Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational Physics 222 (2007) 265–286
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Compression of magnetohydrodynamic simulation data using singular value decomposition

D. del-Castillo-Negrete, S.P. Hirshman *, D.A. Spong, E.F. D'Azevedo

Oak Ridge National Laboratory, Oak Ridge, TN 37831, United States

Received 18 April 2006; received in revised form 7 July 2006; accepted 24 July 2006
Available online 7 September 2006

Abstract

Numerical calculations of magnetic and flow fields in magnetohydrodynamic (MHD) simulations can result in extensive data sets. Particle-based calculations in these MHD fields, needed to provide closure relations for the MHD equations, will require communication of this data to multiple processors and rapid interpolation at numerous particle orbit positions. To facilitate this analysis it is advantageous to compress the data using singular value decomposition (SVD, or principal orthogonal decomposition, POD) methods. As an example of the compression technique, SVD is applied to magnetic field data arising from a dynamic nonlinear MHD code. The performance of the SVD compression algorithm is analyzed by calculating Poincaré plots for electron orbits in a three-dimensional magnetic field and comparing the results with uncompressed data.

Keywords: Magnetohydronamics; Singular value decomposition; Generalized low rank approximation; Data compression; Numerical methods

1. Introduction

Present day simulations [1,2] of magnetically-confined plasmas on magnetohydrodynamic (MHD) space and time scales are limited by computational resources to relatively modest values of the magnetic Lundquist number S (the ratio of the resistive diffusion time to the Alfvén transit time) in the range 10^4–10^5. To simulate fusion plasmas with parameters characterizing future burning plasma experiments like the International Thermonuclear Experimental Reactor (ITER [3]) device, where $S \sim 10^9$, computations will be required on much finer meshes and shorter time steps than present computers are capable of handling. Associated with this higher spatial–temporal resolution will be the production of large amounts of numerical data needed to represent the plasma fluid state (most notably, the magnetic field vector **B** and the fluid velocity **v**) at various positions and times.

* Corresponding author. Tel.: +1 865 574 1289.
E-mail address: hirshmansp@ornl.gov (S.P. Hirshman).

0021-9991/$ - see front matter
doi:10.1016/j.jcp.2006.07.022

At the high temperatures in a burning plasma, it will be necessary to augment the classical (short-mean-free path) fluid description of MHD with additional "closure" relations relating the viscosity tensor to the fundamental MHD state variables (density, temperature and fluid velocity). One way of computing these closure quantities is to use a particle-based simulation in conjunction with the evolving MHD fields. Thus, MHD data will be frequently exchanged with a particle-following calculation designed to compute the viscosity tensor from particle motion in the complex magnetic field. The resulting components of the viscosity tensor, appropriately computed on the fine spatial mesh required by the MHD simulation, will be used in the MHD simulation to evolve the fluid equations.

The exchange of fluid and particle-based viscosity data will occur frequently during the time evolution of the simulated plasma. In a multi-processor simulation, it also is necessary to "broadcast" this data to numerous processors. It is therefore useful to consider methods for compressing this data as much as possible while retaining the fine-scale structures needed to accurately reproduce the desired physics processes.

In this paper, we apply singular value decomposition (SVD) to three-dimensional (3D) MHD data to achieve significant data compression. (SVD is known by various other names – principal orthogonal decomposition, or POD; empirical orthogonal functions, or EOF; and principal component analysis, PCA.)

It should be noted that SVD analysis has been previously used to extract dominant features from two-dimensional (2D) turbulence measurements in fusion plasmas [4]. In Ref. [4], SVD was used to expand the fluctuation data using a smaller number of terms as compared to a Fourier (FFT) analysis. This was found to give an advantage to the SVD method in identifying the basic modes and therefore attempting to understand the complex MHD phenomena being analyzed. This form of data compression, applied to real experimental data, is similar to the method described here which is however extended to 3D numerical data. The present method could be applied to 3D experimental data as well. In addition, SVD techniques have been applied [5] to extract 2D current sheets for designing coils for compact stellarators.

The paper is organized as follows. We first review SVD compression on two-dimensional discrete data in Section 2.1. Since subsequent applications will involve compression of three-dimensional magnetic data, we discuss in Section 2.2 techniques by which SVD can be extended to three-dimensions: (a) simple planar slices; (b) stacking (folding) methods and (c) generalized low rank approximation (GLRA [6]). Of these, the GLRA method performs best for our data. In Section 3 we describe several methods for applying SVD to data in both rectangular and non-rectangular domains. We have found that a domain mapping method provides, for our data at least, the largest compression at a given level of accuracy. Section 4 presents an application of GLRA to 3D magnetic field data obtained from a time slice of an $M3D^1$ simulation. There we compare Poincaré plots computed either with the uncompressed or compressed data. Appendix A presents methods we have developed for smoothly "extending" data limited to non-rectangular domains so that discontinuous artifacts are not introduced to pollute the SVD energy spectrum.

2. Application of SVD to data compression

This work shows how to use SVD (or POD) compression techniques to represent magnetic field data in a compact form. The data set of interest here is defined in a toroidal domain that represents the interior of a tokamak or stellarator, which are devices used to confine high temperature fusion plasmas using magnetic fields and internally-generated plasma currents. In the simplest equilibrium case, the magnetic field is tangent to nested tori known as flux surfaces that fill the toroidal domain. Thus, although the magnetic field is in general three-dimensional, in this case the geometry is effectively two-dimensional (2D) when restricted to a given flux surface.

As an introduction to the use of SVD methods for MHD data compression, Section 2.1 reviews standard (2D) SVD. This technique is applied to compress two-dimensional (2D) planar data slices taken on fixed magnetic surfaces. The MHD data sets considered in subsequent sections are intrinsically three-dimensional (3D). The generalization of SVD compression methods to such 3D data is described in Section 2.2.

The orthogonal real-space coordinates used to describe the domain for the MHD problem are the cylindrical coordinates (R, φ, Z). In these coordinates, the boundary of the toroidally-confined plasma is generally non-rectangular (irregular). For example, it may be a nearly circular (or elliptical) torus. It is therefore useful to introduce magnetic flux-based curvilinear coordinates, denoted by (s, θ, φ), which describe a set of

nested toroidal surfaces. These are defined so that the $s = 1$ surface defines the plasma boundary (where the plasma pressure gradient vanishes) and $s = 0$ is the "magnetic axis", which is the singular axis of this "polar-like" coordinate system. Here, $0 \leqslant \theta \leqslant 2\pi$ denotes the (poloidal) angle variable, and φ can be chosen to be the cylindrical toroidal coordinate. The mapping between these two coordinate systems is described in Section 3.2.

2.1. Review of SVD in 2D

SVD is a well-known technique for extracting dominant "features" and coherent structures from 2D data [7] and "compressing" that information into a few low order "weights" (singular values) and associated orthonormal eigenfunctions [8]. Let $A(x, y)$ be a scalar field representing one of the components of the magnetic field vector. (Here, x, y are either local cylindrical coordinates R, Z in a fixed toroidal plane, or angular coordinates θ, φ on a fixed magnetic flux surface.) It is convenient to define a 2D *rectangular* grid (x_i, y_j) for $i = 1, N_I$ and $j = 1, N_J$. The discretized field data specified at these grid points are denoted by the $N_I \times N_J$ matrix $A_{ij} = A(x_i, y_j)$. The maximum number of terms in the SVD expansion of A_{ij} is $N_{\mathrm{SVD}} = \min(N_I, N_J)$. The SVD decomposition (see below) of the data matrix A_{ij} yields a set of N_{SVD} positive weights (*singular* values) and associated orthonormal eigenfunctions in both the i and j indices. While SVD compression is well-known in image processing and other areas, it has not been previously explored in the context of large-scale, particle-based numerical simulations in fusion plasmas. The advantage over Fourier decomposition is that structures of various spatial scales can be expressed in the lowest-order SVD eigenfunctions, a property that SVD shares with 2D wavelet decomposition.

The fundamental SVD theorem states that any 2D matrix A_{ij} may be expanded as a tensor product sum as follows:

$$A_{ij} = \sum_{k=1}^{N_{\mathrm{SVD}}} w_k u_k(x_i) v_k(y_j) \tag{1}$$

In Eq. (1), each value of k labels a single positive (or possibly zero) SVD singular value w_k, which belongs to a monotonically decreasing sequence ($w_{k+1} \leqslant w_k$). The proper orthogonal modes (POM), or SVD "eigenfunctions", are denoted by $\{u_k\}$ and $\{v_k\}$. Efficient numerical methods exist [8,9] for computing the matrix elements $u_{ik} \equiv u_k(x_i)$ and $v_{jk} \equiv v_k(y_j)$, as well as the singular values w_k, appearing in Eq. (1). These discrete "eigenfunctions" comprise an orthonormal basis for the discrete x_i and y_j spaces, respectively:

$$\begin{aligned} \langle u_k, u_{k'} \rangle &\equiv \sum_{i=1}^{N_I} u_{ik} u_{ik'} = \delta_{kk'} \\ \langle v_k, v_{k'} \rangle &\equiv \sum_{j=1}^{N_J} v_{jk} v_{jk'} = \delta_{kk'} \end{aligned} \tag{2}$$

The indices (k, k') in Eq. (2) vary from 1 to N_{SVD}. Note that throughout this paper, only *discrete* data are represented in Eq. (1). Such data arise naturally from numerical output on discrete angular and radial meshes. When values are required at points *between* mesh points (as for example when following particle orbits, see Section 4), biquadratic interpolation over the discrete i, j mesh values is used. (Alternatively, spline or trigonometric polynomials could be fit to the SVD modes to perform this interpolation, but that is not done here.)

At first glance, the SVD sum in Eq. (1) does not seem to be a promising candidate for compression, since there are $N_{\mathrm{SVD}}(1 + N_I + N_J)$ terms to compute and store. This number exceeds the original $N_I N_J$ data points. However, a key observation that makes SVD viable for compression is that the singular values w_k often decay rapidly with increasing k. This implies that only a small fraction of all the terms in Eq. (1) are typically needed to accurately represent the dominant (physically relevant) features of the data. Based on this, we introduce a rank-r approximation for A_{ij}:

$$A_{ij}^{(r)} = \sum_{k=1}^{r} \mu_k u_{ik} v_{jk} \tag{3}$$

and inquire how, for a given rank r, to choose the μ_k weight factors to best match the original data matrix. To do this, it is convenient to define the rank-r error as the following (squared) Frobenius norm:

$$e(r) = \sum_{i=1}^{N_I} \sum_{j=1}^{N_J} \left| A_{ij} - A_{ij}^{(r)} \right|^2 \tag{4}$$

Note that when $r = N_{\mathrm{SVD}}$, the error $e(N_{\mathrm{SVD}}) = 0$ for $\mu_k = w_k$. For $r < N_{\mathrm{SVD}}$, the rank-r truncation error may be written [using the orthonormality relations in Eq. (2)] as:

$$e(r) = \sum_{k=1}^{r} (\mu_k - w_k)^2 + \sum_{k=r+1}^{N_{\mathrm{SVD}}} w_k^2 \tag{5}$$

The first term in Eq. (5) vanishes by choosing $\mu_k = w_k$ for $k = 1, \ldots, r$. The second term is minimized since the SVD singular values are ordered to decrease monotonically with k. Thus, Eq. (5) simplifies to:

$$e(r) = \sum_{k=r+1}^{N_{\mathrm{SVD}}} w_k^2 \tag{6}$$

It can be shown [8] that the SVD solution in Eq. (3), with $\mu_k = w_k$ yields the minimum norm over *all possible* rank-k approximations [not just for the simple form we chose in Eq. (3)].

The result in Eq. (6) is similar to the Parseval relation in the theory of continuum orthogonal function decomposition. Its physical interpretation is also similar. Using Eqs. (4) and (6), note that the "energy content" $E = \sum_{i=1}^{N_I} \sum_{j=1}^{N_J} A_{ij}^2$ of the original data set can be written $E = e(0) = \sum_{k=1}^{N_{\mathrm{SVD}}} w_k^2$. Thus, $e(r)$ in Eq. (5) may be interpreted as the energy *deficit* in the rank-r compression, showing that SVD compression is equivalent to a truncation based on the energy content of the data. A rank-r SVD approximation will be "good" provided the normalized energy $\eta(r)$ is "small", where

$$\eta(r) \equiv \frac{e(r)}{e(0)} \tag{7}$$

In our analysis of model data and actual magnetic data (Section 4), it is generally observed that even for relatively low ranks $r \ll N_{\mathrm{SVD}}$, values of $\eta(r) \sim 10^{-6}$–10^{-8} can readily be achieved. This yields a large SVD compression ratio R_{C}, which is the ratio of the original $N_I N_J$ data points to the number of terms in the rank-r approximation in Eq. (3):

$$R_{\mathrm{C}} \equiv \frac{N_I N_J}{r(N_I + N_J + 1)} \tag{8}$$

For the case when $N_I \sim N_J \gg 1$, this reduces to $R_{\mathrm{C}} \sim N_{\mathrm{SVD}}/(2r) \gg 1$, signifying a large compression ratio can be achieved whenever $\eta(r) \ll 1$ is satisfied for $r \ll N_{\mathrm{SVD}}$.

The residual energy expression in Eq. (6) may also be used as an adaptive quantitative measure of the minimum rank necessary to achieve a prescribed accuracy of the data fit. Such adaptation of the rank may be necessary as the MHD solution evolves shorter spatial scales in time.

2.2. SVD compression in 3D

The MHD data sets we wish to compress are distributed throughout a three-dimension (3D) toroidal volume. While the SVD theorem is not directly applicable for dimensions greater than two, it is possible to extend the standard procedure to compress 3D data by applying the basic SVD procedure to a sequence of 2D data structures. We will briefly consider three methods for obtaining low rank approximations for 3D data structures in a torus. In what follows, the array $A(s_i, \theta_j, \varphi_k) \equiv A_{ijk}$ represents one of the scalar MHD potential functions or a component of the magnetic field (or the vector fluid flow field), typically represented in a flux coordinate system (see Section 3.2). For the MHD problem considered here, the integer indices (i, j, k) correspond respectively to discrete radial, poloidal and toroidal angle points, with the ranges of i, j defined above, and $1 \leqslant k \leqslant N_K$.

2.2.1. Planar slices

A straightforward approach is to apply SVD *separately* to the data in each toroidal plane (corresponding to individual values of k). There will now be an indexed set (with respect to k) of SVD weights and eigenfunctions. In this case, the compression ratio scales as in the 2D problem, Eq. (8), with r now denoting the *maximum* rank of the 2D SVDs over all planes. However, as shown in the following two sections, the data compression achievable in 3D can actually be much larger than this, scaling as the *square* of the 2D compression ratio, so this simple method is not efficient. However, an advantage of the planar slice method is that only $2r$ multiplies and r additions are needed to evaluate the value of the data at any given point of the i, j, k cube. This turns out to be smaller (by a factor of r) than the number of operations required by the two higher-compression methods described below. Thus, there is a trade-off between a large compression ratio and the number of elementary operations needed to reconstruct the original data.

2.2.2. Stacking (folding) methods

Another method for compressing higher dimensional data is called *stacking* [10] or *matrix unfolding*. Due to their periodicity, either of the angular coordinates (ϑ, φ) can be *folded* together with the radial coordinate, or with each other, to create a new (stacked) coordinate. Folding preserves the continuity of function values with respect to the stacked coordinate, due to the periodicity of the angle coordinates. This is important for the sequential application of SVD because – as will be discussed in Section 3 – the SVD spectrum generally decays rapidly (yielding large compression ratios) *only* in the absence of discontinuities in the data. (The exception to this is when the discontinuities are aligned along coordinate lines.)

Folding reduces the 3D data matrix to be decomposed to an equivalent 2D matrix for which SVD applies. The resulting SVD is unfolded and SVD is again applied to each of the 2D basis functions that correspond to significant weights in the folded decomposition. The significant result of this procedure is that the net SVD compression scales as the *square* of the 2D compression ratio.

The folding (and unfolding) of the general 3D data matrix A_{ijk} will now be described. It is convenient to define a 2D integer *stacking* coordinate operator, $[j, k]$, as follows:

$$[j,k] \equiv j + N_J(k-1) \tag{9}$$

The range of this operator is $(1, N_J N_K)$. The stacking operator provides a mapping of the 3D data into the 2D $i, [j, k]$ *folded* plane. The only restriction on the stacking $[j, k]$ is that the coordinate indexed by j in Eq. (9) must be one of the periodic angles. This is necessary to preserve the continuity of data in the folded space. Observe that the data will vary continuous with $[j, k]$ if, and only if, there are no jumps between the $j = 1$ and $j = N_J$ points when k is incremented by one in Eq. (9). This is satisfied by requiring the j-coordinate to be periodic.

Now SVD can be applied to construct a rank-r_1 approximation of the 2D folded data array $A_{i[j,k]} = A_{ijk}$:

$$A_{i[j,k]} = \sum_{\alpha=1}^{r_1} w_\alpha u_{i\alpha} v_{[j,k]\alpha} \tag{10a}$$

The original 3D array can be recovered from Eq. (10a) by *unstacking* the vectors $v_{[j,k]\alpha}$ for each SVD index α. Further compression is then achieved by applying SVD to construct rank-r_2 approximations for each of the r_1 2D (unfolded) arrays $v_{[j,k]\alpha}$:

$$v_{[j,k]\alpha} = \sum_{\beta=1}^{r_2} \tilde{w}_{\alpha\beta} \tilde{u}_{j\alpha\beta} \tilde{v}_{k\alpha\beta} \tag{10b}$$

Eq. (9) is used to perform the unfolding transformation back to the original j, k coordinates in performing (for each α) the SVD indicated in Eq. (10b).

Repeating the calculations leading to the 2D compression ratio given in Eq. (8) yields the 3D compression ratio for this method:

$$R_C = \frac{N_I N_J N_K}{r_1[1 + N_I + r_2(1 + N_J + N_K)]} \tag{11a}$$

Assuming comparable dimensions $N_I \sim N_J \sim N_K = N$ and $N \gg r_2 \gg 1$ yields the approximate compression ratio

$$R_C \sim 2(N/2r_1)(N/2r_2) \tag{11b}$$

where $N/2r_1$ and $N/2r_2$ are the 2D compression ratios (see Eq. (8)). The fact that Eq. (11b) is *symmetric* in r_1 and r_2 suggests that the compression ratio is independent of the choice of folding angle (a result consistent with a limited number of calculations we have performed using folding). Thus the 3D compression ratio scales like twice the *product* of the 2D compression ratios. The price for this increased compression (compared with the planar slices method) is the larger number of elementary arithmetic computations [$r_1(1 + 2r_2)$ multiplications, compared with $2r$] that must be performed to reconstruct the original data matrix. (Of course, r_1 and r_2 may be smaller than r, so the estimate given here is only suggestive of the increased computational complexity of reconstructing data with this method.)

2.2.3. Generalized low rank approximation

A generalized low rank approximation method [6] for 3D compression has been developed which is particularly well suited for the MHD data considered here. This is because the MHD data variation in one of the coordinate directions (the toroidal angle) is both periodic and slowly varying. The GLRA method is an iterative algorithm for minimizing the (squared) Frobenius norm of the difference between the 3D data matrix A_{ijk} and a specialized form called the "generalized low rank approximation":

$$\sum_{k=1}^{N_K} \|\widetilde{A}_k - LD_kR^{\mathrm{T}}\|^2 \tag{12}$$

For each value of k in Eq. (12), $(\widetilde{A}_k)_{ij} \equiv A_{ijk} \in R^{N_I \times N_J}$ is a 2D matrix with dimension $N_I \times N_J$. The k-coordinate is chosen to be the toroidal angle (slowly varying periodic coordinate direction). The two matrices $L \in R^{N_I \times r_1}$ and $R \in R^{N_J \times r_2}$ are *independent* of the index k and have r_1 and r_2 orthonormal columns, respectively. (Here r_1 and r_2 are the *reduced ranks*.) The matrices $D_k \in R^{r_1 \times r_2}$ are generally *not* diagonal except in the 2D case where $N_K = 1$. In that case, the matrices L and R reduce to the standard SVD matrices U and V. In 3D, L, R, and D_k are determined by an iterative procedure [6] that minimizes (at least locally) the norm in Eq. (12), for prescribed values of r_1 and r_2. The resulting compression ratio R_C is:

$$R_C = \frac{N_I N_J N_K}{N_I r_1 + N_J r_2 + N_K r_1 r_2} \tag{13}$$

For the usual case when the last term in the denominator of Eq. (13) is dominant, the compression ratio for this method will scale as

$$R_C \sim (N_I/r_1)(N_J/r_2) \tag{14}$$

which is (four times) the *product* of the 2D SVD compression ratios associated with the average (over k) matrices L and R.

The two methods described in Sections 2.2.2–2.2.3 generally yield comparable results for the compression ratio as a function of the rank reduction (note that GLRA yields *twice* the ratio of the stacking method). The recursive method converges quite rapidly (typically in two or three iterations). Moreover, since GLRA requires SVD of a larger number of substantially smaller matrices than the folding method, the computational speed of the two methods tends to be similar.

3. Effect of domain geometry on data compression

SVD decomposes data as a sum of Cartesian (tensor) products (Eq. (1)). It is therefore expected to work efficiently only when the data are prescribed in rectangular domains. For our purposes, a rectangular domain of a 2D space with coordinates (w, z) is a domain whose boundaries are $w = \text{const.}$ and $z = \text{const.}$ coordinate lines. We shall consider two classes of examples that occur naturally in MHD problems, which illustrate the effects of geometry on SVD compression.

3.1. Rectangular domains

First consider a rectangular domain, which can be illustrated by MHD *equilibrium* fields with *nested* toroidal magnetic surfaces. In this case it is useful to decompose the data *independently* on each surface. Within each 2D surface section, the domain is spanned by (θ, φ), where both angular coordinates have the range $[0, 2\pi]$. To illustrate the SVD compression of magnetic data in this domain, we consider the magnetic field magnitude $|B|$ in a quasi-poloidally symmetric stellarator [11] (QPS). Fig. 1a shows a 2D flux surface plot (on one magnetic surface) of the $|B|$ contours computed on a 100×100 grid. Figs. 1b–d show the rank 2, 4 and 6 SVD (POD) approximations to the data. The rank-2 approximation already captures the main coherent (longer wavelength) features of the field, while the rank-6 approximation captures most of the fine-scale details of the field as well. Fig. 2 shows the decay of the singular values w_k as function of the rank k for the QPS data. Note that for rank $k = 6$, the 2D compression ratio is $R_C \sim 10$, and the residual energy in the higher-order (truncated) modes is $\eta(6) \sim O(10^{-6})$. This may justify ignoring the higher-order SVD components (at least for equilibrium calculations).

3.2. Non-rectangular domains

There are many situations of practical importance in which the data may be specified over an irregularly-shaped domain. We have already mentioned the case of a toroidally-confined plasma with a circular boundary. Several procedures can be used to extend SVD techniques to such domains. Two categories of such

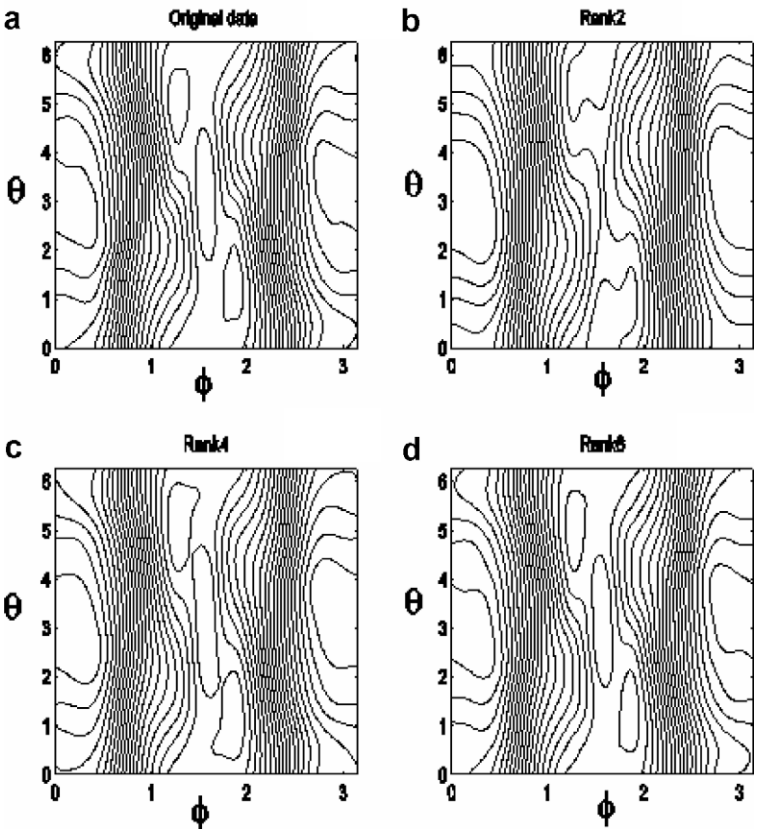

Fig. 1. Effect of SVD data compression for a QPS (3D stellarator) magnetic field $|B|$ computed from a VMEC equilibrium. Fig 1a shows the originally computed contours of $|B|$ vs. the two angles (θ, φ) on a single magnetic surface. Figs. 1b–d show the reconstructed contours using various rank-order truncations for the SVD spectrum.

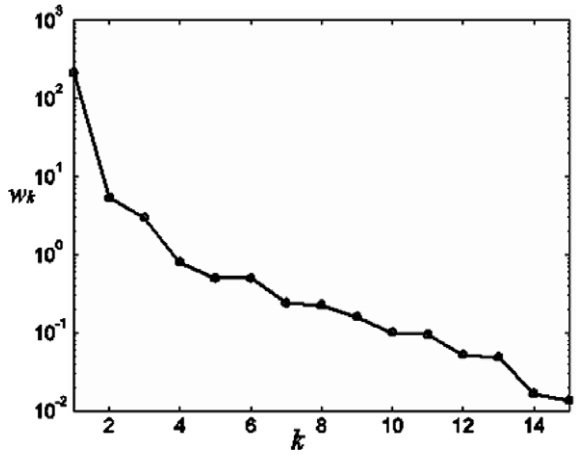

Fig. 2. SVD singular values as a function of the rank for the QPS data shown in Fig. 1.

algorithms are discussed below: (1) region mapping; and (2) grid extension. The region mapping algorithm *maps* the irregular domain into a rectangular one, while the grid extension method attempts to fill the region between the true (irregular) boundary and a minimum circumscribing rectangle with "suitable" values. The

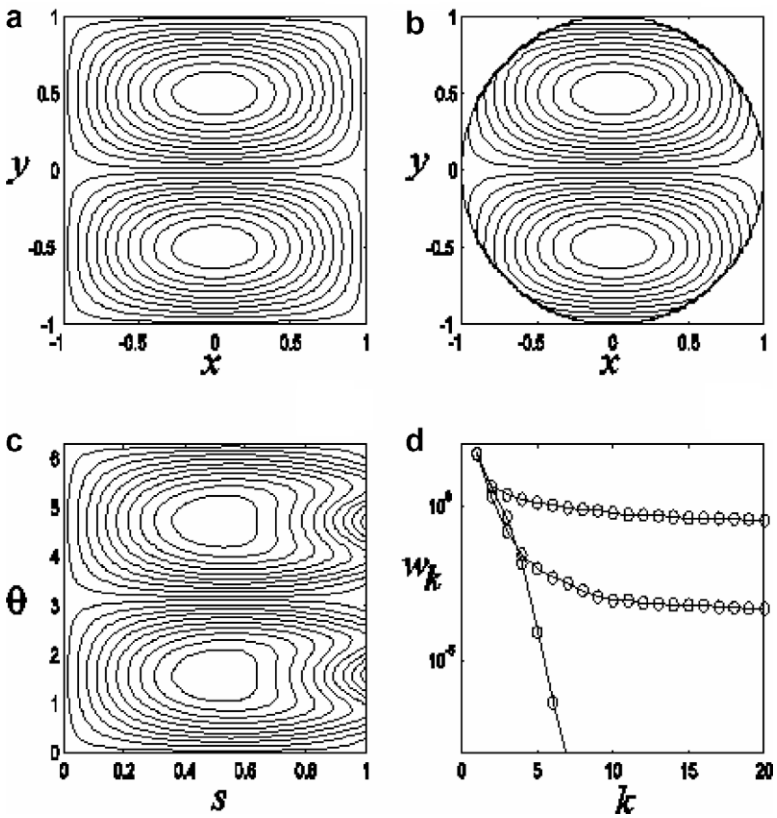

Fig. 3. Original model 2D field is shown on the square domain ($-1 < x < 1$, $-1 < y < 1$) in Fig. 3a. Fig. 3b shows the data masked by the circular plasma boundary. Fig. 3c shows the data in the flux (polar) coordinates system (s, θ). Fig. 3d shows the decay with rank of the SVD spectrum for various grid extension and mapping options. The curve at the top corresponds to the direct SVD decomposition of the non-rectangular domain in Fig. 3b, the middle curve is the spectrum with the data extended using the grid extension algorithm discussed in Appendix A, and the bottom curve depicts the spectrum of the SVD applied to the flux-coordinates data in Fig. 3c.

grid extensions are sensitive to the *continuity* of the "ghost" region with the real (interior) domain and may result in the slow decay of the SVD singular value spectrum if improperly applied.

3.2.1. Region mapping

For the MHD data considered here, we have found that mapping the non-rectangular domain defined in cylindrical coordinates into a rectangular one defined in flux coordinates leads to maximum SVD compression ratios. (This might be expected, since no arbitrary "ghost" data are introduced.) As previously described, in each toroidal plane $\phi = \text{const.}$, there is a natural magnetic "polar" coordinate system which describes a nested set of *equilibrium* surfaces. Note that here we refer to the underlying coordinate system as nested. This is different from the actual 3D MHD magnetic surfaces which, even if they exist, need not be nested. Recall that the coordinate ranges are $0 \leqslant s \leqslant 1$ and $0 \leqslant \theta \leqslant 2\pi$, so that (s, θ) defines a rectangular domain. For the MHD data generated by the *M3D*[1] code, the coordinate mapping is available from the underlying *VMEC* [12] equilibrium. In the case of "stellarator symmetry", where $R(s, \theta, \varphi) = R(s, -\theta, -\varphi)$ and $Z(s, \theta, \varphi) = -Z(s, -\theta, -\varphi)$, the mapping has the following form:

$$R(s,\theta,\varphi) = \sum_{m,n} R_{mn}(s)\cos(m\theta - n\varphi)$$
$$Z(s,\theta,\varphi) = \sum_{m,n} Z_{mn}(s)\sin(m\theta - n\varphi) \tag{15}$$

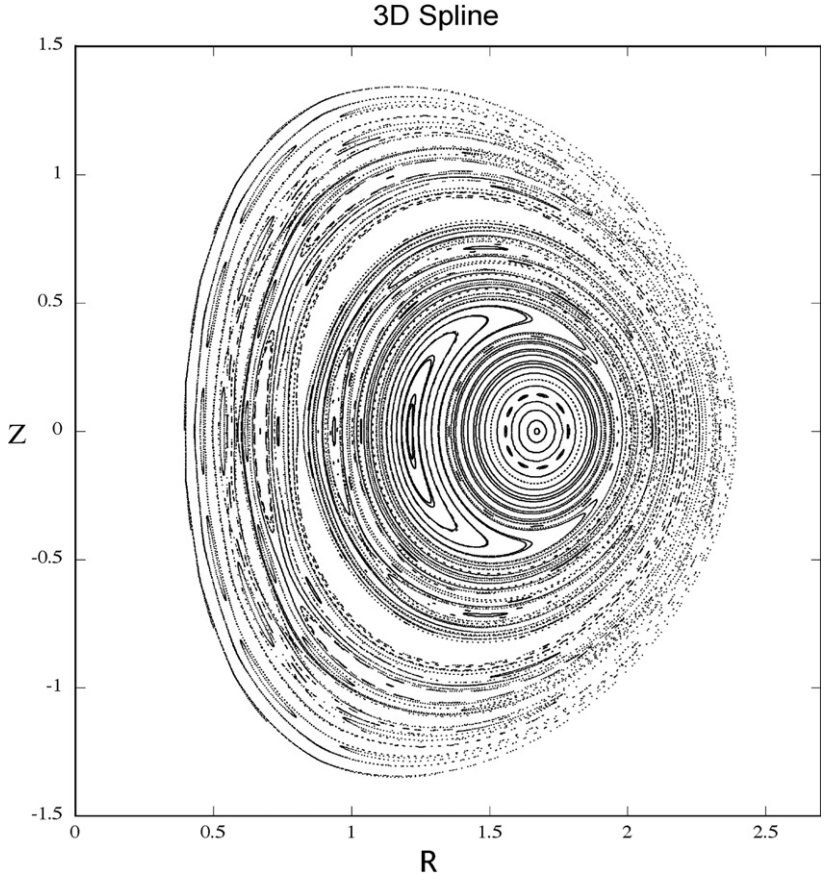

Fig. 4a. Poincaré puncture plot displayed in cylindrical spatial coordinates for the plane $\phi = 0$, using data for a magnetic field computed using *M3D* with a finite $m = 1$ island. Electron orbits were computed in flux coordinates and mapped back to real space. The magnetic field was obtained using spline fits to the original data set consisting of a mesh of size $235 \times 79 \times 49$ in θ, s, and ϕ flux coordinates, respectively.

The equilibrium calculation [12] provides the mapping coefficients $R_{mn}(s)$, $Z_{mn}(s)$ in Eq. (15), which can be generalized to cases of non-symmetric plasmas.

Using Eq. (15), the magnetic data can be converted to a mesh *cube* in (s, θ, φ) space to which the GLRA method described in Section 2.2 can be applied. This method is used in Section 4 to follow particle orbits using compressed data for a realistic, 3D magnetic field.

3.2.2. Grid extension algorithms

For situations where a domain mapping of the form given in Eq. (15) is inconvenient to use, it is possible to "extend" the data into the region outside the irregular data domain. This "ghost" region is initially devoid of any data and extends between the data boundary and a minimum circumscribing (bounding) rectangle. (In the case of MHD plasmas, this region consists of zero pressure-gradient plasma and is often referred to as the "vacuum".) The method of extension is not unique and has a strong influence on the decay rate of the SVD. If no attempt is made to extend the data in a continuous manner, then the SVD spectrum can decay very slowly compared with the case of a rectangular domain filled with similar continuous data. (See Section 3.2.3 below for an example of this.)

Appendix A describes two methods of grid extension which preserve data continuity and higher-order derivatives.

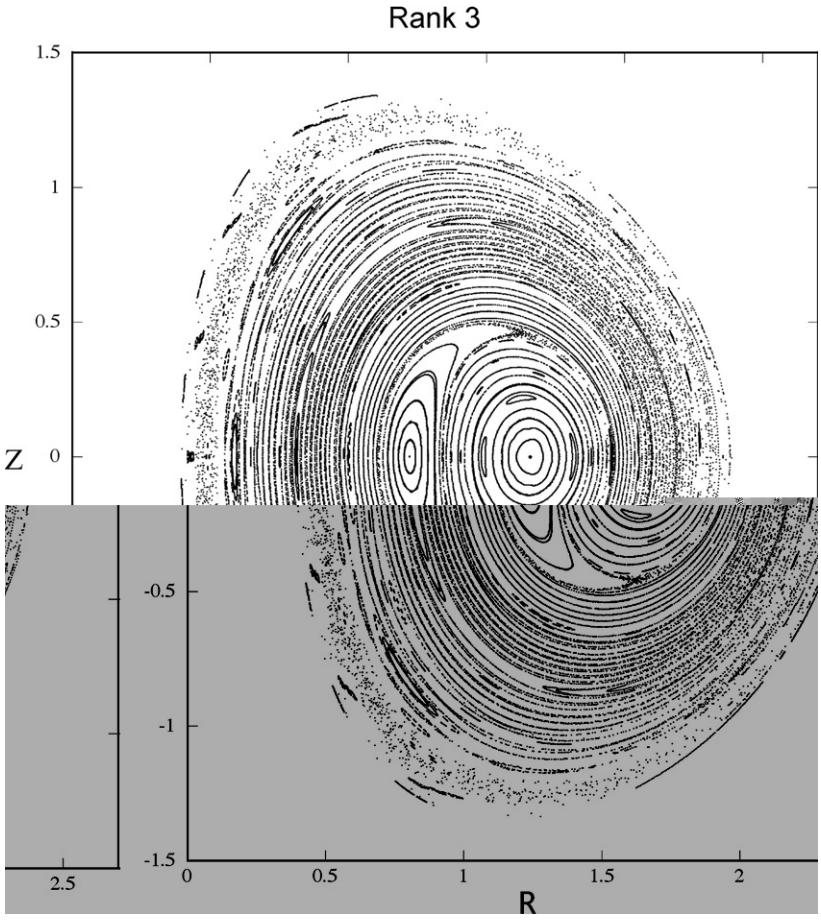

Fig. 4b. Same Poincaré puncture plot as shown Fig. 4a, but using a rank-3 approximation (with a compression ratio >1000) for the magnetic field data.

3.2.3. Example: compression in an irregular domain

The effects of irregular boundaries on SVD compression in 2D can be illustrated by using a simple scalar field to model one of the components of the magnetic field:

$$b(x,y) = \cos(\pi x/2)\sin(\pi y) \tag{16}$$

Note that since this scalar is itself a Cartesian product, its SVD over the rectangular domain $(-1 \leqslant x, y \leqslant 1)$is composed of only one singular value. Contours of $b(x, y)$ over this domain are shown in Fig. 3a. For a confined plasma, the boundary of the plasma is generally not rectangular and the data are known (from a simulation, for example) only inside an irregular boundary. For this example, we choose a *simple* non-rectangular (circular) boundary for the plasma:

$$x^2 + y^2 = r^2 \tag{17}$$

Restricting even this very simple data set to a non-rectangular domain can have significant effects on the resulting SVD (see Fig. 3d). Three methods for performing the SVD were considered: (1) the data set was extended into the rectangular bounding box by simply zeroing the field values in the "vacuum" region, as shown in Fig. 3b; (2) the grid was extended while trying to maintain continuous data values at the boundary; and (3) the circular domain was mapped into flux coordinates (s, θ) using the simple polar map $x = s\cos\theta$, $y = s\sin\theta$, which yields a rectangular domain in s and θ [but *not* a Cartesian product for $b(s, \theta)$]. The mapped data are plotted in Fig. 3c. The first method produces a discontinuity at the circular boundary, which has two deleterious effects on the SVD spectrum. First, the spectrum decays slowly with rank. Second, there is a plateau in

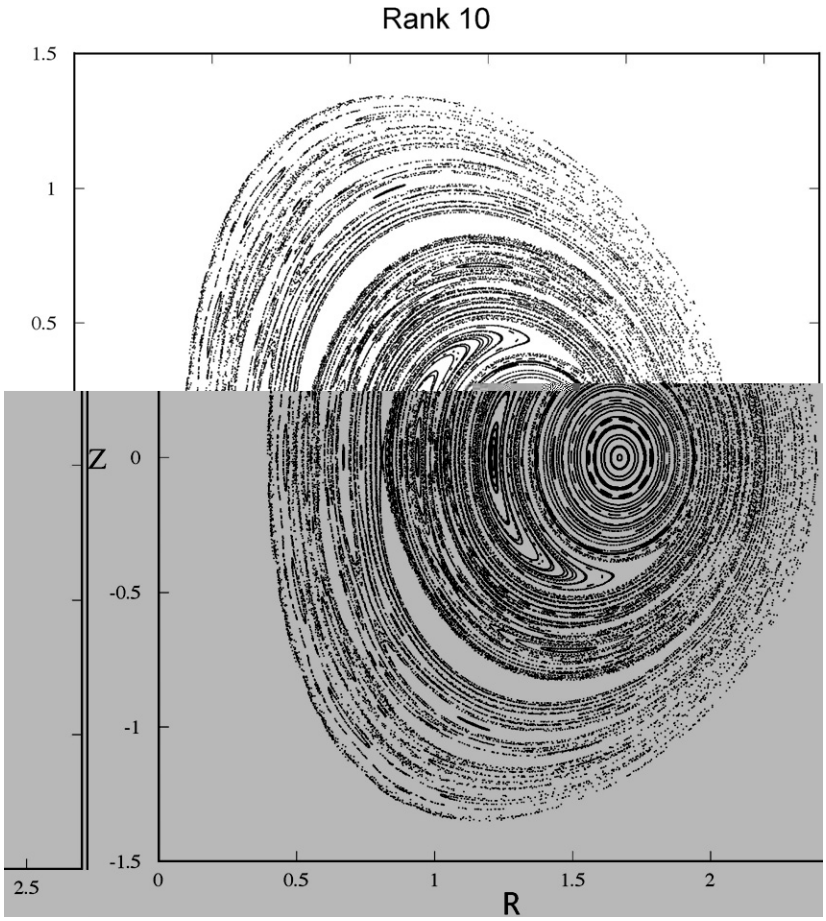

Fig. 4c. Poincaré puncture plot using rank-10 SVD approximation for the fields in Fig. 4a.

the SVD spectrum where the SVD singular values show no appreciable reduction with rank. This implies very little data compression would be possible for this decomposition. Both of these effects are similar to the impacts on Fourier spectra when discontinuities in the data exist. Fig. 3d compares the decay rate of the SVD singular values for the various algorithms. The rapidly-decaying curve corresponds to the region-mapping algorithm, while the slowest decay is obtained when the data are simply zeroed in the vacuum region. Grid extension, while maintaining continuous data values, produces a result intermediate between these two extremes. Of course, if the grid extension was "optimal", it would reproduce the original data and have only a single non-zero SVD weight!

4. Application of SVD compression to MHD fields: calculation of electron particle orbits

In this section we use SVD compression to compress the magnetic field data needed to evolve particle orbits in a realistic 3D magnetic field. The *DELTA5D* [13] particle-based Monte-Carlo code was used to compute the particle orbits. The magnetic data set was obtained from a nonlinear tokamak plasma simulation using the *M3D*[1] MHD code. The data correspond to a plasma state that has evolved to a non-nested magnetic topology with a large internal magnetic island at an interior magnetic flux surface where the mean rotation number is $\iota = 1$. The rotation number is the ratio of the mean number of circuits of the magnetic field in the poloidal and toroidal directions, respectively. An island is a change in the magnetic topology which splits the originally nested magnetic surfaces due to resonant perturbations of the magnetic field occurring at low order rational values of ι. Several other smaller islands are also present throughout the plasma volume (see Fig. 4a).

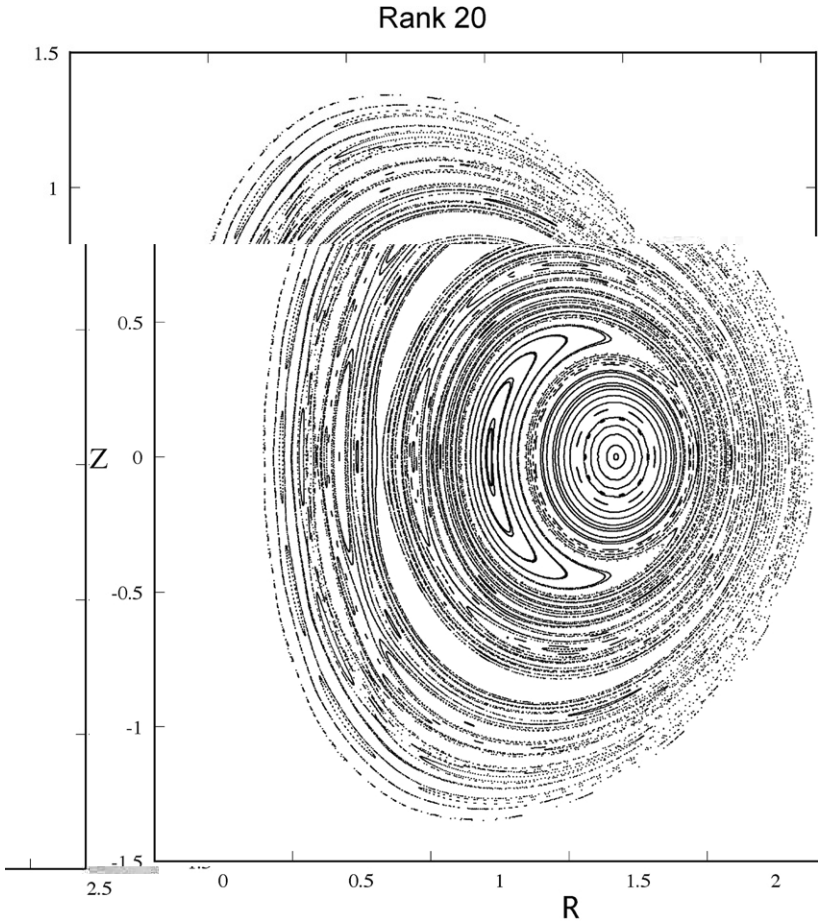

Fig. 4d. Poincaré puncture plot using rank-20 SVD approximation for the fields in Fig. 4a.

The orbit evolution is obtained by integrating the particle drift orbit equations [13]. We will consider electron orbits as the simplest example of how SVD data compression can be integrated with a particle orbit code (the effects of cross-field magnetic drifts and particle trapping will be ignored in this simulation).

Recall that SVD compression of magnetic data is, as discussed in Section 3, most naturally performed in terms of the flux "action-angle" coordinates (s, θ) defined by the coordinate mapping Eq. (15). The chain rule can be used to express the motion in cylindrical space (R, φ, Z)in terms of action-angle coordinate velocities as follows:

$$\begin{aligned} \dot{R} &= R_s\dot{s} + R_\theta\dot{\theta} + R_\varphi\dot{\varphi} \\ \dot{Z} &= Z_s\dot{s} + Z_\theta\dot{\theta} + Z_\varphi\dot{\varphi} \end{aligned} \tag{18}$$

Table 1
RMS error and compression ratios for various SVD rank approximations corresponding to the data in Figs. 4a–4d

Rank	Normalized RMS error	Compression ratio
3	0.01	660
10	0.001	113
20	0.00005	35

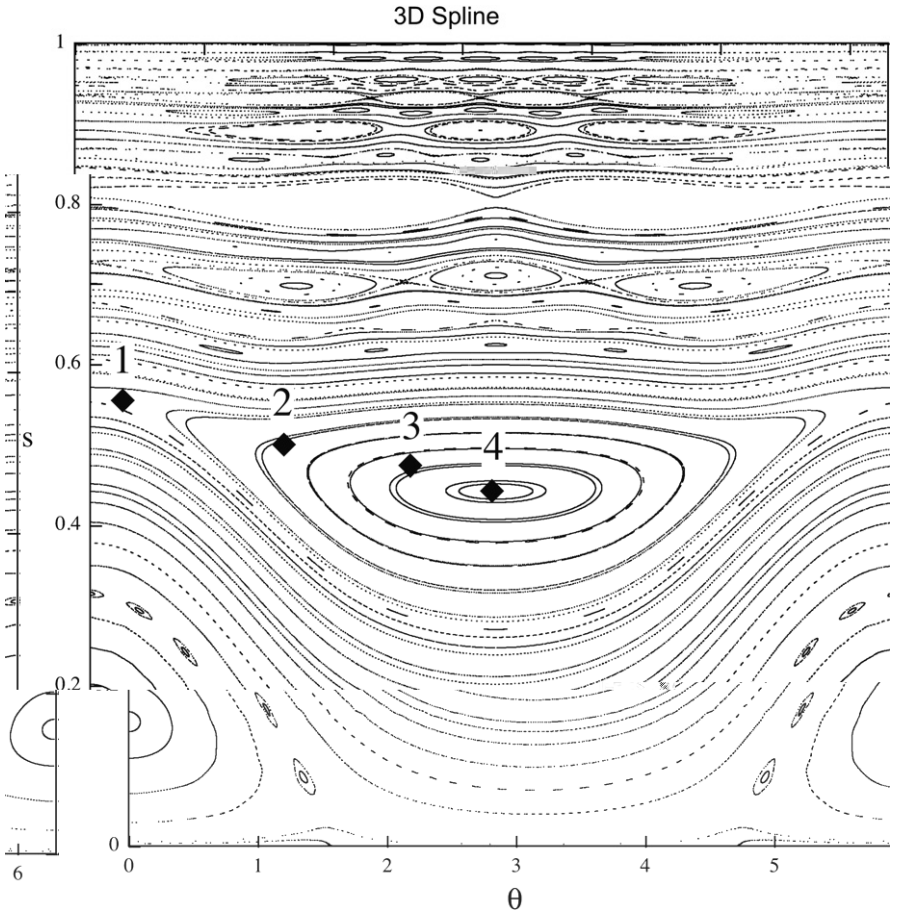

Fig. 5a. Poincaré plots obtained for the same field data used in Fig 4a, but now displayed in flux coordinate space (s–θ). The starting points of four selected orbits – 2 near the separatrix of the $m = 1$ island and 2 near the elliptical point of the island – are marked in each of the plots. Full rank data is plotted.

Here, the dot over a coordinate indicates the total time derivative along the particle orbit. Note that for $X \in (R, Z)$, $X_\alpha \equiv \partial X/\partial\alpha$ can be computed in action-angle flux coordinates from the inverse-mapping in Eq. (15). Since φ in Eq. (18) is the true cylindrical toroidal angle coordinate, $\varphi_s = \varphi_\theta = 0$. Inverting Eq. (18) yields the following flux coordinate time evolution equations:

$$\begin{aligned} \dot{s} &= \frac{Z_\theta \mathbf{v}_R - R_\theta \mathbf{v}_Z}{\tau} \\ \dot{\theta} &= \frac{-Z_s \mathbf{v}_R + R_s \mathbf{v}_Z}{\tau} \end{aligned} \tag{19}$$

Here, $\tau \equiv \partial(R, Z)/\partial(s, \theta) = R_s Z_\theta - R_\theta Z_s > 0$ is the 2D Jacobian (for each toroidal plane) between the cylindrical and flux coordinates. The speeds appearing in Eq. (19) are:

$$\begin{aligned} \mathbf{v}_R &= \dot{R} - R_\varphi \dot{\varphi} \\ \mathbf{v}_Z &= \dot{Z} - Z_\varphi \dot{\varphi} \end{aligned} \tag{20}$$

Equations (19) and (20), together with the mapping Eq. (15), may be used to evolve the particle orbits in action-angle coordinates, since the time derivatives on the right of Eq. (20) may be expressed in terms of the SVD-compressed magnetic field data (which are also known in action-angle space).

The simplest application of these orbit equations is to follow electron orbits for which the electron drift normal to the surfaces can be ignored. In that case, Eq. (20) reduces to:

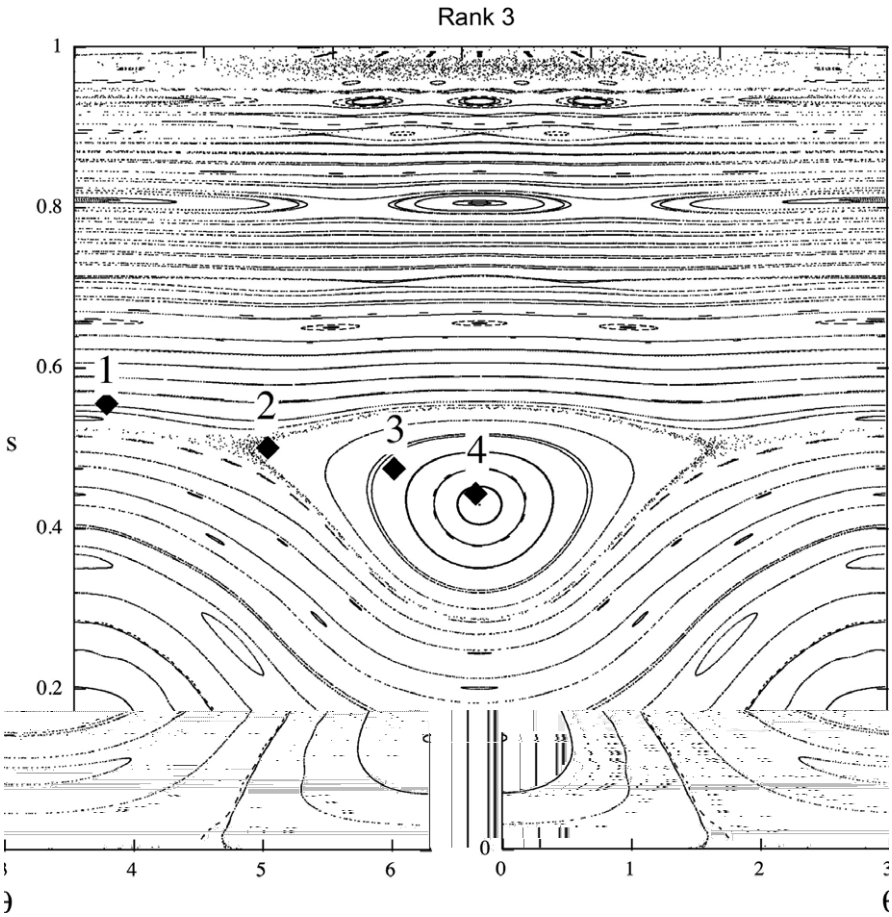

Fig. 5b. Same as Fig. 5a, but using rank-3 data.

$$\mathbf{v}_R = \mathbf{v}_\parallel \frac{RB_R - R_\varphi B_\varphi}{R|B|}$$
$$\mathbf{v}_Z = \mathbf{v}_\parallel \frac{RB_Z - Z_\varphi B_\varphi}{R|B|} \tag{21a}$$

with

$$R\dot{\varphi} = \mathbf{v}_\parallel \frac{B_\varphi}{|B|} \tag{21b}$$

Here, $|B| = B_R^2 + B_Z^2 + B_\varphi^2$ is the magnitude of the magnetic field and $\mathbf{v}_\parallel$ is the speed of the electrons along the magnetic field.

The expressions in Eqs. (19) and (21) can be integrated numerically to obtain the electron orbits in the *M3D*-generated magnetic fields (once the parallel speed $\mathbf{v}_\parallel$ is know). Significant departures from the lowest-order, nested equilibrium surfaces $s = \text{const.}$ will be indicative of the presence of resonant, magnetic-island producing perturbations in the magnetic data. These equations can be further simplified by dividing Eq. (19) by Eq. (21b) to eliminate the parallel speed. This is valid for transiting (untrapped) particles which are not reflected by the magnetic well and do not otherwise stagnate. Thus, the orbits trace the magnetic field lines.

The Poincaré maps for the electron orbits in the complex magnetic field structure under consideration are shown in Figs. 4a–4d for an *M3D* calculation with a large $m = 1$ magnetic island. One hundred orbits (field lines) distributed along the midplane horizontal x-axis were followed 1000 times around the torus to generate

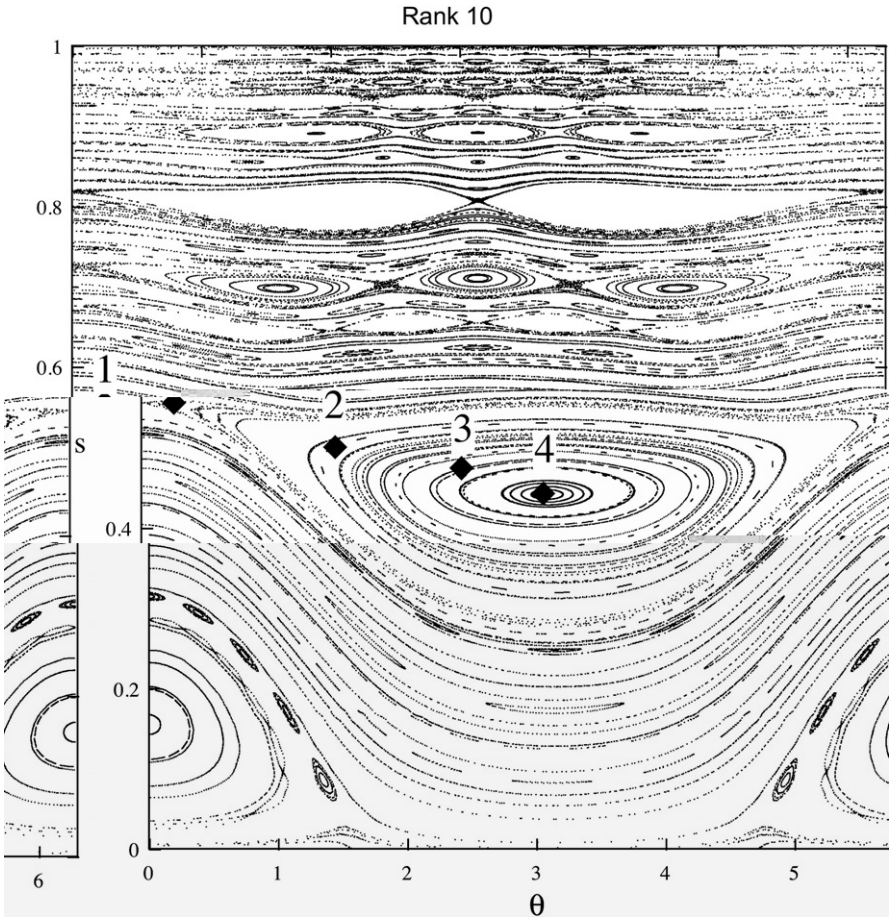

Fig. 5c. Same as Fig. 5a, but using rank-10 data.

these plots. The original magnetic data set was provided on a $235 \times 79 \times 49$ (s, θ, ϕ) grid. In Fig. 4a, the exact orbits based on uncompressed field data are displayed, where a 3D cubic spline was used for interpolation. Figs. 4b–4d show the field line puncture points (electron orbits) generated by varying the rank of the SVD used to approximate the three components of the magnetic field. Table 1 summarizes some of the features of the SVD rank variation. While the normalized RMS field error decreases rapidly up to about rank-20, so does the compression ratio. At rank-20, the compression ratio is still significant (35) while retaining the accuracy needed to closely approximate the original data. Above rank-20, the compression ratio diminishes faster than the RMS field error (which plateaus at about this rank), so rank-20 represents a good compromise value between accuracy and compression for this case.

The Poincaré orbit maps shown in Fig. 4 provide convincing evidence that the 3D MHD field data can be significantly compressed while still maintaining accuracy. At the rank-20 approximation, the energy norm is considered small enough so the resonant magnetic perturbations – which produce islands at rational surfaces – should be less than 1% of the minor radius. A more detailed measure of the accuracy of the compressed data can be obtained by comparing the particle *dynamics* for a few selected orbits in the vicinity of a magnetic island. Fig. 5 shows the Poincaré plots for the field data used in constructing Fig. 4, but now in the flux-coordinate (s–θ) space. The four solid dots mark the starting points for four orbits which are evolved according to the mapping in Eq. (19). They represent points near the separatrix (1–2) and inside (3–4) the $m = 1$ island of a bifurcated tokamak equilibrium.

Fig. 6 shows the particle orbits vs. ϕ (the toroidal angle) for these four points and for various ranks (3, 10, 20, and full). The orbits contain high-frequency variations superimposed on a slower periodic modulation.

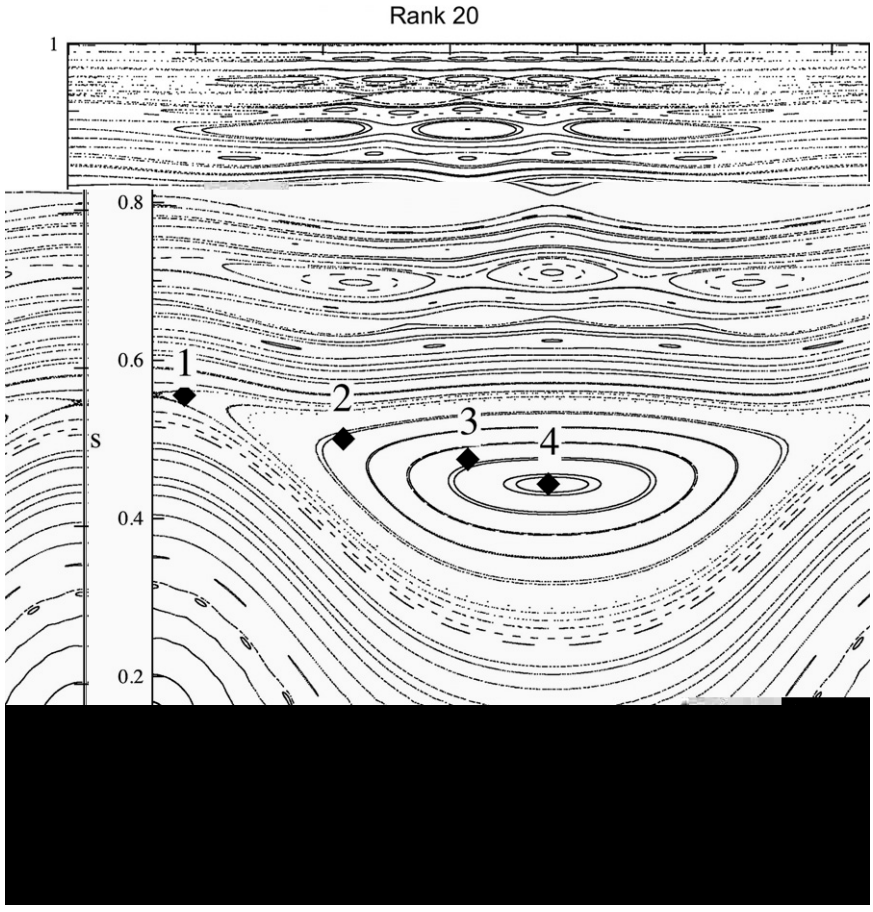

Fig. 5d. Same as Fig. 5a, but using rank-20 data.

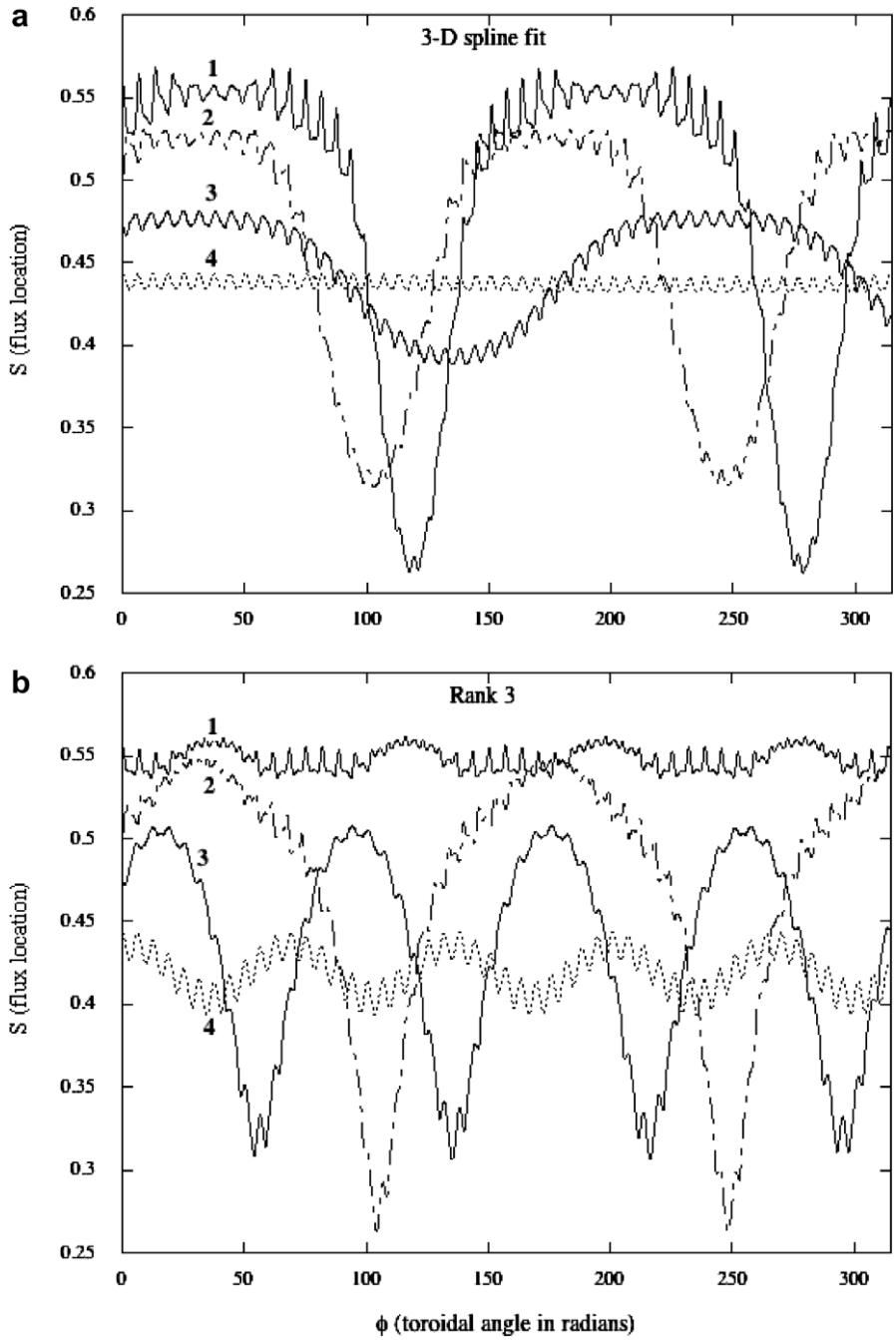

Fig. 6a–b. Radial (s) orbit traces for the four starting points marked in Fig. 5 plotted vs. the toroidal angle ϕ for full rank (spline) and low rank-3 field data.

The high frequency period is approximately 2π in the toroidal angle and is related to the radial orbit excursion as it traverses the island during each toroidal circuit. The lower frequency modulation is due to the slow motion around the island which occurs over many (10–20) toroidal transits. The differences in orbits between the various field ranks can be attributed to changes in the island structure and the resulting phase changes between the orbit and the island. For example, the orbit labeled "1" in Fig. 5 lies just *within* the separatrix of the $m = 1$ island for the full rank (3D spline), as well as the rank 10 and 20 approximations. For the

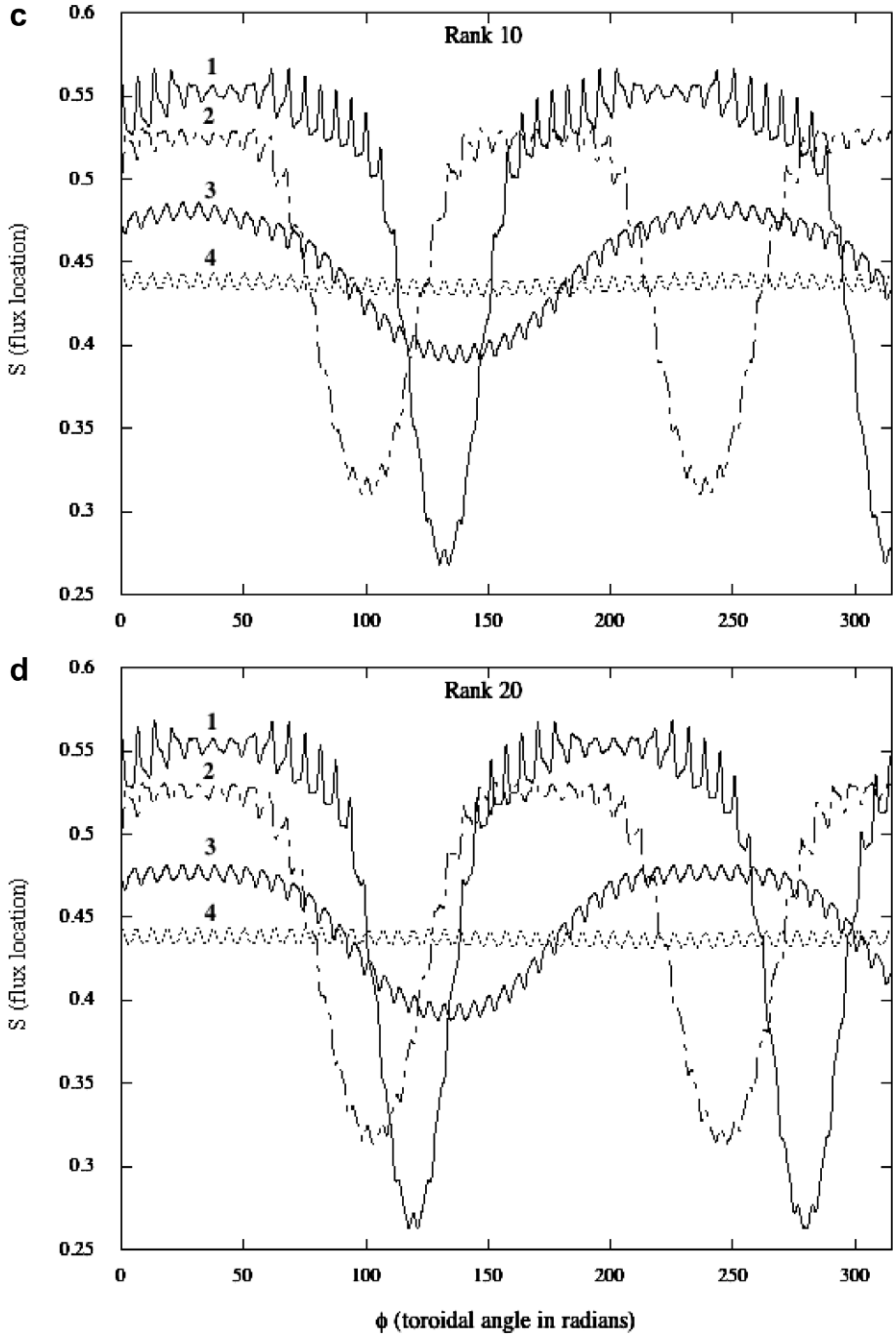

Fig. 6c–d. Radial (s) orbit traces for the four starting points marked in Fig. 5 plotted vs. the toroidal angle ϕ for rank-10 and rank-20 field data.

rank-3 fields, however, that orbit is just *outside* the separatrix. As a result, orbit "1" for rank-3 is confined to a much narrower range of s values than the same orbit for the more accurate field models. As can be seen, all the orbits shown for the rank-20 SVD approximation are in good agreement with the full rank model (there are very minor differences occurring at isolated points such as near the minima of orbit "1"). Being close to the separatrix, the amplitude of orbit 1 gives a good estimate of the island width, which as the plots show is well approximated even at rank-10.

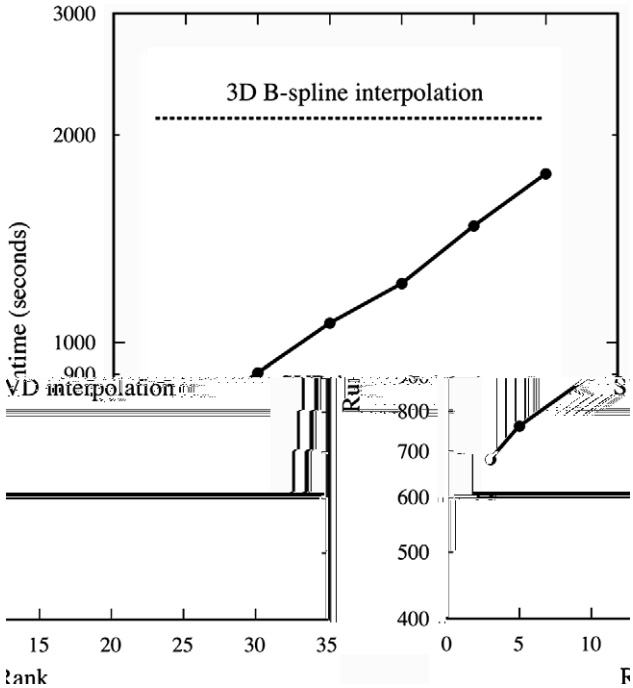

Fig. 7. Runtime comparison, vs. SVD rank, for the compressed SVD data used in Fig. 4, for a single-processor Opteron chip.

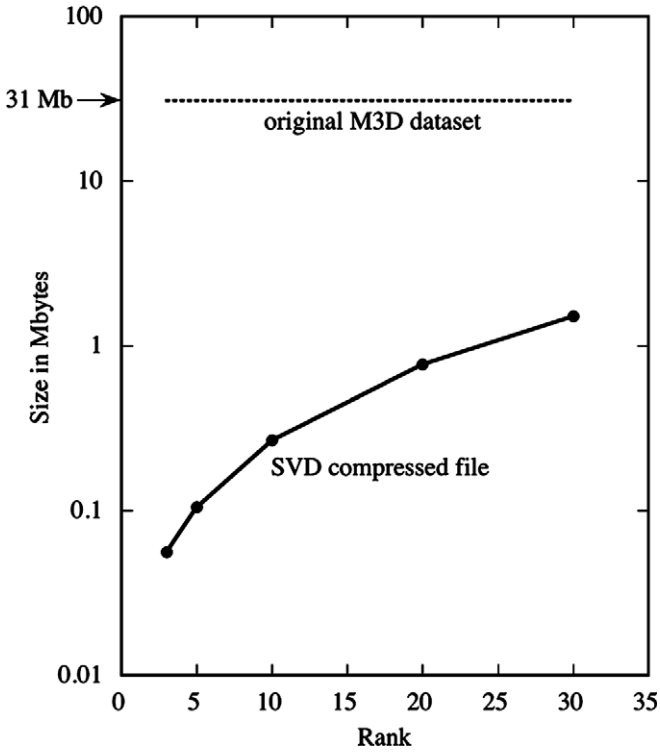

Fig. 8. File size vs. rank of the compressed data, compared with the original data set of $235 \times 79 \times 49$ points.

We have tested the computational efficiency of the SVD compression/interpolation technique in single processor mode on a Cray XT3 computer (the Jaguar computer at the National Center for Computational Sciences at Oak Ridge National Laboratory). This system is a 2.4-GHz AMD Opteron cluster (each processor has 2GB of memory). Although the serial performance tested here will not precisely reflect performance impacts in a multiprocessor hybrid fluid-particle code, it nevertheless provides an indication of the computational attractiveness of the present SVD compression scheme. Equal ranks $r_1 = r_2$ were used in the SVD-GLRA compression method. For the data used in Figs. 4 and 5, these were varied from values of 3 to 30 For comparison, the same orbit calculation was done using a 3D B-spline fit to the original *M3D* magnetic field data. The results are shown in Fig. 7. The timings for the two methods approach similar values at about a rank of 35. However, as shown in Figs. 4a–4d, good approximations for the magnetic island structures were obtained using ranks in the range of 15–20. For these ranks, the SVD-based runs were about a factor of two faster than the 3D B-spline based case. While the reduced computational time is important, the much smaller memory footprint ($\sim$600 kb vs. 75 Mb) for the SVD data (see Fig. 8) makes this method of significant interest for transferring data between macroscopic fluid and kinetic particle codes. The compactness of the SVD data set will lead to more rapid communication among processors as well as potentially allowing most of the particle code and its data to fit into cache memory. Furthermore, the truncation of the SVD spectrum provides automatic smoothing of the discretized field data provided by the MHD code.

5. Conclusions

SVD techniques have been used to achieve significant compression ratios, while preserving accuracy, for complex, three-dimensional magnetic data arising from the MHD simulation code *M3D*[1]. For a rank-$r \sim 20$ approximation, compression factors of 35 corresponding to a residual magnetic energy $\sim 10^{-8}$ are obtained. One could consider other compression methods, such as truncated spectral representations. However, as discussed before, by construction the SVD expansion is optimal (in terms of minimizing the Frobenius norm) over all possible tensor product representations and will therefore exhibit the best possible convergence properties. The resulting compressed magnetic data set was used to accurately integrate electron orbits for a thousand transits around the toroidal plasma in the presence of magnetic islands and regions of stochasticity. The reduction in the size of the data set due to compression is expected to facilitate rapid data exchange between multi-processors in future simulations of high temperature plasmas using Monte-Carlo particle methods to compute closure relations for fluid MHD equations. In the future, the reduced description of MHD fields available from SVD compression may be useful in developing low dimensional models of such important physics processes as magnetic reconnection. SVD compression (together with noise filtering, which has not been discussed here) of multi-dimensional drift orbit data resulting from Monte-Carlo particle codes may also be possible.

Acknowledgments

The authors are grateful to Dr. J. Breslau and the M3D team for providing access to M3D data that formed the basis for the analysis described in this paper. The thoughtful suggestions of Dr. R. Sanchez are appreciated. Finally, we are grateful to the referees for providing many constructive comments that have been incorporated into this paper.

Research sponsored by the Laboratory Directed Research and Development Program of Oak Ridge National Laboratory (ORNL), managed by UT-Battelle, LLC for the US Department of Energy under Contract No. DE-AC05-00OR22725. This research used resources of the National Center for Computational Sciences at Oak Ridge National Laboratory, which is supported by the Office of Science of the US Department of Energy under Contract No. DE-AC05-00OR22725.

Appendix A

In this appendix we briefly describe a couple of methods that can be used to continuously "extend" data from an irregular domain to a circumscribing rectangle, thus allowing SVD to be applied.

An extension of data into the surrounding "vacuum" region which preserves function values and first derivatives can significantly improve the resulting SVD compression ratio. This extension can be done by constructing a ghost scalar field A_g in the vacuum region and applying SVD compression to the extended field $(A + A_g)$. We now describe two techniques to compute the ghost field.

The first method creates a non-uniform mesh, or coordinate system, in the vacuum region by "expanding" the plasma boundary along suitably-defined radiating lines. Given a parametric representation of the plasma boundary as a curve $[x_0(\tau), y_0(\tau)]$ in the x–y plane, we construct a family of N nested curves $[x_k(\tau), y_k(\tau)]$, $k = 1, \ldots, N$ by expanding and rectifying the plasma boundary curve. These curves are uniquely defined by the parameterization of the boundary curve ($k = 0$). Based on these nested curves an $N \times M$ grid is constructed by discretizing the τ parameter, $\tau_1, \ldots, \tau_M$. The grid elements are defined by the mapping $(\tau_j, \gamma_k) \Rightarrow [x_k(\tau_j), y_k(\tau_j)]$ where γ_k denotes the coordinate *normal* to the expanded boundary curve at level k. Using this grid, the ghost field is computed by extrapolating the field values $A_{kj} \equiv A[x_k(\tau_j), y_k(\tau_j)]$ from the previous k surface along the $\tau = \text{const.}$ coordinate lines using a first order Taylor expansion:

$$A_{k,j} = A_{k-1,j} + (\partial A/\partial\gamma)_0[\gamma_k(\tau_j) - \gamma_{k-1}(\tau_j)] \tag{A.1}$$

The directional derivative in Eq. (A.1) is computed from the relation $\partial A/\partial\gamma \equiv [A(\gamma_{j+1}) - A(\gamma_j)]/\Delta\gamma_j$ Here the approximation $(\partial A/\partial\gamma)_k \approx (\partial A/\partial\gamma)_0$ was made to make the function gradient constant (i.e., $\partial^2 A/\partial\gamma^2 = 0$ along the parametric curve $\tau = \text{const.}$) in the "vacuum" region. Fig. 3d shows the decay of the SVD spectrum using this extension algorithm for the scalar test data in Eq. (16).

Another way to compute continuous field values (and derivatives) in the vacuum region is to use discrete local 2D gradient information to extend the data from the data boundary towards the circumscribing rectangular. Using Taylor series expansions and starting at the boundary between the data region and the "vacuum", one sets up a local, low-order matrix equation to compute the values of the first "layer" of vacuum grid points (layers are numbered to increase away from the data region, into the vacuum). For example, fitting only function values and first derivatives leads to a local expansion of the form:

$$F(\Delta x, \Delta y) = A + B\Delta x + C\Delta y \tag{A.2}$$

In Eq. (A.2), $\Delta x = x - x_0$, $\Delta y = y - y_0$ are the deviations in x, y from the vacuum point (x_0, y_0) where $F(0, 0) \equiv A$ is to be determined. At least three points are chosen (in practice, about twice that number leads to a smoother extrapolation) in the data region where F is known and which are the closest to the vacuum point. An integer search over grid indices is used to rapidly determine these points. The resulting set of overdetermined equations is solved (either by SVD or least squares) to obtain the coefficients in Eq. (A.2) and in particular, the value of A. The method proceeds in this fashion for all the adjacents "vacuum" points. The newly-determined vacuum values are not treated as known until the next layer of the iteration process begins (otherwise the method would be dependent upon the choice of starting points). In this way, the vacuum is filled as these layers "grow" outward from the original known data points until the entire vacuum region grid has been assigned values.

References

[1] W. Park et al., Phys. Plasmas 6 (1999) 1796.
[2] A.H. Glasser, C.R. Sovinec, R.A. Nebel, T.A. Gianakon, S.J. Plimpton, M.S. Chu, D.D. Schnackand the NIMROD Team, The NIMROD Code: A New Approach to Numerical Plasma Physics, Plasma Phys. Control. Fusion 41 (1999) A747–A755.
[3] ITER Physics Basis, Nucl. Fusion 39 (1999) 2137.
[4] S. Ohdachi, K. Toi, G. Fuchs and LHD experimental Group, Analysis of two-dimensional fluctuations using singular value decomposition, in: Proceedings of the 30th EPS Conference on Controlled Fusion and Plasma Physics, St. Petersburg, Russia, 2003.
[5] N. Pomphrey, L. Berry, A. Boozer, et al., Nucl. Fusion 41 (2001) 339.
[6] Jieping Ye, Generalized low rank approximations of matrices, Mach. Learn. 61 (2005) 167.
[7] P. Holmes, J.L. Lumley, G. Berkooz, Turbulence, Coherent Structures, Dynamical Systems and Symmetry, Cambridge University Press, New York, USA, 1996.
[8] G.H. Golub, C.F. van Loan, Matrix Computations, third ed., The Johns Hopkins University Press, London, 1996.
[9] W.H. Press, B.P. Flannery, S.A. Teukolsky, W.T. Vetterling, Numerical Recipes: The Art of Scientific Computing, Cambridge University Press, Cambridge, 1986.
[10] L. de Lathauwer, B. de Moor, J. Vandewalle, SIAM J. Matrix Anal. Appl. 21 (4) (2000) 1253.

[11] D.A. Spong, S.P. Hirshman, L.A. Berry, et al., Nucl. Fusion 41 (2001) 711.
[12] S.P. Hirshman, W.I. van Rij, P. Merkel, Comput. Phys. Commun. 43 (1986) 143.
[13] R.H. Fowler, J.A. Rome, J.F. Lyon, Phys. Fluids 28 (1985) 338;
D.A. Spong et al., Nucl. Fusion 40 (563) (2000).