Fast $k$-Nearest-Neighbors Calculation for Interpolation of Radar Reflectivity Field*

FENG GAO

Institute of Urban Meteorology, Beijing, China

(Manuscript received 26 September 2008, in final form 12 January 2009)

ABSTRACT

To efficiently implement the interpolation methods (e.g., Shepard’s method and its variants) for the radar reflectivity field, a fast method that calculates the $k$-nearest-neighbor nodes (sampling points in radar volume scan) of the interpolated point (grid point) is described and proved. Several geometric propositions of radar volume scan on which the method is based are suggested and proved. Finally, the computational cost of the method is analyzed. The method is fast enough for real-time applications.

1. Introduction

The radar reflectivity fields are usually interpolated to grids for producing radar mosaic or for those processes that are more easily carried out for gridded data (Dixon 2005).

Shepard’s method (Shepard 1968) and its variants (Lodha and Franke 1994) are the inverse distance–weighted interpolation methods. They are suitable interpolation methods for scattered samplings and can be easily extended to arbitrary dimensions. They can be used to interpolate the radar reflectivity fields to grids.

The interpolant $F_j$ at grid point $G_j$ is defined to be the weighted average of samples $f_i$ at nodes (sampling points) $S_i, i = 1, \ldots, n$:

$$F_j = \sum_{i=1}^{n} w_i f_i,$$

where $w_i$ are weight functions,

$$w_i = \frac{\sigma_i}{\sum_{i=1}^{n} \sigma_i},$$

and

$$\sum_{i=0}^{n} w_i = 1,$$

where $\sigma_i$ is the inverse distance weight function (i.e., function of the distance from $G_j$ to $S_i$).

The interpolant is global, that is, all samples are used in calculation for the grid point, which means that 1) the amount of all samples may be too large to be computationally feasible, which is especially true for radar volume scan, and 2) the local variations may be smoothed away. Therefore, the interpolant is usually made local through multiplying $\sigma_i$ by a damping function $\lambda_i$ so that $\sigma_i$ will vanish outside some neighborhood of $G_j$ (Lodha and Franke 1994). In this way, only some $k$-nearest-neighbor nodes of $G_j$ are used in the calculation.

This paper describes a fast method, which takes advantage of the geometric properties of the radar volume scan, to find the $k$-nearest nodes of each grid point. It does the calculation on a small neighborhood of the grid point and picks up just $k$-nearest nodes.

2. Geometric foundations

As shown in Fig. 1, a radar beam with $n$ nodes starts at $S_1$, and goes to $S_n$. The grid point to be interpolated is $G$. The azimuth angles of the radar beams $S_1S_n$ and $G$ are $\phi_1$ and $\phi_n$, respectively, and $\theta_1$ and $\theta_n$ are elevation angles of $S_1S_n$ and $G$, respectively. The distance from $G$ to $S_i$ is $|G - S_i|$, where $1 \leq i \leq n$. Point $P$ on $S_1S_n$ is such that $GP$ is perpendicular to $S_1S_n$. In other words, $|G - P|$ is the distance from $G$ to $S_iS_n$.

Some assumptions are made about the geometric structure of the radar volume scan: 1) the radar beams on the same sweep are of the same elevation angle, 2)
the nodes on radar beams are uniform (i.e., the gate width is constant), and 3) the grid points are all inside the radar volume scan (i.e., no extrapolation is allowed).

\textit{a. Proposition 1}

As shown in Fig. 2, suppose that \( S_N \) is the nearest node on the beam to the grid point \( G \):

- if \( P \) is to the left of \( S_N \) or \( |P - S_1| \leq |S_N - S_1| \), then
  \[ |G - S_N| \leq |G - S_{N-1}| \leq |G - S_{N+1}| \leq \ldots, \]
  \hspace{1cm} (1)
- if \( P \) is to the right of \( S_N \) or \( |P - S_1| \geq |S_N - S_1| \), then
  \[ |G - S_N| \leq |G - S_{N-1}| \leq |G - S_{N+1}| \leq \ldots; \]
  \hspace{1cm} (2)

the equalities hold when \( P \) is on \( S_N \).

\textbf{PROOF}

It can be easily proved by using Pythagoras theorem. QED

Since \( P \) is on \( S_1S_n \) or the line extended from both \( S_1 \) and \( S_n \), it can be described parametrically by

\[ P = S_1 + t(S_n - S_1), \]

where \( t \) is the parameter and (Weisstein 1999)

\[ t = \frac{(S_n - S_1) \cdot (G - S_1)}{|S_n - S_1|^2}. \] (3)

From parameter \( t \), one can find out which node is nearest to \( P \) and thus nearest to \( G \). The nearest node is \( S_N \) with

\[ N = \begin{cases} 
  [m + 0.5] + 1, & \text{if } \lfloor m + 0.5 \rfloor + 1 \geq 1, \\
  1, & \text{otherwise},
\end{cases} \] (4)

where \( [x] \) is the greatest integer no larger than \( x \). Note that \( \lfloor m + 0.5 \rfloor + 1 \) cannot be greater than \( n \) since \( G \) is in radar volume scan.

Once the nearest node on the beam is calculated by Eqs. (3) and (4), the next nearest node on the beam can be calculated by Eqs. (1) or (2), and so on. In cases when \( i \leq 1 \) or \( i \equiv n \), \( S_i \) is skipped.

\textit{b. Proposition 2}

As shown in Fig. 3, suppose that \( S_N \) and \( S_{N'} \) are the nearest nodes to \( G \) on \( S_1S_n \) and \( S'_1S'_n \), respectively.

If \( \alpha \leq \alpha' \), then \( |G - S_N| \leq |G - S_{N'}|. \)

\textbf{PROOF}

First, suppose both \( P \) and \( P' \) are on their corresponding beams, or \( 0 \leq t, t' \leq 1 \). Since \( \alpha \leq \alpha' \), \( N' \leq N \). And since \( |S_{N'}| = |S_N|, |G - S_{N'}| \leq |G - S_N| \).

If \( N' = N \), then \( |G - S_N| = |G - S_{N'}| \leq |G - S_{N'}| \).

Otherwise, if \( N' < N \), since \( |P - S_N| \leq G/2 \) (is the gate width), \( |P - S_{N'}| \leq |G - S_{N'}| \).

Therefore, \( |G - S_N| \leq |G - S_{N'}| \).

Next, suppose \( P' \) is not on \( S'_1S'_n \), or \( 0 \leq t \leq 1 \) and \( t' < 0 \).

Then \( S_1' \) is the nearest node on \( S'_1S'_n \), and \( N' = 1 \).

Thus \( N' \leq N \), and \( |G - S_N| \leq |G - S_{N'}| \) by repeating the above reasoning.

Finally, suppose both \( P \) and \( P' \) are not on their corresponding beams, or \( t, t' < 0 \). Then \( N = N' = 1 \), and \( |G - S_N| \leq |G - S_{N'}| \) by repeating the above reasoning. QED

The beam is defined to be no farther than another beam if its nearest node is no farther to the grid point than another beam’s nearest node.

\textit{c. Proposition 3}

As shown in Fig. 4, \( S_1S_n \) and \( S'_1S'_n \) are on the same sweep with an elevation angle of \( \theta \). If \( |\phi_k - \phi| \leq |\phi_k - \phi'|, \)

\( S_1S_n \) is no farther than \( S'_1S'_n \).

\textbf{PROOF}

The far end nodes of \( S_1S_n \) and \( S'_1S'_n \) are

\( 1 \) The angles are from \( G \) to \( S_1S_n \) and from \( G \) to \( S_1'S_n \), and \( 0 \leq \alpha, \alpha' \leq \pi \).

\( 2 \) The situation where \( t > 1 \) will never happen, since only grid points in radar volume scan are considered.

\( 3 \) The azimuth angle differences are directed, from (the projection of \( G \) to (the projection of \( S_1S_n \) and from (the projection of \( G \) to (the projection of \( S_1'S_n \), \( -\pi \leq \phi_k - \phi, \phi_k - \phi' \leq \pi \). They are positive if the directions are clockwise and negative otherwise.
Let and similarly, 

\[ S_n = (|S_n| \cos \theta \cos \phi, |S_n| \cos \theta \sin \phi, |S_n| \sin \theta) \]

and 

\[ S'_n = (|S'_n| \cos \theta \cos \phi', |S'_n| \cos \theta \sin \phi', |S'_n| \sin \theta), \]

respectively. The grid point is 

\[ G = (|G| \cos \theta \cos \phi, |G| \cos \theta \sin \phi, |G| \sin \theta). \]

Let \( \delta = \phi - \phi \) and \( \delta' = \phi - \phi' \), then \( \phi = \phi - \delta \) and \( \phi' = \phi - \delta' \). Then,

\[ \cos \alpha = \frac{S_n \cdot G}{|S_n||G|} = \cos \theta \cos \phi \cos \delta + \sin \theta \sin \phi \sin \delta \]

and similarly,

\[ \cos \alpha' = \frac{S'_n \cdot G}{|S'_n||G|} = \cos \theta \cos \phi' \cos \delta' + \sin \theta \sin \phi' \sin \delta'. \]

Suppose \(|\delta| \leq |\delta'|\), then \( \cos \alpha \geq \cos \alpha' \); thus, \( \alpha \leq \alpha' \) since \( 0 \leq \alpha, \alpha' \leq \pi \). The proposition is proved by proposition 2. QED

For a grid point, its azimuth angle can be used to find the nearest beams on each sweep.

### 3. Algorithms

The algorithms used to calculate the nearest-neighbor nodes of each grid point are based on the propositions in the last section.

Each time when asked, the next nearest node in volume to the grid point is calculated and returned. The heap data structures (Kruse and Ryba 1998) are used to keep the beams or nodes sorted on their corresponding heaps by the distances of nearest nodes on beams to the grid point or the distances of nodes to the grid point, respectively.

Algorithm 1 does the necessary initialization. Algorithm 2 returns the next nearest beam in volume. Algorithm 3 returns the next nearest-neighbor node in volume. Algorithm 4 calculates, for the grid point, \( k \)-nearest-neighbor nodes in volume.

A simple illustration of the algorithms is shown in Fig. 5. In the figure, the nodes on each beam are sorted by distance to the grid point (i.e., \( S_1 \) is no farther than \( S_{21}, S_{22} \) is no farther than \( S_{31}, \) and so on). The solid dots indicate that the corresponding nodes are on the node heap. The hollow dots indicate the corresponding nodes are not on the node heap—either already popped from the heap or not yet pushed to the heap.

Initially, there is only one node on the heap. It is the nearest node on the nearest beam, as shown by (1) in Fig. 5.

During the first pass of algorithm 3, since the beam is the only one with a node on the heap, it is also the farthest beam with a node on the heap. Therefore, the nearest node on the next nearest beam in volume is pushed on the heap, as shown by (2) in Fig. 5. The \( S_{11} \) was popped and is returned as the next nearest node in volume (the nearest node in volume at this time).

During the second pass of algorithm 3, if \( S_{12} \) is no farther than \( S_{21} \), it is popped and will be returned as the next nearest node in volume. Since the beam, which \( S_{12} \) is located on, is not the farthest of those beams with nodes on the heap, lines 5, 6, and 7 of algorithm 3 are skipped and \( S_{12} \) is returned as the next nearest node in volume, as shown by (3a) in Fig. 5. Otherwise, \( S_{21} \) is
popped from the heap. Since the beam, which $S_{21}$ is located on, is the farthest of those beams with nodes on the heap, the nearest node $S_{31}$ on the next nearest beam in volume is pushed on the heap, and $S_{21}$ is returned as the next nearest node in volume, as shown by (3b) in Fig. 5.

Note that for a beam, only one node of it may appear on the heap.

The following proof shows that the $k$-neighbor nodes calculated by algorithm 4 are actually the nearest-neighbor nodes of the grid point. In the proof, the heap is the node heap.

**Proof**

One only needs to prove that just before line 3 of algorithm 4 in the loop, the next nearest nodes in volume are on the heap, and algorithm 3 returns the nearest of them.

Just before line 3 of algorithm 4 in the loop, the beams with nodes on the heap are the nearest beams in volume. Line 5 of algorithm 1 and lines 1 and 2 of algorithm 3 ensure that each one has only one node on the heap. Such a node is no farther than the remaining nodes on the beam, which are not yet pushed on the heap.

If there is no remaining beam whose nodes are not on the heap, it is obvious that the heap contains the nearest node of the remaining nodes in volume and algorithm 3 returns the nearest of them.

Otherwise, line 5 to line 6 of algorithm 3 ensures that the farthest of beams with nodes on the heap have the nearest node of it on the heap. By proposition 2, the node is no farther than the nearest node on the next beam, which does not yet have a node pushed on the heap. Therefore, the heap contains the nearest node of the remaining nodes in volume. By proposition 2, $S$ in algorithm 3 is no farther than $S'$, and algorithm 3 returns the nearest of them. QED

If the grids are uniform, and the radar volume scan data can be preprocessed so that the beams of each sweep are uniform, then the above algorithms need only to be carried out for the grid points with $0^\circ \leq \phi_g \leq 45^\circ$. For others, the nearest nodes can be obtained by symmetric properties of radar volume scan.

The grid point with $0^\circ \leq \phi_g \leq 45^\circ$ has the Cartesian coordinates of $(x_g, y_g, z_g)$, and the calculated nearest nodes are referenced by $(SI, AI, GI)$ with $1 \leq i \leq k$, where SI is sweep index, AI is the azimuth index starting from 0 to 359, and GI is the gate index. The formulas to obtain nodes for seven other grid points are shown in Table 1.

The algorithms can also be used for multiple radars. If a grid point to be interpolated is inside the overlapped volume scan of two or more radars, algorithm 3 can be carried out for each radar alternately until $k$-nearest nodes are obtained.

**4. Conclusions**

Since the algorithms only rely on the assumptions made in section 2, the range bins, azimuth spacing, and elevation spacing do not have any influence on the algorithms.

To avoid the heap overflow when computing $k$-nearest nodes of a grid point, the sizes of beam heap and node heap in the program are set to $2k$ and $4k$. Therefore, in the worst case, the complexities of pushing a beam on the beam heap and a node on the node heap are $O(\log 2k)$ and $O(\log 4k)$, respectively. The complexities of getting the (next) nearest beam on a sweep and the (next) nearest node on a beam are both $O(1)$, or constant. Therefore, the upper bound of complexity of the algorithms is $O(1)$.

---

A sample program is included as an electronic supplement.
**Table 1. Formulas to obtain nearest nodes for those that are symmetric to grid point \((x_g, y_g, z_g)\).**

<table>
<thead>
<tr>
<th>Grid point</th>
<th>Nearest nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>((y_p, x_p, z_p))</td>
<td>([S_i, (90 - A_i + 360) \mod 360, GI])</td>
</tr>
<tr>
<td>((y_p, -x_p, z_p))</td>
<td>([S_i, (90 + A_i + 360) \mod 360, GI])</td>
</tr>
<tr>
<td>((x_p, -y_p, z_g))</td>
<td>([S_i, (180 - A_i + 360) \mod 360, GI])</td>
</tr>
<tr>
<td>((-x_p, -y_p, z_g))</td>
<td>([S_i, (180 + A_i + 360) \mod 360, GI])</td>
</tr>
<tr>
<td>((-y_p, -x_p, z_g))</td>
<td>([S_i, (270 + A_i + 360) \mod 360, GI])</td>
</tr>
<tr>
<td>((-y_p, x_p, z_g))</td>
<td>([S_i, (90 - A_i + 360) \mod 360, GI])</td>
</tr>
</tbody>
</table>

\[k[O(\log 2k) + O(\log 4k) + 2O(1)] = 2kO(\log k) + C_1k,\]

where \(C_1 = \log 2 + \log 4 + 2O(1)\).

When the algorithms are used in calculations for \(n\) grid points, since \(n\) is usually much greater than \(k\), the total complexity of calculating \(k\) nearest nodes in the worst case is

\[n[2kO(\log k) + C_1k] = [2O(\log k) + C_1]nk \approx O(k).\]

Figure 6 shows the relationship of time used to calculate \(k\)-nearest nodes in a single radar volume scan for \(460 \times 460 \times 10\) (2 116 000 grid points) and \(k\), with azimuth angles rounded to the nearest azimuth resolution unit (1°). It is obvious that the relationship is nearly linear.

The test results shown in Fig. 6 are obtained by running the program on a Dell Optiplex GX 745 (Intel Pentium D 3.4 GHz, 2 GB RAM). If \(k = 32\), the used time is less than 2 s for 2 116 000 grid points. It is reasonably fast and can be used for real-time applications.

**Acknowledgments.** The research in this paper is sponsored by the Beijing (China) Natural Science Foundation under Grant 8062014.

**APPENDIX**

**List of Algorithms**

**a. Algorithm 1: Initialization**
1) Get the nearest beam on each sweep (proposition 3)
2) Push those beams on the beam heap
3) Pop a beam \(B\) from the beam heap
4) Get the nearest node \(S\) on \(B\) (proposition 1)
5) Push \(S\) on the node heap

**b. Algorithm 2: Get next nearest beam in volume**
1) Pop a beam \(B\) from the beam heap
2) Get next nearest beam \(B'\) on the sweep on which \(B\) is located (proposition 3)
3) Push \(B'\) on the beam heap
4) **Return** \(B\)

**c. Algorithm 3: Get next nearest node in volume**
1) Pop a node \(S\) from the node heap
2) Get next nearest node \(S'\) on the beam \(B\) on which \(S\) is located (proposition 1)
3) Push \(S'\) on the node heap
4) **If** \(B\) is the farthest of the beams with nodes on the node heap, then
5) Get next nearest beam \(B'\) in volume (algorithm 2)
6) Get the nearest node \(S''\) on \(B'\) (proposition 1)
7) Push \(S''\) on the node heap
8) **End if**
9) **Return** \(S\)

**d. Algorithm 4: Calculate k-nearest nodes in volume for a grid point**
1) Do initialization for the grid point (algorithm 1)
2) **Repeat**
3) Get next nearest node in volume (algorithm 3)
4) **Until** the number of obtained nodes is \(k\)

**REFERENCES**


