APPENDIX A. SIMILARITY MEASURE

We wish to evaluate the similarity between a user's search and a dataset; specifically, we wish to quantify and replicate the observed user perception of "closeness" from Study 1. Our observations indicated that user judgments were not based solely on the nearest point, but also included some consideration of the furthest extent as well. As mentioned in Section 3.2, our candidate similarity measure adapts a feature-space model, with each search term being treated as a separate dimension.

If a search term specifies a desired data range and a matching variable is found in a dataset, we need a measure that can provide a similarity value that resonates with the intended users. Cognitive science has long recognized that people frequently use distance as a metaphor for similarity, including interpreting time (and other entities) as distance [1]. We use a distance-based measure that compares the search range to the data range. While we use distance as a basis, we are approximating similarity, thus, a smaller distance translates to a higher score. There are many options for representing the proximity of two entities (in our case, the search term and the matching dataset feature), with varying computational complexities [2]. We build on a common and easily-calculated distance measure: the minimum (and maximum) distance between two features, which can be estimated by knowing the features' bounds. We can use this measure to identify whether a feature is completely within the search term bounds and so is a complete match; or whether the two are disjoint; or if they overlap, and, if so, by how much.

Our distance measure for a one-dimensional variable, such as salinity or time, is shown in Equations 1-4; Fig. 12 depicts our measure graphically. In essence, we regard the observation values within a dataset as a distribution of "distances" from the query center, with a single point value (such as a constant value for a variable, or a single time) being the most constrained distribution. Each search term itself represents a desired distribution of the relevant variable. (At present, we regard the contents as being equally distributed between the bounds, as is common in database indexing [3]; future research may consider alternate distributions.)

Let Q_{Rmn} and Q_{Rmx} represent the lower and upper bounds of the search term, and let v_{Xmn} and v_{Xmx} represent the minimum and maximum values of observations for the matched variable v in a dataset d. (In the case of time, for calculation purposes all times are translated into a monotonically increasing real number, for example "Unix time").

Equation 1 calculates v_{Rmn}, the distance of variable v's minimum value from the search term's "center", i.e., the mean of Q_{Rmn} and Q_{Rmx}, and then scales the result by the search term "radius" (half the size of the term's interval). Similarly Equation 2 calculates v_{Rmx}, the "scaled variable-range distance" of the variable's maximum value. Equation 3 calculates an overall distance for this variable's range from the search term's range, normalized by the search term radius. The first subcase applies to variable values completely within the term's range, and thus at a distance of 0 radii. The second through fourth cases account for a variable range overlapping the search range at the high end, at the low end, and on both sides, respectively; these subcases have the distance calculation adjusted based on the percent of the variable's range that is estimated to be inside the search range. The last subcase accounts for a dataset completely outside of the search term's range.

In Equation 4, we then apply a scaling function s to v_{Rdist} to convert the calculated distance from the search term center into a relevance score v_{Rs}, while allowing a weighting factor to be applied to the distance result. Per Montello [4], this implicit scaling factor may change for different users or different tasks; our current implementation uses $s(v_{Rdist}) = (100 - f * v_{Rdist})$. That is, if the distance is f "radii" (currently $f = 10$) from the search term's center it is considered "too far away" to be relevant and given a score of 0 or less, while a distance of 0 (i.e., completely within the search term's range) is given a score of 100.

Equations 5 and 6 adapt this measure to the case of ranking geospatial features by an estimate of overall "closeness" to the search term. Similarly to the one-dimensional measure in Equations 1-4, we use minimum and maximum distance between the search term and dataset feature. This measure takes the overall shape into account, unlike a nearest-neighbor approach, is more nuanced than the often-used categorization of the spatial relationship of two shapes into *contains*, *intersects* or *disjoint*, and is easy to calculate. For simplicity in the interface, the search term is represented as a rectangle on a map. The geolocations within the dataset can be represented by any of the common geometries: point, line, polyline or polygon (e.g., convex hull) [5].

For a geospatial feature: let C represent the center location

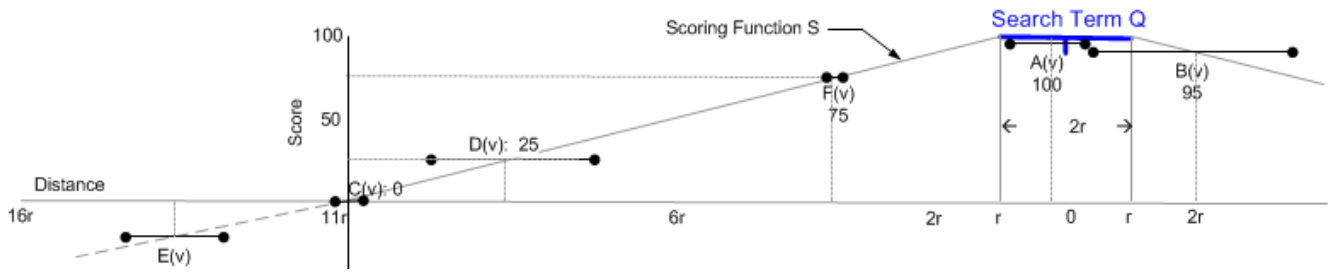

Fig. 12. Graphic depicting a portion of the candidate distance measure, as applied to a one-dimensional variable (v) to calculate a similarity score. The search range Q is shown in blue; the search radius r is one-half of the range of Q. The range for variable v in each of six datasets (labeled A to F) is shown, along with each dataset's resulting score for this search term. For each dataset, the scoring function S identifies the middle of the range of the variable v, scales it by the search radius, and converts the result to a score. Datasets wholly inside the search range (dataset A) are given a score of 100. Datasets wholly outside the search range are given a score based on the number of radii that the middle of the range is from the search center (datasets C-F). Datasets whose middle is at 11 radii from the search center are given a score of 0, and ones further away may receive negative scores (dataset E). Datasets that overlap the search range are scored based on the proportion of the dataset range that is inside the search range (dataset B).

$$v_{Rmn} = \frac{v_{Xmn} - ((Q_{Rmx} - Q_{Rmn})/2 + Q_{Rmn})}{(Q_{Rmx} - Q_{Rmn})/2} = (2v_{Xmn} - Q_{Rmx} - Q_{Rmn})/(Q_{Rmx} - Q_{Rmn}) \quad (1)$$

$$v_{Rmx} = (2v_{Xmx} - Q_{Rmx} - Q_{Rmn})/(Q_{Rmx} - Q_{Rmn}) \quad (2)$$

$$v_{Rdist} = \begin{cases} 0 & v_{Xmn} \geq Q_{Rmn}, v_{Xmx} \leq Q_{Rmx} \\ \frac{(|v_{Rmx}|-1)^2}{2|v_{Rmx} - v_{Rmn}|} & v_{Xmn} \geq Q_{Rmn}, v_{Xmx} > Q_{Rmx} \\ \frac{(|v_{Rmn}|-1)^2}{2|v_{Rmx} - v_{Rmn}|} & v_{Xmn} < Q_{Rmn}, v_{Xmx} \leq Q_{Rmx} \\ \frac{(|v_{Rmx}|-1)^2 + (|v_{Rmn}|-1)^2}{2|v_{Rmx} - v_{Rmn}|} & v_{Xmn} < Q_{Rmn}, v_{Xmx} > Q_{Rmx} \\ (|v_{Rmn} + v_{Rmx}|/2) - 1 & v_{Xmn} > Q_{Rmx} \vee v_{Xmx} < Q_{Rmn} \end{cases} \quad (3)$$

$$v_{Rs} = s(v_{Rdist}) \quad (4)$$

$$d_{Gdist} = \begin{cases} 0 & d_{Gmx} \leq r \\ \frac{(d_{Gmx}/r - 1)^2}{2(d_{Gmx} - d_{Gmn})} & d_{Gmn} \leq r, d_{Gmx} \geq r \\ (d_{Gmn} + d_{Gmx})/2r - 1 & d_{Gmn} > r \end{cases} \quad (5)$$

$$d_{Gs} = s(d_{Gdist}) \quad (6)$$

of the geospatial search term and r the radius.[1] Let the locations of all the observations within a single dataset d be represented by a geometry g. Let d_{Gmn} and d_{Gmx} represent the minimum and maximum distances of the geometry from C, using some distance measure such as Euclidean distance. Then Equation 5 calculates the overall distance measure for three subcases: the dataset's geometry is completely within the search radius; it overlaps the search area, or it is completely outside the search area. In each case, a distance is calculated. Equation 6 gives a geospatial-relevance score d_{Gs} for dataset d by again applying the same scaling function s to the calculated overall distance measure.

Lastly, the scores v_{Rs} for each search term, including any geospatial score d_{Gs}, are combined to give an overall score d_{score} for this dataset. We take a simple average of these scores. Note that each of these scores has been converted to a unitless value implicitly weighted by the user via the radii of the individual search terms.

We tested several variations of our distance measure on the datasets judged by the study participants. (Our approach is in common with TREC approaches that evaluate alternate measures against previously judged documents, rather than the entire corpus.) The candidate measure, as described in Equations 1-3, uses the center of a variable's range as the point to which distance is calculated; that is, mindist + range/ 2. The variations evaluated different weightings of the closest edge versus the center, per indications from the first user study and supporting informally expressed opinions by some study participants. In particular, we evaluated:

SN: mindist
S2: mindist + range/ 8
S3: mindist + range/ 4
S4: mindist + 3/ 8*range
SX: mindist + range (i.e., maxdist)

In addition, we tested several variants of a "Euclidean" measure. There are multiple ways that a Euclidean distance can be calculated between two regions (in particular, between a query and a dataset summary). A commonly used surrogate for distance between geographic entities is centroid-to-centroid distance, although it is regarded by some as a poor approximation when the entities are large and close together [2] and it is not known how well it reflects how users perceive distances. However, alternatives (such as closest-point distance) have similar or worse problems, so we chose centroid distance. To use this measure for queries involving incomparable dimensions — for example, kilometers and days—requires scaling of some sort. We calculated distance based only on matched search terms and for each component value we use the search-centroid and the summary-centroid. The Euclidean distance tested in Section 4.2.3 normalizes a value by the maximum range of the corresponding variable in our corpus. (For example, a temperature value is divided by the difference between the maximum and minimum values present in the corpus.) We also tried several variants on normalization, including no normalization. The overall results of each variant were similar, although the individual categories of searches showed significant variations. None of these variants performed as well as our measure. Note that our measure scales based on query bounds and not just the query centroid, and also adjusts for overlaps between search-term ranges and summary bounds.

The results of the evaluation are shown in Section 4.2.3.

[1] While the formulae described here are for circular search regions for simplicity in exposition, we have adapted these formulae for non-circular search regions by using an analog to radius based on the search region shape.

APPENDIX B. ALGORITHM FOR DATASET SCORING

We treat searches as a conjunction of desired features. In general, we represent each search term as a tuple of the form *<variable, range, units>*. For example:

```
{<"time", [2010-04-15:2010-07-15], date>,
     <"temperature", [5:10], C>, … }
```

Dataset summaries, stored in our catalog, are likewise a set of features of the same form. In general, we have one feature for each column within a tabular dataset. (Exceptions may exist. For example, we combine latitude and longitude into a single geospatial feature, and have considered adding elevation.) This model gives us symmetry between the search terms and the dataset features, which allows a returned dataset to itself serve as a search query to find similar datasets.

Conceptually, each search generates a rating of every catalog entry (although for performance reasons, our implementation avoids doing so). Fig. 13 shows simplified pseudo-code for our algorithm.

For each dataset, we first match each search term to a feature of this dataset, or to no feature (function Match). Our initial matching function, used in the user study and reflected in the pseudo-code here, simply looks for an exact match between the variable named in the search term and a named column in the dataset. (This approach is naïve and is not required by our model [6]. Work is underway to address the problems of normalizing the variable names in the archive [7].)

Next, we score each search condition and matched variable. If no variable was matched to a search term (for example, there is a temporal search condition but the data is from an unknown time), we give that search condition a "null" score. If a variable is matched and the search term merely requests the existence of that variable in the dataset (e.g., `<"temperature",,>`) we count it as a complete match for the term and give it the maximum possible score (MaxCondScore). If a variable is matched and the search term contains a range, the similarity between the search term and matching variable is scored using the distance measure described in Appendix A.

After processing all terms we normalize the final score by the number of terms in the search to give an overall score for each catalog entry. As described in Appendix A, in our current implementation we set the MaxCondScore to 100; the minimum score is not bounded.

Finally, we sort the resulting list of scores and return the catalog entries with the highest scores. We are exploring optimizations to this pseudo-code, such as traversing hierarchical relationships between datasets, to derive the same results more efficiently.

Subsetting Data Into Hierarchies

An individual dataset maybe partitioned into multiple "virtual datasets", as when a dataset covering a long time period or large geographic area is divided into smaller subsets.

Partitioning choices must in general be made before a dataset is scanned and its features inserted into the metadata catalog. At CMOP, these choices are made for each major category of data collected; for example, temporary sensors mounted on mobile platforms are partitioned by time and path segmentation, while permanently installed sensors on stationary platforms are partitioned by time only. While partitioning on a variable value is also possible, we have not yet encountered a case where this partitioning was requested by our scientists; however, a smaller time period or geographic area will generally translate to a smaller range for each observed variable.

A parent dataset and its contained subsets coexist in the metadata catalog; in fact, multiple different segmentations of the same data can coexist within the metadata catalog [8]. The hierarchical relationship of the contained subsets can be captured in the metadata catalog. The search engine returns virtual or real datasets from all levels of the hierarchy based on their scores, allowing the "closest" dataset subset to be returned for a search. Thus, subsets and supersets of the same data may be returned for the same query, but at different places in the ranking, with the objective of returning the most useful dataset subset for the current search.

Scoring Textual Data

In addition to the large quantity of numeric data, some datasets contain one or more fields of textual data. For all columns containing textual data, we currently allow searches for the existence of these variables.

There are a few data fields, such as research notes or study descriptions, for which traditional textual search may apply. However, discussions with our scientists lead us to believe that traditional text search is inappropriate for most textual variables. Examples cited at CMOP of such textual variables include quality levels, species names, and unique hybrid textual-numeric identifiers given to water samples and test locations. The notion of distance still seems germane to such variables. For example, quality levels are ordered, species are more or less related, and the numbering schemes were often selected for memorability by embedding a notion of distance. Thus, we expect notions of distance still apply, though the details of the similarity measures may need to change.

We experimented with using our distance measure for ordinal categorical data, such as quality levels. We assigned each quality level a value within a range that respects its ordinal position, with "no quality control" having the lowest value and "full quality control" having the highest. We convert the search term range and data ranges to the matching numeric values and apply our distance measure to these values. While this approach has been well accepted by our implementation's users, it relies on assessing each textual variable independently for applicability and to assign the ordinal values, and so we do not see this approach as viable for large archives with many variables. Other measures will be required for distances that cannot be captured by a linear order, for example, distance between species names.

```
Inputs: Search specification S, catalog entry collection D, desired number of highest-scoring entries
k

Initialize array Scores[]
For each catalog entry d in collection D do:
  Scores[d] = Score(S,d)
Sort values in Scores[]
Return: Scores[1..k]

Score(S,d):
// Calculate similarity score for search S on
// dataset d
  MaxCondScore = 100 // Implementation choice
  C = Match(S,d)
  s = 0 // initialize score accumulator to 0
  For each tuple (s,c) in C do:
    If (c is Null): // No match found
       Continue     // No change to score
    Elif (s is a variable existence condition):
        s = s + MaxCondScore
    Else // variable with range condition
        s = s + Scoring(s, c) //apply measure
  Return: s/|S|

Match(Q,d):   // Naïve version
// Match search terms to features
   matched = {}  // initialize empty set
   For each search condition q in Q do:
     If (requested variable v exists in d):
        Add (q,v) to matched
     Else:
        Add (q,Null) to matched
   Return: matched
```

Fig. 13. Simplified pseudo-code for dataset summary ranked search algorithm.

Combining textual and numeric search terms remains an area for future research. While it is mathematically possible to combine scores from these two methods – for example, by including the score for each textual term in the final score normalization – we do not yet have a model for how scientists perceive these combinations. In particular, unlike the continuous numeric and existence measures, we have not validated these additional approaches with formal user studies. While there are certainly technical issues to be addressed, we believe more user studies exploring how users expect the systems to operate to be the most pressing issue.

REFERENCES

[1] G. Lakoff, *Where Mathematics Comes From: How the Embodied Mind Brings Mathematics Into Being*, 1st ed. New York NY: Basic Books, 2000.

[2] H. J. Miller and E. A. Wentz, "Representation and spatial analysis in geographic information systems," *Annals of the AAG*, vol. 93, no. 3, pp. 574–594, Sep. 2003.

[3] V. Markl, M. Kutsch, T. Tran, P. Haas, and N. Megiddo, "MAXENT: consistent cardinality estimation in action," in *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, 2006, pp. 775–777.

[4] D. R. Montello, "The measurement of cognitive distance: Methods and construct validity," *Journal of Environmental Psychology*, vol. 11, no. 2, pp. 101–122, 1991.

[5] J. R. Herring, Ed., "OpenGIS® Implementation Standard for Geographic Information - Simple Feature Access - Part 1: Common Architecture." Open Geospatial Consortium, 2010.

[6] V. M. Megler and D. Maier, "When big data leads to lost data," in *Proceedings of the 5th Ph. D. workshop on Information and knowledge (PIKM)*, 2012, pp. 1–8.

[7] V. M. Megler, "Taming the Metadata Mess," presented at the Phd Workshop for ICDE 2013, Brisbane, 2013.

[8] V. M. Megler and D. Maier, "Finding haystacks with needles: Ranked search for data using geospatial and temporal characteristics," vol. 6809, Springer Berlin / Heidelberg, 2011, pp. 55–72.