15B.2

THE "CHROMATIC HEXAD" METHOD FOR ADAPTIVE ANISOTROPIC COVARIANCE SYNTHESIS

R. James Purser*

National Centers for Environmental Prediction

1. INTRODUCTION

We shall describe a systematic method to efficiently generate a spatially adaptive anisotropic covariance operator on a computational lattice for data assimilation in three dimensions. The computed self-adjoint covariance operator is formed as the product of a (not necessarily symmetric) "square-root" factor and its adjoint. This square-root factor is generally taken to be a small sum of operators, each of which resembles the quasi-Gaussian convolution obtained as the outcome of a general diffusive process acting for a finite time. A superposition of this kind can produce desirable features, such as relatively fat tails to the final covariance at large separations, or negative side-lobes, which a single Gaussian term alone could not reproduce.

The key to achieving efficiency is to form each of the quasi-Gaussian terms by combining the "Hexad" geometrical decomposition principle (described in Purser et al. 2003b) with the method of spatial recursive filtering (Purser 2003a) along the line segments that the hexad algorithm indicates and with the degree of smoothing that the hexad method also specifies. The spatial dispersion of each quasi-Gaussian contribution is quantified by the "aspect" tensor, defined as twice the product of the tensorial diffusivity and the "time" over which such diffusion must act to produce that contribution; under homogeneous conditions, it is also the centered second moment of that quasi-Gaussian. The hexad principle exploits the additive property of the general 3D diffusion operator that enables any such operator to be linearly decomposed locally into a unique hexad of (i.e., six) rank-one contributions of one-dimensional diffusion along possibly oblique generalized lines of the lattice, where the configuration of these lines' directions are collectively bound by certain simple geometrical rules of validity. In the abstract linear space of the six aspect components, the procedure by which the given aspect tensor is resolved into such linear components can be interpreted geometrically as a linear projection. However, for the chosen hexad of six generalized grid lines to be valid for this aspect tensor, the projection must result in all six of the resolved components being non-negative, since each one determines a linear degree of smoothing along the respective line (and in this context, a *negative* degree of Gaussian smoothing is not meaningful). The rules of validity for a hexad of grid lines guarantee that only one hexad validly resolves a given generic aspect tensor, and the basic hexad algorithm, as briefly outlined in Purser et al. (2003b), defines a reliable and efficient method by which this hexad is determined. We may think of the hexad method as a way of organizing line-diffusion operations that, collectively, combine to produce the outcome of a three-dimensional diffusion associated with a generalized (tensorial) diffusivity. The diffusion method has been directly applied in analysis by Derber and Rosati (1989) and, with tensorial diffusity, by Weaver and Courtier (2001). The hexad method, in combination with the recursive filters, simply makes this style of covariance synthesis computationally efficient.

For inhomogeneous covariance models the hexad of lines will, in general, not be the same at all points of the computational lattice but will change in response to the smoothly changing aspect tensor. In the basic form of the hexad method, the change from one hexad to a neighboring one involves the resolved component along one of the six lines (the one not shared by the neighboring hexad) going continuously to zero at the transition. Unfortunately, experience has shown that this descent to zero in the line-smoothing coefficient tends to be rather abrupt, leading to unsightly numerical "dislocations" at the inter-hexad transition interfaces. This presentation will therefore discuss a recent refinement, the "blended hexad" method, which overcomes this defect at the modest expense of requiring, at each lattice point, a larger set (normally 13) of lines to participate in the local line-smoothing operations.

2. HEXAD AND BLENDED HEXAD METHODS

In a topologically simple computational lattice, meaning one whose points are all integer-coefficient

* SAIC, Beltsville, MD

Corresponding author address: R. James Purser, W/NP2 RM 207, WWBG, 5200 Auth Road, Camp Springs, MD 20746

3-vectors associated with the three standard basis vectors generating the lattice, we also can define *any* generalized grid line's orientation as an integer nonvanishing 3-vector, $\boldsymbol{g}$, and note that the same line is also defined by the oppositely directed generator, $-\boldsymbol{g}$. However, we shall restrict the definition of a "generator" to vectors whose three integer components do not share a common factor greater than unity. Expressed in standard grid units, the degenerate aspect tensor that results from a unit degree of diffusive smoothing along such a line is just $\boldsymbol{g}\boldsymbol{g}^T$. If we have six generators, $\boldsymbol{g}_i$, with $i \in \{1, \ldots, 6\}$, such that the six tensors $\boldsymbol{g}_i\boldsymbol{g}_i^T$ are linearly independent, then *any* symmetric tensor $\boldsymbol{A}$ can be uniquely resolved into the "basis" that this set of six special tensors provides:

$$\boldsymbol{A} = \sum_{i=1}^{3} w_i (\boldsymbol{g}_i \boldsymbol{g}_i^T). \tag{1}$$

Suppose that all six of these w_i are positive. If we apply a sequence of diffusive operations throughout the grid where, in step 1, we smooth along lines generated by $\boldsymbol{g}_1$ by an amount whose second moment (in these $\boldsymbol{g}_1$ grid units) is w_1, and so on for all the other five orientations with their corresponding smoothing weights, then, by the linearity of second moments under sequential smoothing operations, we shall end up with a composite smoothing operation whose second moment tensor is just the tensor, $\boldsymbol{A}$, that was resolved into convenient line-associated components by (1).

But at this point, we have no criteria by which we can choose the $\boldsymbol{g}_i$ that lead always to the happy circumstance that all the associated w_i are positive (or at least, non-negative). However, let us consider the set G of *all* the tensors of the form $\boldsymbol{g}\boldsymbol{g}^T$, where each $\boldsymbol{g}$ is a valid generator in the sense defined above. It is easy to show that each $\boldsymbol{g}\boldsymbol{g}^T$ lies on the convex hull of G. A careful analysis reveals that the boundary of the convex hull of G is a polyhedral shell whose "facets" are each a five-dimensional polytope having exactly six of the $\boldsymbol{g}$ as its vertices. In other words, each facet is a "6-simplex". The interior of each facet is an aspect tensor describable as a convex mixture (weights summing to unity) of the degenerate aspect tensors forming the six vertices, but *any* valid aspect tensor whatsoever is a positive multiple of such a convex mixture (since the convex polyhedral shell in question spans the entire "cone" of valid aspect tensors). By an appropriate linear transformation of the original lattice into itself, any facet of the boundary of the convex hull of G can be mapped into any other, so the analysis of the configuration of any one "hexad", $\boldsymbol{g}_i$, associated with an arbitrary facet will suffice to describe *all* of the valid hexads in this natural convention for carrying out aspect tensor decompositions.

It emerges that every hexad of generators defined this way, and their negatives, can be identified with the 12 vertices of an affinely transformed "cuboctahedron" of the integer grid and is such that, by choosing the labels and orientations of these generators according to the freedom that the symmetry of the situation allows us, we can take the first three generators to form a matrix with unit determinant, i.e:

$$\det\{\boldsymbol{g}_1; \boldsymbol{g}_2; \boldsymbol{g}_3\} = +1, \tag{2}$$

while the three remaining generators obey the 3-cyclic relations:

$$\boldsymbol{g}_4 = \boldsymbol{g}_3 - \boldsymbol{g}_2, \tag{3a}$$

$$\boldsymbol{g}_5 = \boldsymbol{g}_1 - \boldsymbol{g}_3, \tag{3b}$$

$$\boldsymbol{g}_6 = \boldsymbol{g}_2 - \boldsymbol{g}_1. \tag{3c}$$

This puts the first three generators at the corners of a triangular facet of the cuboctahedron and puts $\boldsymbol{g}_1$ and $\boldsymbol{g}_4$ at opposite corners of a quadrilateral facet, with pairings likewise for $\boldsymbol{g}_2$ and $\boldsymbol{g}_5$ and for $\boldsymbol{g}_3$ and $\boldsymbol{g}_6$. [Note, this convention differs from that used in the description of Purser et al. (2003b), because the present convention has been found to be more convenient in practice.]

Finding the correct hexad of generators for a given aspect tensor $\boldsymbol{A}$ involves an iterative process which, formally, resembles the classical "Simplex" algorithm (e.g., Dantzig, 1963). At an intermediate point in the iterations we have a trial hexad onto which the tensor $\boldsymbol{A}$ is resolved to give the corresponding trial weights w_i. However, some of these weights might be negative, implying that the iteration is not yet complete. One offending negative-weight direction is discarded. The cuboctahedron possesses an opposing pair of quadrilateral facets that *do not* contain this offending direction's generator (or its negative); the replacement generator is found by multiplying by two the centroid of this particular quadrilateral. It is easily shown that the resulting set of six generators (five old, one new) and their negatives do indeed define the corners of a new grid-embedded cuboctahedron, and it is not difficult to find a suitable labelling of this new set that satisfies the conventions set out in (2) and (3). Of course, the aspect-tensor-space 6-simplexes associated with the unit-weight images of this old and new hexad are just adjacent facets of the convex polyhedral shell we described above. The iterations proceed until *all* weights w_i are non-negative, whereupon the correct hexad is found at last.

The blended hexad method differs from the decomposition described above by incorporating an implicit smoothing of the weights w_i that is carried out in the linear space of aspect tensors

in the following way. Instead of resolving each aspect tensor as the single point in aspect space projected onto the six basis tensors associated with the unique hexad that contains the aspect tensor point in question, we replace the single point in aspect space as if by a symmetrical "cloud" centered on the original point, but now considered to carry the "weight" of the original point distribution in a symmetrically smeared-out way. Technically, we define a smoothing kernel in aspect space which, in a well-defined sense, is approximately "radially symmetric" about the original point. The existence of a natural intrinsic metric, defined:

$$||\boldsymbol{A}_1, \boldsymbol{A}_2||^2 = \frac{1}{2}\text{trace}\{[\log(\boldsymbol{A}_1^{-1}\boldsymbol{A}_2)]^2\} \qquad (4)$$

between any pair of aspect tensors makes the concept of "radially symmetric" meaningful in this six-dimensional space for kernels of infinitesimal radius (although at finite radii, the intrinsic curvature of the space implied by the Riemannian metric (4) would invalidate the concept in the strict sense). The kernel is defined to have its support wholly within a ball of sufficiently small radius that the distribution of the implied cloud of points intersects only a limited number of neighboring hexads. For a kernel of a given finite radius, however small, it is always possible to find locations of aspect space where this kernel will overlap 16 hexads, that being the maximum number of hexads that meet at the generic (i.e., nonsingular) aspect tensor points. At such worst-case overlaps the total pool of generators $\boldsymbol{g}$ involved can be shown to number exactly 13. Therefore, it is theoretically possible to resolve each point of the smoothing kernel's "cloud", with its proper weight, into a hexad taken from this pool of 13 and, by superposition, integrate over the cloud to blend all the contributing hexads into an equivalent 13-line smoothing configuration that is guaranteed to reproduce the intended smoother, but with weights w_i that fade more smoothly to zero than in the basic hexad method.

Remarkably, the geometry of the hexads in relation to one another admits sufficient symmetry to make it quite feasible, for appropriate choices of the smoothing kernel radial profiles (essentially they must be generalized "Beta functions"), to carry out the implied integrations for a tabulated array of points in aspect space that cover a representative hexad's image in this space. The ability to exploit the symmetry by which one hexad is transformed into any other through a lattice isometry means that, given any aspect tensor, one may "look up" in the prepared table the set of 13 active grid generators together with their associated smoothing weights and, by sequential applications of the line-diffusion operators applied in these 13 orientations, synthesize the smoothing effect producing the desired tensorial dispersion. The blended hexad method can be made virtually free of the undesirable numerical artifacts that mar the basic hexad method applied to inhomogeneous smoothing problems.

3. AN APPLICATION OF GALOIS FIELDS

Whether we choose to resolve the gridded field of aspect tensors using the basic hexad method (six directions at each point) or the blended version (13 directions at each point), it is important that the line smoothing operations at different locations are coordinated so that there is not a misguided attempt to filter concurrently two lines that intersect. This is obviously especially relevant in a parallel processing computing environment, since we want to be able to distribute the computing load equitably and apply several line-smoothing filters simultaneously.

Fortunately, a connection between formal geometrical characterizations of the hexad manipulations and some aspects of the mathematical theory of "Galois Fields" (for example, Dickson, 1958; Hirschfeld 1998) has been recently identified to provide a rigorous basis for a kind of "color coding" of all possible orientations of oblique lines in the computational lattice. By performing smoothing operations simultaneously only when they all belong to the *same* color assignment, we can guarantee that no conflicts can arise.

In the basic hexad method, a seven-color assignment associated with the Galois field "GF(8)" ensures that no valid hexad contains the same color of line twice. In the blended hexad, a coding of 13 colors associated with the Galois field "GF(27)" will suffice to ensure that no lines of the sequence of 13 operations specified by the blended hexad algorithm can contain the same color twice.

A "Field" in the mathematical sense is a set of quantities obeying certain algebraic rules that include generalizations of the concepts of addition and multiplication. In the Galois fields, the set of quantities is finite and of a number of the form p^n where p is a prime and n any positive integer. For our purposes, it is mainly the additive structure of the Galois fields that is of relevance, since there is a natural mapping from the points of an n-dimensional lattice to the elements of each GF(p^n) obtained by arranging repeated copies of the elements of the GF in n-dimensional hypercubes of linear dimension p so that vector summation on the lattice, modulo-p in each of the n dimensions, reproduces the additive structure of the Galois field. For GF(8), this means that 2*2*2 cubes of the eight field elements are repeated indefinitely in all three cartesian directions of the lattice with the additive identity of the GF(8) being at the lattice

origin. From this origin, all the grid points that are "visible", which is equivalent to all the valid grid generators, is then possessed of an image ("color") in GF(8) that is *not* the origin. This establishes a coloring from a palette of seven. A similar construction with the elements of GF(3^3) makes the 26 elements that are not the additive identity of this field visible from the lattice origin, but in such a way that the mutually opposing generators thereby identified with two field elements will always link the *same* 13 pairs, that we can therefore identify with those generators' "colors" in this scheme.

The special value in these identifications stems from the fact that any valid hexad whatsoever always has its six opposing pairs of vertices of six *different* colors (in both the GF(8) and GF(27) schemes) and, more remarkably, the 13 lines of the blended hexad set corresponding to any aspect tensor are always associated, through the GF(27) color identification, with each one of the 13 colors of this palette.

4. GENERALIZATIONS

It is clear that the application of the ideas of convexity employed in section 2 will carry over into the generic n dimensional lattice case. In two dimensions, the "Triad" algorithm is a simple restriction of the Hexad method and involves a cone of three-dimensional aspect points that are always resolvable into three line-smoothing operations. In four dimensions the convex shell formed as the boundary of the convex hull of the set analogous to the G of section 2 is tiled not with a single repeated simplex shape, but a mixture of two different polytopes. One is a simplex, and therefore possesses 10 vertices, but the other is a 12-pointed object (whose associated polytope in the four-dimensional lattice is an affinely transformed "24-cell", a particularly symmetrical regular polytope with some remarkable properties). Thus, in four dimensions, we do not strictly obtain a simple "Decad" algorithm for an aspect tensor that belongs to the 12-pointed configuration. The difficulty is partly solved by carving the 12-pointed regions symmetrically into "pie-slices", which *does* lead to a decomposition into 10-simplexes, as required. However, the artificially-added points ("pie-centers") in this decomposition, and the dictates of symmetry, imply that even the basic (unblended) algorithm will generally involve 12 smoothing directions, not merely 10, in order to synthesize the general anisotropic Gaussian in any regular four-dimensional lattice.

The algorithm for this decomposition has also been written and tested and could find future applications in synthesizing the covariances of the model error terms in a weak-constraint four-dimensional variational assimilation scheme.

REFERENCES

Dantzig, G. B.. 1963: *Linear Programming and Extensions*, Princeton, New Jersey. 648 pp.

Derber, J. C., and A. Rosati, 1989: A global ocean data assimilation system.*J. Phys. Ocean.,* **19,** 1333–1347.

Dickson, S., 1958: *Linear Groups (with an Exposition of Galois Field Theory)* Dover.

Hirschfeld, J. W. P., 1998: *Projective Geometries over Finite Fields (Second Ed.)* Oxford University Press.

Purser, R. J., W.-S. Wu, D. F. Parrish, and N. M. Roberts, 2003a: Numerical aspects of the application of recursive filters to variational statistical analysis. Part I: Spatially homogeneous and isotropic Gaussian covariances. *Mon. Wea. Rev.,* **131,** 1524–1535.

——, W.-S. Wu, D. F. Parrish, and N. M. Roberts, 2003b: Numerical aspects of the application of recursive filters to variational statistical analysis. Part II: Spatially inhomogeneous and anisotropic general covariances. *Mon. Wea. Rev.,* **131,** 1536–1548.

Weaver, A., and P. Courtier, 2001: Correlation modelling on the sphere using a generalized diffusion equation. *Quart. J. Roy. Meteor. Soc.,* **127,** 1815–1846.