, producing a tune spread and modifying all lattice functions seen by the individual macros, so the matching of the beam to the lattice is altered. These effects can be studied through the One Turn Matrix, calculated for each macro. This matrix will differ from the M^{OT} for the bare lattice, that has the general form (no natural coupling)

$$M_{bare}^{OT} = \begin{pmatrix} c_x & s_x & 0 & 0 & 0 & D \\ c'_x & s'_x & 0 & 0 & 0 & D' \\ 0 & 0 & c_y & s_y & 0 & 0 \\ 0 & 0 & c'_y & s'_y & 0 & 0 \\ E & E' & 0 & 0 & 1 & G \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}. \quad (2)$$

Matrix elements D and D' represent the dispersion of the lattice, and E and E' the synchro-betatron coupling terms. It can be shown that the symplecticity of the M^{OT} is independent from the term G and is satisfied if

$$\begin{pmatrix} E \\ E' \end{pmatrix} = \begin{pmatrix} c' & -c \\ s' & -s \end{pmatrix} \begin{pmatrix} D \\ D' \end{pmatrix}. \quad (3)$$

In a general tracking, in the presence of SC forces, the 6×6 matrix elements can be dynamically calculated by solving 36 linear equations among particle 6 coordinates in 6 successive turns

$$u_i^{(n+1)} = \sum_{j=1}^6 M_{i,j}^{OT} u_j^{(n)}, \quad i = 1, 6, \quad n = N, N+5.$$

There will be an M^{OT} for each macro in the simulation. In the new calculated matrix, the elements appear somewhat different than in Eq. (2), in particular the 0's are in general not 0's anymore, showing the coupling induced by SC.

In the **transverse** dimension, fractional betatron tunes for each macro can be calculated from the eigenvalues λ of M^{OT} [4]³

$$\tilde{\nu} = (1/2\pi) \arctan[Im(\lambda)/Re(\lambda)]. \quad (4)$$

Also the twiss functions α_T and β_T , for x (indices 3 and 4 for y), from the M^{OT} elements, are calculated as

$$\alpha_T = \frac{M^{OT}(1,1) - M^{OT}(2,2)}{2 Im(\lambda)}, \quad \beta_T = \frac{M^{OT}(1,2)}{2 Im(\lambda)}. \quad (5)$$

As an example, apply the above to the transverse dynamics of the AGS for three cases, with

- no lattice coupling elements, no acceleration,
 - $E_k = 1.2$ GeV,
 - unnormalized emittance $\epsilon_x, \epsilon_y = 40, 10$ mm-mrad,
 - beam profile = Gauss (transverse), uniform (long.),
 - no. of PIC macros = 10^5 ,
 - high RF harmonic: $h = 24$, voltage: $V = 80$ kV.
- (1) Low intensity, beam charge 10^9 A.s,
 - (2) High intensity, $4 \cdot 10^{14}$ A.s, bunch length 360 deg,
 - (3) Short bunches: 10 deg, high space charge: $4 \cdot 10^{12}$.

Results are in Table 1. In the table, $\Delta\nu$ is the maximum "Laslett" tune depression in a round beam with Gaussian charge distribution

$$\Delta\nu = -\frac{r_0}{(2\pi)^{3/2}} \frac{N}{\epsilon\beta^2\gamma^3 B_f}, \quad (6)$$

with r_0 , classical particle radius, N , number of particles, ϵ , r.m.s. emittance, $B_f = I/I_{peak}$ the bunching factor.

The betatron tune footprint is shown in Fig. 1, where the Laslett tune is calculated for various effective emittances defined as follows

$$\epsilon_x = (1-C)\epsilon_x^{(0)} + C <\epsilon>, \quad \epsilon_y = (1-C)\epsilon_y^{(0)} + C <\epsilon> \\ <\epsilon> = (\epsilon_x^{(0)} + \epsilon_y^{(0)})/2,$$

with a coefficient C that represents the SC induced coupling between transverse modes.

The **longitudinal** motion is described by

$$\begin{cases} \phi_{n+1} = R(5,1)x + R(5,2)x' + \phi_n + R(5,6) \frac{\Delta E}{pc} \Big|_n \\ \frac{\Delta E}{pc} \Big|_{n+1} = \frac{V}{pc} \beta \frac{2\pi h}{L_T} (\sin \phi_n - \sin \phi_s) + \frac{\Delta E}{pc} \Big|_n \end{cases}$$

The first equation gives the phase advance at each machine element, with $R(5,1)$, $R(5,2)$ representing the

³ The method gives ambiguous results when the two betatron tunes are close together (e.g. Montague Resonance).

TABLE 1. Examples for the sample macro shown in Fig. 1 below

	\tilde{v}_x	α_x	β_x	Δv_x	\tilde{v}_y	α_y	β_y	Δv_y
(MAD) bare	[8].6920	-1.309	15.451		[8].7337	1.303	15.375	
(1) 10^9	0.6920	-1.309	15.451	0	0.7337	1.303	15.376	0
(2) $4 \cdot 10^{14}$	0.6572	0.396	22.491	-0.0348	0.6830	-0.235	10.514	-0.0507
(3) $4 \cdot 10^{12}$	0.8325	0.313	21.476	-0.0395	0.5261	-0.217	10.795	-0.0628

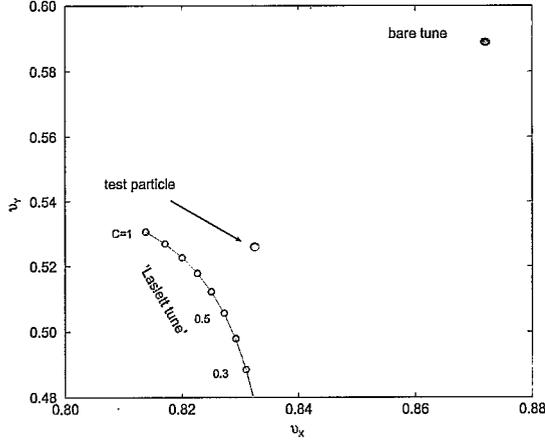


FIGURE 1. Tune footprint for a high intensity AGS beam. Laslett tune is calculated for different effective emittances.

synchro-betatron coupling. The second, essentially non linear, applies at an acceleration station between turns n and $n + 1$.

For the entire machine, with γ_s the synchronous energy and γ_T the transition energy, write

$$M^{OT}(5,6) = \frac{L_T}{\beta_s^2} \eta = \frac{L_T}{\beta_s^2 \gamma_s^2} - \frac{L_T}{\beta_s^2 \gamma_T^2}, \quad (7)$$

with L_T = machine length. The sign of $M^{OT}(5,6)$ is positive below transition and negative above⁴. From Eq.(7), that we assume holds also in the presence of SC, the value of γ_T can be calculated as

$$\gamma_T = \gamma_s \left(1 - \frac{M^{OT}(5,6)}{L_T} \beta_s^2 \gamma_s^2 \right)^{-1/2}. \quad (8)$$

By matrix multiplication in the full machine we obtain the following relation between $M^{OT}(5,6)$ and the dispersion elements in the arcs⁵

$$M^{OT}(5,6) = \sum R(5,6) - \sum [R(5,1)_i R(1,6)_{i-1} + R(5,2)_i R(2,6)_{i-1}].$$

⁴ $(5,6)$ in a single element does not in general change sign, while it is the $(5,6)$ element of the OTM that changes its sign at transition, due to the cumulative effects of the dispersion terms along the ring

⁵ If the arcs are all identical, the second term in the r.h.s. vanishes.

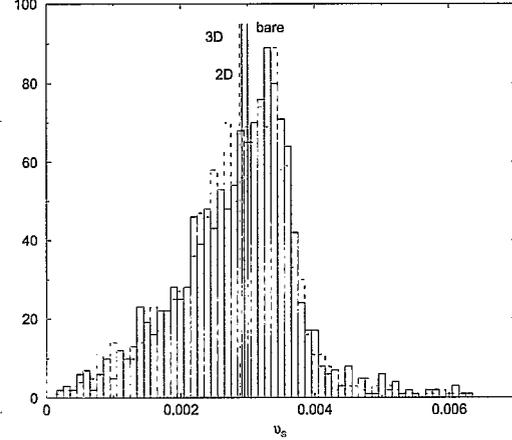


FIGURE 2. Synchrotron tune for a FODO, bare and with SC. The differences of 3D vs. 2D calculation of the space charge, as it will be described in the next section, are small

The synchrotron oscillation frequency can be calculated from the eigenvalues of the M^{OT} with Eq.(4), like for the transverse betatron. For small oscillations and no SC the tune from the eigenvalues matches the classical result

$$v_s = \sqrt{(eV/E)(h|\eta| \cos \phi_s / 2\pi \beta^2)}. \quad (9)$$

For large SC, the synchrotron tune and the value of γ_T show a finite spread. For example, in a test FODO with no acceleration, and

$E_k = 1.4$ GeV, $V_{RF} = 800$ KV, $h = 1$, $L_T = 384$ m, the tune, calculated from the eigenvalues, and γ_T , from the M^{OT} , compared with the bare lattice values, are (see also Fig. 2)

Space charge	v_s	γ_T
0.	0.00302	7.00745
10^{13} (2D)	0.00292 ± 0.00117	8.607 ± 1.291
10^{13} (3D)	0.00289 ± 0.00118	8.882 ± 1.332

Similarly to the transverse dimension, where SC generates some mismatch (effective Twiss function different from the bare lattice values), also in the longitudinal direction SC creates a **mismatch** that in particular affects the behaviour at transition and the symplecticity condition of beam transport. Although the symplecticity of an accelerator matrix is independent of the value of the el-

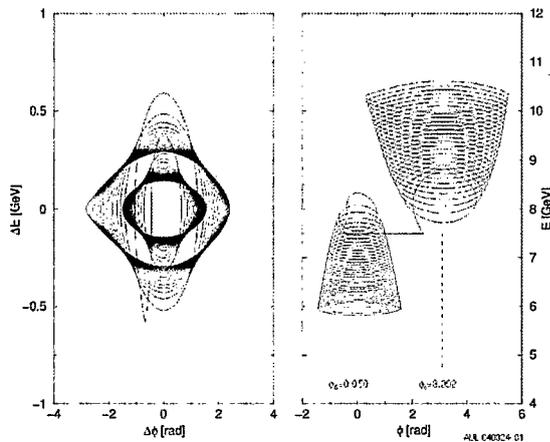


FIGURE 3. Longitudinal dynamics across transition in the AGS. Short bunches suitable for the proposed BNL Neutrino Factory

ement (5,6), other conditions for symplectic transport, like the one expressed by Eq.(3), are affected by SC.

The evolution of the phase space patterns in the presence of SC induced mismatch is rather complicated. A reference plot for the longitudinal motion across transition in the AGS, for a macro particle with γ_T in the 'center of mass' of the distribution is shown in Fig. 3. In particular, note in the figure how the one particle in the longitudinal phase space rotates in a counter clockwise direction below transition and clockwise above.

PARALLEL COMPUTATION

A meaningful 3-D PIC simulation requires between 10^5 and 10^7 macroparticles. It is presently impractical to simulate by a single computer process the transport of so many macros in a synchrotron over thousand of turns. Parallel computation is a must. *SIMBAD* is being implemented using the Message Passing Interface, MPI[5]. The code models non-linear effects, space charge in particular, and utilizes a genetic load balancing algorithm for performance.

The code simulates the accelerator by representing each element of the machine, or operation on the beam, as a **node** in the ring. For simple tracking with no space charge, machine elements take the form of 6×6 matrices to first order or larger maps to second order. Parallelization is of the embarrassing variety where the herd of macroparticles is distributed evenly among the processes. When space charge calculations are performed the parallelization is more complex.

In **two dimensions** the space charge calculations take place at given nodes, situated around the circumference of the ring. Conveniently, but not necessarily, we place

an SC node at the location of each physical element in the ring. This is generally equivalent to perform tens of SC kicks per betatron wavelength.

The herd of macro particles arrives at each node independent of time and is represented as a set of superimposed flat disks of charged particles with different particle densities, consistent with the $\Delta\phi$ coordinate of the particles. Particles are then binned onto a 2-D mesh using bilinear interpolation and the resulting charge density is inserted into the Poisson equation to yield the scalar potential. A coefficient is used for each disk, proportional to the particle density in the disk. The Poisson equation is solved by means of a sparse LU solver and is not done in parallel since the 2-D meshes involved are not excessively large. The directional derivative of the potential at each mesh point is the component of the SC force, which is applied to a macro as a kick proportional to the length of the space charge element.

In this 2-D case the only parallelization beyond the embarrassing case is the global reduction applied to obtain the charge density. At each space charge node the processors bin their macro particles onto a local mesh and then sum the charge values at the local mesh points onto a global mesh. The global mesh is then used to calculate the potential and subsequent forces. This scheme scales **linearly** with the number of processors.

To better represent the variation of beam density both transversely and longitudinally, and its time evolution, another model is used, characterized as **3-D space charge**. Here, the parallelization is no longer trivial, but should possess the capability to effectively and efficiently handle almost any beam configuration.

In 3-D the simulation can no longer be independent of time. Instead, at each SC node the beam, represented as a flat disk to this point, is expanded longitudinally by transforming all six phase space coordinates of each macro particle so that time becomes the independent variable. By doing so, the longitudinal density of the beam is modelled as well as the transverse density that varies with longitudinal position s . The beam is then divided into longitudinal **segments**, all brought at the same time. This is particularly important and physically correct, because not only the true structure of the beam is reconstructed, but the interaction between adjacent beam slices happens with the slices at the same time. Fig. 4 shows a beam bunch in a test FODO synchrotron, expanded and "frozen" in time. The bare beam envelope, proportional to $\sqrt{\beta_{T,bare}}$ is also shown.

In summary, in 3-D, the propagation of the beam through the lattice at a SC node consists of the following sequence of operations:

- (1) transfer of coordinates through maps,
- (2) expansion of the beam longitudinally, by space to time transformation,
- (3) application of space charge kicks,

(4) compression of the beam again into a disk by t to s transformation,

.....

(1) transfer of coordinates to the successive node.

At a non-SC node, only step (1) is performed. In 2-D, only steps (1) and (3).

Each process takes a number of segments and performs all calculations locally on 2-D grids placed in the center of each segment. Longitudinal SC forces are calculated at each x, y point from a fitting of the potential difference between contiguous segments

3-D parallelization is translated to a problem of optimal **load balance**. A naïve approach where each process is assigned the same number of space charge segments may be typified in the following example. For a ring with K space charge segments, N total macros, and P processes, N/P macros would initially be injected into the ring by each process. Each macro particle has no initial constraint regarding its longitudinal coordinate upon injection and therefore may be found in any of the K segments. When the first space charge segment is encountered, the processes synchronize and transform their phase space coordinates to a time dependent frame. The ring is spatially decomposed along its length and so each process is assigned K/P space charge segments with one of the processes taking the remainder. The macro particles are then exchanged based on the longitudinal boundary positions between segments. After the exchange concludes, each process contains all the macro particles in the global herd that belong to its K/P segments. The communication involved is large only for the first space charge segment, as the synchrotron motion of the particles is relatively slow and cause infrequent migration of particles between processes.

The processes then perform the 2-D transverse space charge calculation for each of their K/P segments before collapsing the beam back to a space dependent frame and continuing tracking. The procedure repeats itself at the subsequent space charge node, though with less particle exchange.

This algorithm works well for a beam with a uniform longitudinal distribution. If this is not the case and the beam is not longitudinally uniform, as will occur if the beam is being accelerated or confined, the simulation will not be efficiently load balanced and the situation may arise where one process has many more macro particles than another. Several additional factors must be considered when decomposing the problem over the process domain.

The computational burdens are dependent on two variables: the number of space charge segments over which the Poisson equation must be solved and the number of macro particles in the local herd. Then, rather than simply dividing the number of segments evenly among the processes, it is more efficient to consider the number of

segments assigned to a given process as a function of the number of macro particles contained within them. Therefore, *SIMBAD* dynamically calculates an optimal decomposition scheme to balance the load, using a **genetic algorithm** that optimizes the number of SC segments to be assigned to each process.

The algorithm utilizes two parents (M,F) and two children (D,S), each of which represents a different distribution of elements among the processes. After the parents are initially created, they mate to produce two offspring. All four are then tested for optimal load balance at which point the process repeats. The entire space is efficiently searched and an efficient allocation generated. The algorithm continues by iterating through **mating, natural selection, and mutation**.

The mating phase combines the parents using an alternating element scheme to create two offspring. The four resulting instances are then compared to select two which will mate in the next iteration (natural selection). The metric which establishes the optimum load balance is a sum of two functions f and g , where f is the function defining the optimum distribution of macros and g the function defining the optimum distribution of elements. A weighting factor, w , is included that determines the precedence and importance of f relative to g . The weight is significant and variable and depends on the total number of macros used in the simulation and the size of the Poisson grid. The former, since overloading one process may overwhelm the available memory, and the latter, because solving the Poisson equation may easily consume the majority of clock cycles, being largely independent of the number of macro particles.

The functions are

$$f = \sum_{i=1}^P \left(1 - P \cdot \frac{E_i}{E_T} \right), \quad g = \sum_{i=1}^P \left(1 - P \cdot \frac{N_i}{N_T} \right)$$

where P is the number of processes, E_i is the number of segments per process, E_T is the total number of segments to be distributed, N is the number of macros per process, and N_T is the global number of macro particles. As stated previously, w is a function of N_T and E_T .

The comparison function is $h = f + w \cdot g$. The successive mating pair is chosen from the available four by choosing the two with the lowest value of h . Before mating the chosen pair, a small mutation is introduced into one of them and the process repeats. The number of iterations required to evolve the optimum solution is

$$P \cdot E_T \cdot \ln(P \cdot E_T).$$

An example of genetic algorithm flow is given in Fig. 5.

UNIFIED ACCELERATOR LIBRARIES

SIMBAD is a part of and can be run through the Unified Accelerator Libraries, UAL[6], an environment designed

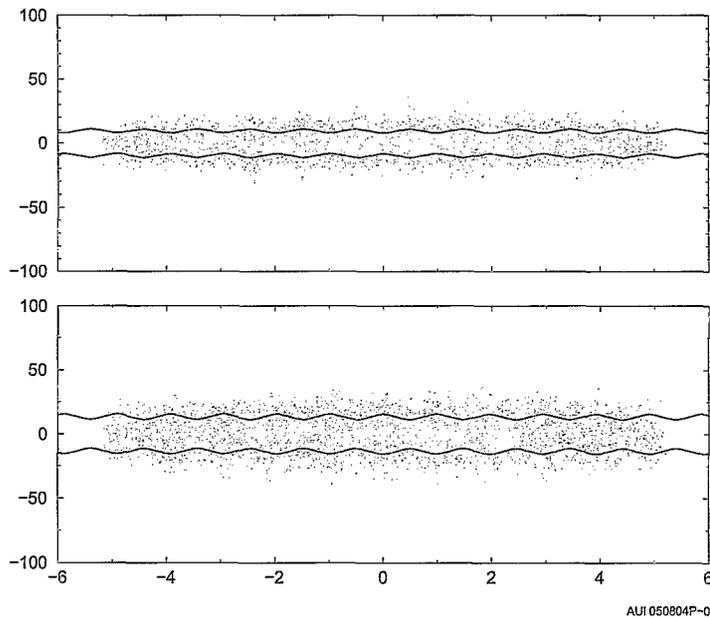


FIGURE 4. Beam profile: Upper figure: $(z - y[m - mm])$, Lower figure: $(z - x[m - mm])$. FODO - 3-D. $N = 10^{13}$, $h = 4$, $V = 800KV$, $\Delta\phi = 30\text{ deg}$. Distribution: Gaussian (x, y) , parabolic-Gaussian $(\Delta\phi, \Delta E / pc)$.

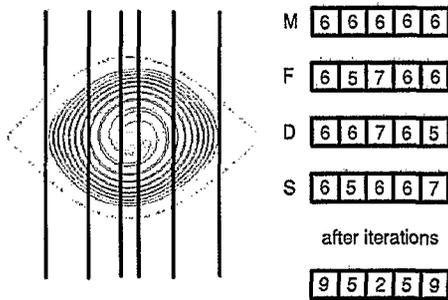


FIGURE 5. Example of one generation of selection. Arrays on the right represent slices per process during the various stages. The phase space graph on the left corresponds to an optimal solution, shown by the bottom array and the lines that indicate the process boundaries.

to address the complex simulation tasks of beam dynamics studies. UAL offers an open collection of accelerator algorithms and a consistent mechanism for building configurable project-specific accelerator off-line models.

At this time, UAL joins several accelerator programs, as shown in Fig. 6: PAC (Platform for Accelerator Codes), ZLIB (Numerical Library for Differential Algebra), TEAPOT (Thin Element Program for Orbits and Tracking), ACCSIM (Accumulator Simulation), SIMBAD (Simulation of Beams, Advanced Dynamics). Modules that are under active development are ICE (In-

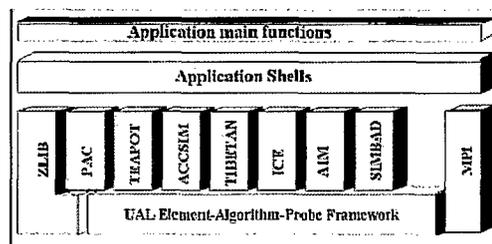


FIGURE 6. UAL Libraries

coherent and Coherent Effects), AIM (Accelerator Instrumentation Module), SPINK (spin tracking in accelerators), and TIBETAN (longitudinal phase space tracking). The Application Programming Interface (API), in Perl, provides a shell for integrating and managing all project extensions.

A cornerstone of UAL is the **Element-Algorithm-Probe** framework which identifies the association among three concepts: accelerator element, tracking algorithm, and evolved object (such as Bunch, Taylor map, etc.). Having initially rejected any implicit linkage between algorithms and elements, the framework connects them in application according to the user-specific **Accelerator Propagator Description Format (APDF)** file, similar to a propagator extension to the MAD lattice description.

A typical simulation model with SIMBAD trackers and AIM monitors is defined as indicated in Ta-

TABLE 2. UAL model

```

apdf>
<propagator id="simbad" accelerator="ring">
  <create>
    <link algorithm="SIMBAD::TSCPropagatorFFT" types="Default"/>
    <link algorithm="AIM::Monitor" types="Monitor"/>
    <link algorithm="AIM::PoincareMonitor" types="Instrument"/>
  </create>
</propagator>
</apdf>

```

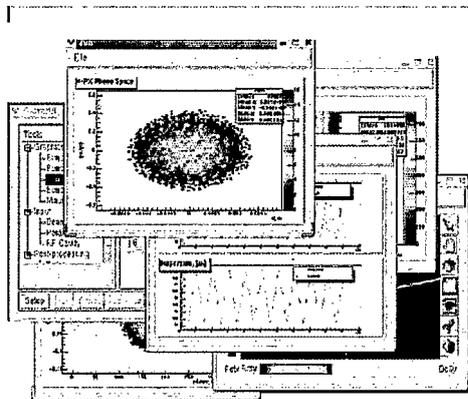


FIGURE 7. the Accelerator Physics Player

ble 2 In this example, a SIMBAD tracker is associated with all element types but monitors. Internally, the SIMBAD::TSCPropagatorFFT class is implemented as a composite model combining a space charge kick and a conventional tracker selected from a catalog of the UAL algorithms, such as TEAPOT thin-lens integrator, ZLIB Taylor map. The same approach has been applied on parallel computers by mixing together sequential and parallel tracking components.

Recently, UAL has been also integrated with the QT GUI development framework and the ROOT analysis environment [7]. The new package extends the UAL simulation algorithms with an open collection of analysis and visualization components. The original Perl-based describing interface has also been transformed into a GUI application. Its main window is implemented as a configurable and interactive **Accelerator Physics Player**, which coordinates data flows among tracking engines, interactive graphics, and data-processing (see Fig. 7).

Development of UAL is strongly prejudiced toward importing existing codes rather than developing new ones. Importation of codes into UAL is an ongoing enterprise and when a code is said to have been imported it does not necessarily mean that all features are supported.

In practice, to run any code, or some algorithm of a code, through UAL, the user needs to write a suitable **driver** that contains the relevant values of the parameters. UAL provides the environment and the graphic interface. One could, e.g. produce transfer maps with TEAPOT, track with PAC and calculate space charge effects with SIMBAD. Diverse codes are to form “modules” in an evolving, unified, coherent environment. To facilitate such unification, UAL has introduced an open architecture in which diverse accelerator codes are connected together via common accelerator objects such as *Element*, *Bunch*, *Twiss*, etc. In this architecture each accelerator code is implemented as an object-oriented library of C++ classes

ACKNOWLEDGMENTS

Discussions with W.Waldo MacKay on the symplectic properties of the transport matrices, in particular the implications of Eq.(3), are acknowledged.

REFERENCES

1. A.U.LUCCIO and N.L.D'IMPERIO: *Simbad User's Manual. Version v.1.36*. Technical Report C-A/AP/222, <http://www.agrshichome.bnl.gov/People/luccio/>, Brookhaven National Laboratory. Upton, NY, October 2005.
2. R.W.HOCKNEY and J.W.EASTWOOD: *Computer Simulation Using Particles*. Adam Hilger, IOP Publishing, New York, 1988.
3. H.GROTE and F.CH.ISELIN: *The MAD program, Vers.8.19*. Technical Report CERN/SL/90-13, Geneva, Switzerland, 1996.
4. A.U.LUCCIO and N.L.D'IMPERIO: *Eigenvalues of the One-turn Matrix*. Technical Report C-AD/AP/126, Brookhaven National Laboratory. Upton, NY, December 2003.
5. W.GROPP, E.LUSK, A.SKJELUM: *Using MPI*. MIT Press, 1998.
6. N.MALITSKY and R.TALMAN: *Unified Accelerator Libraries*. Technical Report AIP 391, <http://www.ual.bnl.gov>, American Institute of Physics, Melville, New York, 1996.
7. BRUN, et al.: *The ROOT Framework*. Technical Report AIHEPNH-96, <http://root.cern.ch>, CERN, 1996.