Application of a weighted Dice Similarity Coefficient (DSC) for Structure Comparison in Radiation Treatment Planning

by
Zachary Fried

A THESIS

submitted to

Oregon State University

Honors College

in partial fulfillment of
the requirements for the
degree of

Honors Baccalaureate of Science in Chemistry
(Honors Associate)

Presented August 19, 2019
Commencement June 2020

AN ABSTRACT OF THE THESIS OF

Zachary Fried for the degree of Honors Baccalaureate of Science in Chemistry presented on August 19, 2015. Title: Application of a weighted Dice Similarity Coefficient (DSC) for Structure Comparison in Radiation Treatment Planning.

Abstract approved:_____

David Koslicki

Coded metrics that compare representations of structures within the human body to evaluate their similarity are considered useful for planning radiation therapy on cancer targets. However, existing metrics fail to take into account the dose of radiation that will be received by both the target, and surrounding non-target tissue, during the treatment, distinctions that can have critical clinical significance. A radiation dose modified Dice similarity coefficient (mDSC) function was coded in R and integrated into the RadOnc R package. Comparison of the traditional unweighted DSC to the modified function suggests that the modified function might provide a more relevant structure comparison in the context of radiation therapy.

Key Words: radiation therapy, treatment planning, Dice similarity coefficient, RadOnc

Corresponding e-mail address: friedz@oregonstate.edu

Application of a weighted Dice Similarity Coefficient (DSC) for Structure Comparison in Radiation Treatment Planning

by
Zachary Fried

A THESIS

submitted to

Oregon State University

Honors College

in partial fulfillment of
the requirements for the
degree of

Honors Baccalaureate of Science in Chemistry
(Honors Associate)

Presented August 19, 2019
Commencement June 2020

Honors Baccalaureate of Science in Chemistry project of Zachary Fried presented on August 19, 2019.

APPROVED:

_____

David Koslicki, Mentor, representing Mathematics

_____

Reid Thompson, Committee Member, representing Radiation Medicine

_____

David Hendrix Committee Member, representing Biochemistry and Biophysics

_____

Toni Doolen, Dean, Oregon State University Honors College

I understand that my project will become part of the permanent collection of Oregon State University, Honors College. My signature below authorizes release of my project to any reader upon request.

_____

Zachary Fried, Author

ACKOWLEDGEMNTS

I would like to acknowledge and thank the following for their contributions and support of this project:

Thank you to my thesis committee members, Dr. David Hendrix, for your time commitment and support in the completion of this project.

Thank you to my mentor, Dr. David Koslicki for your continued support and expertise in helping me complete my thesis.

Thank you to Dr. Reid Thompson and Dr. Abhinav Nellore for welcoming me into your lab, introducing me to the problem to be solved, and providing extensive guidance along the way.

Table of Contents:

Introduction

The goal of radiation therapy is to kill cancer cells or shrink tumors using ionizing radiation. Both normal and cancer cells rely on intact and properly behaving genetic material to grow and proliferate. Ionizing radiation carries enough energy to remove electrons from atoms and molecules, posing a risk to the health of exposed cells. Cells with significantly damaged genetic material that is not repaired generally stop dividing and die. Normal cells are better able to repair their genetic material compared to most cancers and it is this distinction that radiation therapy exploits (Baskar *et al*, 2012). For maximum effectiveness, radiation dose, the amount of ionizing radiation received in a given (3-D) region, should be maximized for cancer cells and minimized for normal cells adjacent to the cancer. Exposure optimization aims to kill the cancer cells while preventing serious damage to healthy tissue. In worst case scenarios, healthy tissue can become cancerous if exposed to radiation during radiation treatments, making it vital to correctly identify organs and other structures in the human body during treatment planning (Baskar *et al*, 2012).

Medical imaging, such as with CT scans or magnetic resonance imaging (MRI), allows clinicians to see inside the body and create a virtual patient whose treatment can be designed in software called a treatment planning system (TPS). These systems allow the user to visualize the body structures in 3D inside the body and select dosimetry settings for optimal patient outcomes, among other analytical and planning functions. An R statistical package "RadOnc" (CRAN) was published (Thompson, 2014) that facilitates dosimetric analysis and structural visualization and comparison of planning data imported from a TPS. The structural comparison function typically generates similarity scores for a set of two or more structures. This can be useful, for example, in comparing the accuracy of a structure drawn by a physician, student, or algorithm compared to a

known structure or that of another expert (Boon *et al,* 2018). Such structures are drawn in a process called contouring, in which a clinician uses medical imaging data to delineate the outline of the cancer and any organs at risk (OAR) which should receive minimal radiation (Boon *et al,* 2018). The visible cancer volume is known as the gross tumor volume (GTV). The clinical tumor volume (CTV) includes the GTV with added margins to account for the potential spread of the tumor between the imaging and the therapy session, while the planning target volume (PTV) is larger still, including an additional buffer to account for potential movement by the patient during treatment (Baskar *et al*, 2012). These structures and organs at risk can be compared between treatment plans or within a single plan with the RadOnc package via pairwise analysis with the Hausdorff distance and the Dice similarity coefficient (DSC) (Thompson, 2014). These metrics are successful in describing a distance and similarity between structures, respectively. However, they do not account for the amount of radiation received by each region or how differences in structure may affect whether a structure receives a desirable or undesirable radiation quantity. The DSC, named after its developer, botanist Lee Raymond Dice, is a measure of similarity between two samples as defined below (Dice, 1945).

$$DSC = \frac{2|X \cap Y|}{|X| + |Y|}$$

|X| and |Y| are the number of elements in each sample. In the "DSC" method within the RadOnc package, the space (in which the structures to be compared are located) is divided into a 3-dimensional matrix of a specified resolution, with each matrix cell referred to as a voxel. A structure's set is made up of all of the voxels that have a midpoint contained within the structure's volume. |X| and |Y| are thus the number of voxel midpoints that are contained within each structure and |X∩Y| is the number of voxel midpoints that are contained by both structures

(Thompson, 2014). Maximum similarity corresponds to a DSC of 1 while completely dissimilar structures would return a DSC of 0.

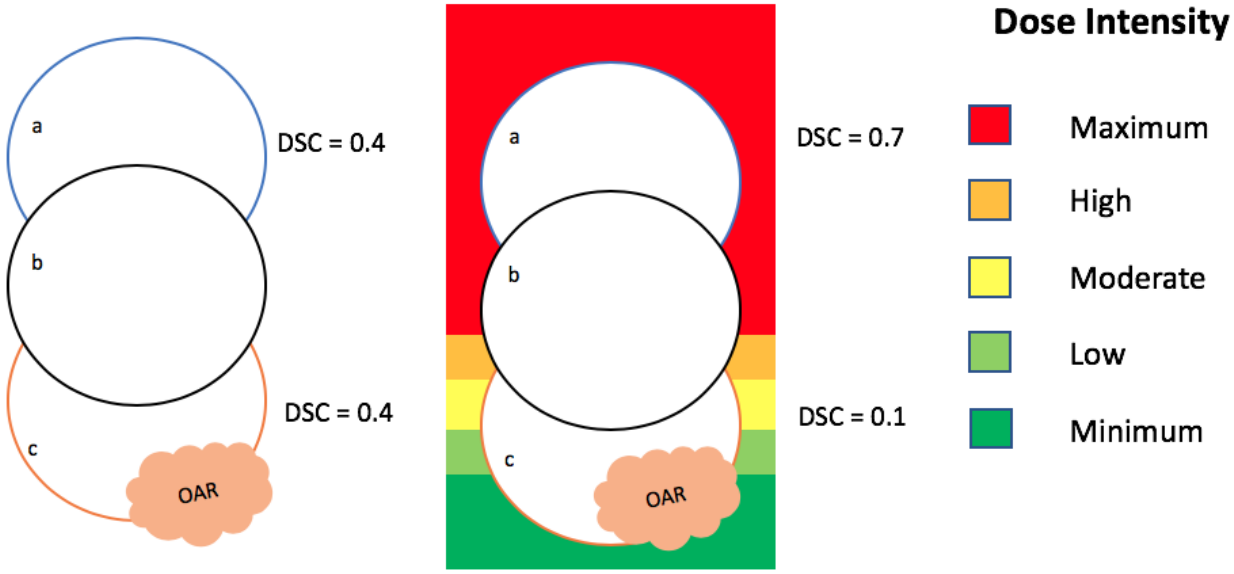

Figure 1. Dice Similarity Coefficient Outputs Same Value for Clinically Different Structures

Figure 1 illustrates how this could become a problem in the context of radiation treatment planning. Structures a and c are created by shifting structure b up or down, respectively. Without an input of radiation dose, the traditional DSC between structure b and structure a and between structure c and structure b in comparison I would be equal. The organ at risk (OAR) is overlapped by structure c and would be in serious risk if the circular structure was designating the radiation target. If structure b represents the area that would ideally be targeted by radiation and structure a and structure c are two attempts to draw the target area, structure a is a better attempt than structure c. While both structures partially miss the ideal target area, structure a does not overlap an organ at risk. Data imported from a TPS often includes a dose grid, a 3-D matrix that contains information about the amount of planned radiation that will be received at regularly spaced x, y, z coordinates in the same space as the structures to be compared

(Thompson, 2014). The planned dose in the volume of tumors is relatively high while organs at risk are located in low dose areas of the dose grid. Comparison II in Figure 1 shows the same structures and organ at risk as compared before but overlaid on a dose grid. The organ at risk receives minimum to low radiation dose intensity as planned in the TPS before data import. Radiation dose increases as one travels away from the organ at risk towards the treatment target. Only considering the proposed target structures and dose grid, it is clear that using structure a as a target would lead to a better patient outcome than using structure c as the target because targeting areas that one TPS suggests requires minimal radiation dose would lead to excessive damage to non-target organs. A modified DSC that weights the importance of each voxel by the radiation dose's behavior at that point could lead to a more meaningful metric in the context of radiation therapy planning. Such a weighted DSC (mDSC) might produce results as shown in comparison II where structure a and structure b are calculated as being more similar than structure b and structure c.

A modified Dice similarity coefficient was coded in the R programming language that weights voxels by the behavior of the dose grid to provide a more meaningful method of set comparison than the traditional Dice similarity coefficient in the context of radiation therapy treatment planning. This tool was added to the compareStructures function in the package RadOnc and thus will be available for free from CRAN for use on multiple computing platforms.

Methods

The mDSC method of structure comparison was added to the RadOnc package using R 3.6.1 on Mac OS X (R Core Team, 2019). To be used as currently designed, the function requires the RadOnc package to be installed along with its dependencies (Thompson, 2014). Evaluating the mDSC of structures is done via the compareStructures(structures, dose, method, pixels)

function used to compare the DSC and Hausdorff distance by specifying the "method" function argument as "mDSC" and providing a dose grid under the function argument "dose". The dose grid is automatically imported and formatted by RadOnc's read.DICOM.RT and is located at data$dose if "data" is defined as the location for the import. This metric uses the closed polygons (closed.polys) to represent the volumetric structures for comparison. Each closed.polys is a list containing sublists of x and y coordinates for each z "slice" from the CT scan or MRI. These polygon-containing 2D z-slices create the 3D structure volumes they are a part of when placed on top of each other, allowing the polygons to be connected between each slice.

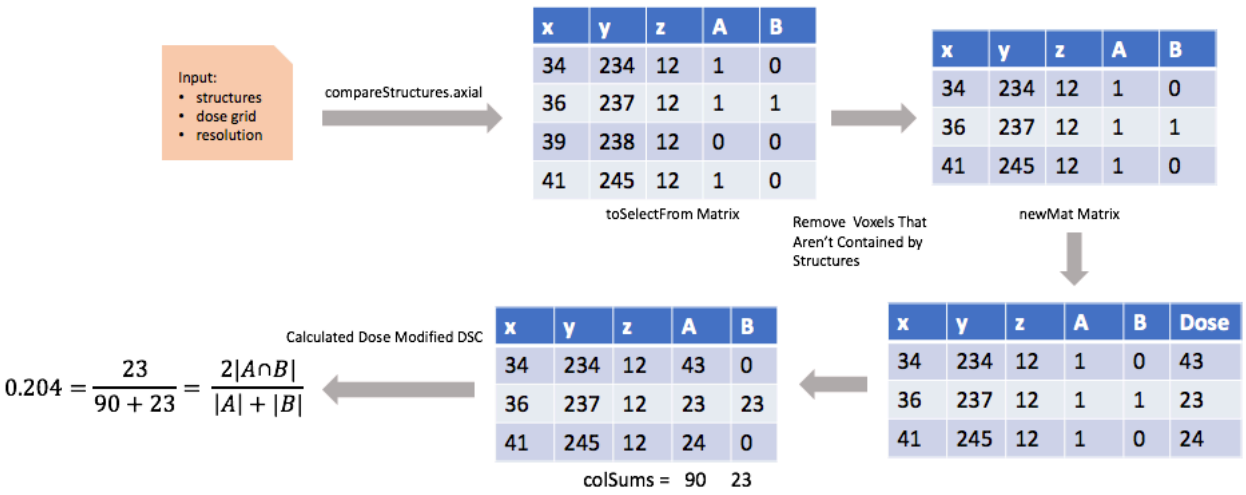

Figure 2. Workflow for compare.Structures with Method = "mDSC"

The initialization of the mDSC method relies on the first half of the DSC method which takes the structures to be compared as input and divides the 3D space they reside in into voxels. A matrix is then created with N+3 columns with N being the number of input structures. The first three columns describe the location of the midpoint of each voxel while the remaining columns, one per structure, indicate whether a structure contains the voxel midpoint. A value of 1 in a column indicates that the structure that the column is associated with contains the voxel midpoint indexed by that row. This matrix is then filtered to remove any rows that describe voxel

midpoints that are not contained by any structure. These voxels will not affect the mDSC calculation, but this early removal saves computation time later in the workflow.

Borders between radiation targets and organs at risk are locations where a steep gradient in radiation dose exists. One object will receive high radiation in order to kill or shrink the cancer while it's next-door neighbor will need to receive minimal to zero radiation to avoid instigating a secondary cancer or other organ damage. The gradient of the dose grid is one way to measure this steep change and must be calculated for each voxel. The function approx3D(data = data$dose, x, y, z) is used to extrapolate from the dose grid to output an approximated dose at any specified x, y, z location. The midpoint dose is approximated as well as the dose at each corner and face of the voxel. The difference in any corner or face dose from the midpoint dose is assigned as the magnitude of a vector pointing from the midpoint through that corner or face. This leads to each voxel midpoint possessing 14 vectors of various magnitudes. These vectors are summed to get an approximated net gradient vector. The magnitude of the net vector is assigned to that voxel. The matrix that used to contain 0s and 1s depending on whether that voxel's midpoint was contained in a structure is altered to contain 0s and weights, with the weights being equal to the magnitude of the gradient centered on that voxel, approximated as just described.

The mDSC is then calculated by summing the columns for each structure and selecting a pair of structures to compare, designating one as X and the other as Y. The column sum for structure X is used in place of |X| and is added to the column sum of Y (instead of |Y|) to compute the denominator. If a voxel is contained by both structures, the magnitude of the gradient at that voxel's midpoint is added to a sum that is later doubled to form the numerator in place of $2|X \cap Y|$. The mDSC is then calculated as before and displayed in a table.

Results

Data from a sample prostate cancer treatment plan from a TPS used at OHSU was imported with RadOnc. The planning target volume (PTV70 CT_1) structure was used for the evaluation of the WDSC metric and is depicted in Figure 3. This structure was shifted in the positive and negative x and y directions by half of its width and height, respectively.

Table 1. Similarity Coefficients from the Traditional DSC Using PTV70 CT_1

	PTV70 CT_1	Posterior Shift	Anterior Shift	Superior Shift	Inferior Shift
PTV70 CT_1	1	0.581	0.584	0.479	0.507
Posterior Shift	0.580	1	0.130	0.336	0.386
Anterior Shift	0.584	0.130	1	0.361	0.362
Superior Shift	0.479	0.336	0.361	1	0.021
Inferior Shift	0.507	0.387	0.362	0.021	1

Table 2. Similarity Coefficients from the Dose Modified DSC Using PTV70 CT_1

	PTV70 CT_1	Posterior Shift	Anterior Shift	Superior Shift	Inferior Shift
PTV70 CT_1	1	0.622	0.628	0.493	0.549
Posterior Shift	0.622	1	0.168	0.366	0.442
Anterior Shift	0.628	0.168	1	0.403	0.412
Superior Shift	0.493	0.367	0.403	1	0.037
Inferior Shift	0.549	0.442	0.412	0.037	1

Table 3. Mean Dose in Each Shift of the PTV Structure

Shift	Mean Dose (Gy)
Anterior	39.25458
Posterior	37.54306
Superior	27.98347
Inferior	49.81673
none	38.85445

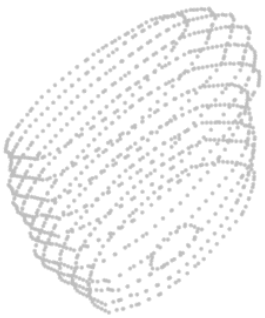

Figure 3. 2.5D representation of the planning target volume PTV70 CT_1

The movement in the anterior and posterior direction had similar DSC values for the traditional DSC at 0.584 and 0.580 for the anterior and posterior shifts respectively, and for the modified metric with DSCs of 0.622 and 0.628. The movement in the superior and inferior

directions produced slightly different DSC values for the traditional metric at 0.479 and 0.507. The dose modified DSC had a much larger difference between the superior and inferior shifts with values of 0.493 and 0.549.

The resolution of the space was altered by changing the number of voxels in each direction by changing the function argument "pixels". The minimum pixels required to complete the calculation was 30 with the structure tested. This value can be increased to any amount but significantly increases the computation time. However, the difference in DSC results for a given structure pairing using pixels = 30 vs the standard pixels = 100 was negligible, at < 0.0001.

Discussion

The results from the traditional and modified DSC metrics are best understood in the context of the mean dose table. This table shows that the posterior and anterior shifted structures receive approximately the same radiation dose. Even with a dose modified metric, the anterior and posterior shifted structures' DSC should be approximately equal, which was observed. The superior and inferior shifted structures receive mean doses with a larger difference. As expected, the difference between the superior and inferior shifted modified DSC is larger than the traditional metric because there must be a change in radiation in these directions as the mean dose changes. This magnification of dissimilarity is due to the radiation dose behavior in the voxels contained within the PTV70 CT_1 structure and its shifted versions.

These results support that the dose modified DSC is able to magnify or reduce the calculated similarity between structures based on the interaction of a structure's shape and position in the dose grid. Further testing of this metric with more applicable clinical data could more definitively determine the efficacy of the code. The computation time required to calculate the DSC is proportional to the desired resolution. If a low resolution is acceptable, the

computation can be completed in less than one minute. If a high resolution is required, the calculation can take several hours to complete, as currently coded. The rate-determining step in the code is the creation of a matrix for every voxel midpoint that contains data on the direction and magnitude of the radiation dose gradient. The approx3D function is used 15 times for every voxel. If the standard 100x100x100 resolution is used, approx3D is calculated 15 million times. To improve function speed, one could store the approx3D result from the last voxel instead of calculating it again. For example, the point at the top of a voxel is the same point at the bottom of the voxel above. If the approximation function is used once for every point rather than up to eight times for a corner, computation time can be minimized.

The standard resolution is a limitation regarding structure size. While higher resolution leads to a more precise estimate of DSC, the computation time required to fill a large structure with many small voxels would likely outweigh the benefits. This limits the ability to easily compare large and small structures to each other due to the mismatch in ideal voxel size. If computation time is not limiting, voxel size should be selected based on the size of the smallest structure to be compared.

This instance of the DSC uses the magnitude of the net dose gradient vector at each voxel midpoint as the voxel's weight. Changing the algorithm to instead calculate the magnitude of the radiation dose as the weight is a potentially useful alternative modification that could be made to DSC for some clinical applications. The advantage of using the gradient as the weight is that accuracy at structure borders, characterized by steep changes in radiation dose, will be weighted higher than other regions. Borders need to be accurately contoured by physicians because inaccuracies in these regions can cause significant damage to non-target organs or render ineffective treatment of the targeted structure. Using the radiation dose magnitude as the weight

of each voxel assigns greater importance to structures receiving a high dose of radiation and less importance to areas with a lower dose. However, a disadvantage of dose magnitude weighting is that differences at structure boundaries, where radiation dosage is frequently reduced, would be weighted as less important, which is antithetical to the goal of high precision structure delineation that motivates structure comparison in the first place. There is no difference in how the algorithm treats high, medium, or low radiation dose regions. Different organs have varying levels of radiation that they can withstand, and tumor targets may have medium to high radiation dose prescribed. Using the dose gradient magnitude weighting facilitates comparison of many types of organs and targets, regardless of the dose magnitude in these regions.

This dose modified DSC is more likely to be used in research than in a clinical setting. Modern treatment planning systems and other available software allow precise contouring of body structures and such a metric would not be useful to the average physician. It might be useful in the development of a contour drawing algorithm, providing a measure of similarity to an ideal structure that could tell the progressing algorithm when it's close to clinical expectations.

Prior to the development of the dose modified DSC, a dose weighted Earth Mover's Distance (EMD) was proposed and partially developed to solve the same problems with structure comparison without dose information as addressed in this paper. The Earth Mover's Distance is a distance metric that has been used in image retrieval and computer vision (Rubner et al, 2000). The distance is based on the solution to a transportation problem in which suppliers must supply consumers with a given capacity. They must find the least expensive flow, or strategy of transport, to satisfy all demand of consumers. The cost of moving supplies can be defined as the ground distance between supply and location of consumer multiplied by the amount or weight of supplies to be moved. The metric gets its name from the idea that it describes a situation in

which one distribution is earth spread out in space while the other distribution is a set of holes spread in that same space that must be filled by the earth, in the same way that consumers must receive their desired supplied products. In this application, one structure would be defined as the supply or earth while the other structure defined as the consumer or holes (Rubner et al, 2000).

The definition of the EMD can be formalized as linear programming problem between two sets, A and B. In the context of radiation therapy, A and B would be structures to be compared. A = $\{(A_1, w_{p1})\ldots, \{(A_m, w_{pm})\}$ where m is the number of points to be compared on structure A and w is a weight defined for that point. B = $\{(B_1, w_{p1})\ldots, \{(B_n, w_{pn})\}$ where n is the number of points on B. The weight in this application could be the radiation dose or the gradient of the dose at the point on the supply structure. The ground distance between A_i and B_j is defined as $d_{i,j}$. The solution is to find a flow $f_{i,j}$between A_i and $B_j$that minimizes overall cost. As the distance between points and weight of points are multiplied, the result of this multiplication is work (Rubner et al, 2000).

$$WORK(A,B) = min \sum_{i=1}^{m} \sum_{j=1}^{n} f_{i,j}\, d_{i,j}$$

There are several constraints to the problem. First, supplies can only be moved in one direction (1).

$$f_{i,j} \geq 0 \qquad 1 \leq i \leq m, \quad 1 \leq j \leq n \qquad (1)$$

The second constraint (2) is that each point on the supply structure A can only supply the amount of weight in a given point and that each point in consumer structure B can only receive supplies up to their weight.

$$\sum_{j=1}^{n} f_{i,j} \leq w_{Ai} \quad \sum_{i=1}^{m} f_{i,j} \leq w_{Bj} \tag{2}$$

The final constraint (3) is that the amount of supplies moved between structures is at a maximum. This is known as total flow.

$$\sum_{i=1}^{m} \sum_{j=1}^{n} f_{i,j} = \min(\sum_{i=1}^{m} w_{Ai,} \sum_{j=1}^{n} w_{Bj}) \tag{3}$$

Once the optimal flow is found and the transportation problem solved, the EMD is defined as work normalized by the optimal flow.

$$EMD\ (A,B) = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} d_{i,j} f_{i,j}}{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{i,j}} \tag{4}$$

The total flow (3) is the total weight of the smaller structure. This normalization prevents favoring smaller structures (Rubner et al, 2000). In image retrieval applications, this normalization is usually sufficient in allowing for partial matching. The EMD is only a true metric if both structures contain the same total weight and the ground distance is a true metric. Structures imported from TPS data have hundreds to thousands of points and will almost

certainly contain different total weights, whether the weight of each point is set to the gradient of the dose grid, the magnitude of the dose, or 1. Comparing structures with different total weight would lead to many points on the larger structure failing to receive supplies and the EMD would be underestimated. To solve this, points would need to be sampled within the volume or between the vertices of the smaller structure until the total weights for both structures are equal. In some cases, it may be more accurate if both structures sample points within their vertices and the EMD is calculated until the result converges.

A problem with the traditional calculation of the EMD in the context of interest is that the only information in calculating the flow or EMD between two points is the weights at each point and the distance between them. If two points are located in two regions of 10 Gy of radiation separated by 1m with a large valley or peak in radiation dose between them, the EMD only takes into account the 10 Gy of radiation and the 1m separation. The behavior of the dose grid in between the points is ignored. Comparing structures for similarity requires knowing about the dose behavior, especially in the area of discrepancy. To account for this, the definition of the EMD can be changed to calculate the line integral of the radiation dose between the points. The flows could be calculated by setting the weights of each point equal to 1, the gradient of the dose, or the dose. After points are paired, the line integral between each point is evaluated and averaged to calculate a modified weighted Earth Mover's Distance. This value would no longer be a distance metric but would include information about the physical distance and dose in between points and may be more meaningful than a traditional EMD in most radiation treatment planning contexts.

The primary limitation with this method in R is computation time. R is not designed for processing at this level. Calculating the EMD between structures with just 300 points required

over 24 hours of computation time. The additional sampling of points to allow the EMD to converge combined with the thousands of points long data sets from medical imaging would require weeks long computation for a single comparison. Development of this method was abandoned due to its unrealistic time requirements within the constraints of the project. Using a more memory efficient language could remove this limitation. Further research should apply the more traditional EMD calculation to structures that have been sampled to increase the number of points so that they match in total weight. Radiation dose gradient or magnitude could be used for weighting and results from this function and the mDSC function should be compared for performance benchmarking and applicability to radiation treatment planning data.

References

Baskar, R., Lee, K.A., Yeo, R., Yeoh, K.-W., 2012. Cancer and Radiation Therapy: Current Advances and Future Directions. Int J Med Sci 9, 193–199. https://doi.org/10.7150/ijms.3635

Boon, I.S., Au Yong, T.P.T., Boon, C.S., 2018. Assessing the Role of Artificial Intelligence (AI) in Clinical Oncology: Utility of Machine Learning in Radiotherapy Target Volume Delineation. Medicines (Basel) 5. https://doi.org/10.3390/medicines5040131

Dice, L.R., 1945. Measures of the Amount of Ecologic Association Between Species. Ecology 26, 297–302. https://doi.org/10.2307/1932409

Rubner, Y., Tomasi, C., Guibas, L.J., 2000. The Earth Mover's Distance as a Metric for Image Retrieval 20.

Thompson, R., 2014. RadOnc: An R Package for Analysis of Dose-Volume Histogram and Three-Dimensional Structural Data. Journal Of Radiation Oncology Informatics 6, 98–110. https://doi.org/10.5166/jroi-6-1-25

Appendix (Code begins on next page)

```
compareStructures <- function(structures, method=NULL,
hausdorff.method=NULL, verbose=TRUE, plot=TRUE, pixels=100,
dose) {
  if (class(structures) != "structure.list") {
    warning("Input 'structures' must be of class
    'structure.list'")
    return()
  }
  empty <- unlist(lapply(structures, function(struct)
  {return(dim(struct)[1] <= 0)}))
  if (any(empty)) {
    warning(paste("Skipping empty structure(s): ",
    paste(names(structures[empty]), collapse=", ", sep=""),
    sep=""))
    structures <- structures[!empty]
  }
  N <- length(structures)
  if (N < 2) {
    warning("Need at least 2 structures to perform
    comparison")
    return()
  }
  method <- match.arg(method, choices=c("axial", "surface",
  "hausdorff", "grid", "DSC", "wDSC"))
  switch(method,
         DSC = {
           contours <- compareStructures.axial(structures,
           pixels=pixels)
           N <- dim(contours)[2]-3
           results <- matrix(0, nrow=N, ncol=N,
           dimnames=list(names(structures), names(structures)))
           for (i in 1:N) {
             for (j in 1:N) {
               if (i == j) {
                 results[i, j] <- 1#
                 next
               }
               results[i, j] <- 2*sum((contours[,i+3]>0) &
                  (contours[,j+3]>0))/(sum(contours[,i+3]>0)+
                  sum(contours[,j+3]>0))
             }
           }
           return(results)
         },
         wDSC = return(compareStructures.wDSC(structures,
         dose, pixels))
  )
```

```
compareStructures.axial <- function (structures, pixels=100)
{
  N <- length(structures)
  z <- as.list(rep(NA, N))
  bounds <- range(structures, na.rm=TRUE)
  x.coords <- seq(from=bounds[1,1], to=bounds[2,1],
  length.out=pixels)
  y.coords <- seq(from=bounds[1,2], to=bounds[2,2],
  length.out=pixels)
  for (i in 1:N) {
    if (length(structures[[i]]$vertices) < 1) {
      next
    }
    z[[i]] <- unlist(lapply(structures[[i]]$closed.polys,
    function(closed.poly) {return(unique(closed.poly[,3]))}))
  }
  z.coords <- unique(unlist(z))
  pts <-
  matrix(nrow=length(x.coords)*length(y.coords)*length(z.coords
  ), ncol=3, dimnames=list(NULL, c("X", "Y", "Z")))
  pts <- matrix(c(rep(x.coords,
  each=length(y.coords)*length(z.coords)), rep(rep(y.coords,
  each=length(z.coords)), length(x.coords)), rep(z.coords,
  length(x.coords)*length(y.coords))),
  nrow=length(x.coords)*length(y.coords)*length(z.coords),
  ncol=3, dimnames=list(NULL, c("X", "Y", "Z")))
  results <- matrix(0, nrow=dim(pts)[1], ncol=N,
  dimnames=list(NULL, names(structures)))
  for (i in 1:N) {
    for (j in unique(z[[i]])) {
      pts.j <- pts[which(pts[, 3]== j), 1:2]
      results.j <- rep(0, dim(pts.j)[1])
      z.j <- which(z[[i]] == j)
      ## THIS LOOP ACCOUNTS FOR AXIAL SLICES WITH MULTIPLE
      SEPARATE CLOSED POLYGONS (e.g. 3 ROOTS FOR SINGLE TOOTH)
      ## IF CLOSED POLYGONS ARE NESTED, THEY WILL BE
      INTERPRETED AS HOLES, SUCH THAT POINTS BETWEEN TWO POLYGONS
      MAY BE INTERPRETED AS EXTERIOR TO THE POLYGONS THEMSELVES
      (NOTE THAT THIS ASSUMES THE POLYGONS DO NOT CROSS EACH OTHER
      AT ANY POINT)
      for (k in 1:length(z.j)) {
        results.j <- results.j +
        as.numeric(pointInPoly2D(pts.j[,1:2],
        structures[[i]]$closed.polys[[z.j[k]]][,1:2]))
      }
      results[which(pts[, 3]== j), i] <- results[which(pts[,
```

```
      3]== j), i]+(results.j %% 2 != 0)
    }
  }
  return(cbind(pts, results))
}


compareStructures.wDSC <- function (structures, dose, pixels
= 100){
  print("Testing if Voxels Contain Any Structures")
  N <- length(structures) # how many structures are there
  bounds <- range(structures, na.rm=TRUE)
  voxelDimensions <- c((bounds[2,1]-bounds[1,1])/(pixels-1),
  (bounds[2,2]-bounds[1,2])/(pixels-1),
  (bounds[2,3]-bounds[1,3])/(pixels-1))
  halfVoxels <- voxelDimensions*1/2 #defines distance from
  midpoint where dose will be approximated
  x.coords <- seq(from=bounds[1,1], to=bounds[2,1],
  length.out=pixels)
  y.coords <- seq(from=bounds[1,2], to=bounds[2,2],
  length.out=pixels)
  toSelectFrom <- compareStructures.axial(structures, pixels)
  columns <- length(structures) + 3 #allows function to work
  with any number of structures
  toFill <- matrix(rep(NA, times =
  columns*nrow(toSelectFrom)), ncol= columns) #this will be
  filled with rows from toSelectFrom that are in at least one
  structure, saves a lot of computation time
  print("Selecting Voxels Containing Structures")
  for (p in 1:nrow(toSelectFrom)) {
    iszero <- c(0)
    for (u in 4:columns) {
      iszero[u] <- toSelectFrom[p,u] == 0
    }
    if(sum(iszero, na.rm = TRUE) == length(structures)){
    #iszero = true when row contains coordintes not in any
    structure, so we leave this row in toFill with NAs
      next()
    }else{
      for (s in 1:columns) {
        toFill[[p,s]] <- toSelectFrom[[p,s]] #if iszero >0,
        then the row contains a coordinate in at least one structure
        and needs to be used in calculating the wDSC
      }
    }
  }
  newMat <- toFill[apply(toFill, 1,
```

```
function(x)!any(is.na(x))), , drop=F] #removes rows with NAs
(that weren't in any structure)
xStep <- halfVoxels[1]  # distance from voxel midpoint
where dose will be aproximated
yStep <- halfVoxels[2]
zStep <- halfVoxels[3]
magnitudesMatrix <- matrix(0, ncol = 1, nrow =
nrow(newMat)) #this will be be the magnitudes of the net
radiation dose gradient vector
dhms <- function(t){  #this function will estimate time
that the calculation will take
  paste(t %/% (60*60*24)
        ,paste(formatC(t %/% (60*60) %% 24, width = 2,
        format = "d", flag = "0")
               ,formatC(t %/% 60 %% 60, width = 2, format =
               "d", flag = "0")
               ,formatC(t %% 60, width = 2, format = "d",
               flag = "0")
               ,sep = ":"
        )
  )
}
ETAseconds <- nrow(newMat)/150
print("Calculating Radiation Dose Gradient At All Voxels")
print("Estimated time for full weighted DSC calculation in
D H:M:S")
print(dhms(ETAseconds))
pb <- txtProgressBar(min = 0, max = nrow(newMat), style =
3)
for (q in 1:nrow(newMat)) {
  setTxtProgressBar(pb, q)
  midpointX <- newMat[q,1]   #this is the x, y, z
  coordinate of the voxel center, we will move in halfVoxels
  steps in all 14 directions
  midpointY <- newMat[q,2]
  midpointZ <- newMat[q,3]
  doseAtMidpoint <- approx3D(data = dose, x = midpointX , y
  = midpointY, z = midpointZ)
  #the below vectorMatrix is a matrix that has columns i,
  j, k, deltaDose. The first three represent the direction of
  the vector and the deltaDose is the difference in radiation
  dose at the end of that vector and the midpoint of the voxel
  vectorMatrix <- matrix(data =
  c(0,0,0,0,-1,1,-1,-1,-1,-1,1,1,1,1,  #i column
     0,0,1,-1,0,0,1,-1,1,-1,1,-1,1,-1,  #j column
     1,-1,0,0,0,0,1,1,-1,-1,1,1,-1,-1,  #k column
     approx3D(data = dose, x = midpointX, y = midpointY, z
```

```
    = midpointZ + zStep) - doseAtMidpoint, #up
    approx3D(data = dose, x = midpointX, y = midpointY, z
    = midpointZ - zStep) - doseAtMidpoint, #down
    approx3D(data = dose, x = midpointX, y = midpointY +
    yStep, z = midpointZ) - doseAtMidpoint, #right
    approx3D(data = dose, x = midpointX, y = midpointY -
    yStep, z = midpointZ) - doseAtMidpoint, #left
    approx3D(data = dose, x = midpointX - xStep, y =
    midpointY, z = midpointZ) - doseAtMidpoint, #forward
    approx3D(data = dose, x = midpointX + xStep, y =
    midpointY, z = midpointZ) - doseAtMidpoint, #backward
    approx3D(data = dose, x = midpointX - xStep, y =
    midpointY + yStep, z = midpointZ + zStep) - doseAtMidpoint,
    #forward up right
    approx3D(data = dose, x = midpointX - xStep, y =
    midpointY - yStep, z = midpointZ + zStep) - doseAtMidpoint,
    #forward up left
    approx3D(data = dose, x = midpointX - xStep, y =
    midpointY + yStep, z = midpointZ - zStep) - doseAtMidpoint,
    #forward down right
    approx3D(data = dose, x = midpointX - xStep, y =
    midpointY - yStep, z = midpointZ - zStep) - doseAtMidpoint,
    #forward down left
    approx3D(data = dose, x = midpointX + xStep, y =
    midpointY + yStep, z = midpointZ + zStep) - doseAtMidpoint,
    #backward up right
    approx3D(data = dose, x = midpointX + xStep, y =
    midpointY - yStep, z = midpointZ + zStep) - doseAtMidpoint,
    #backward up left
    approx3D(data = dose, x = midpointX + xStep, y =
    midpointY + yStep, z = midpointZ - zStep) - doseAtMidpoint,
    #backward down right
    approx3D(data = dose, x = midpointX + xStep, y =
    midpointY - yStep, z = midpointZ - zStep) - doseAtMidpoint
    #backward down left
),

ncol = 4)
for (w in 1:14) { #in this loop, the i, j, and k columns
are multiplied by the deltaDose
  vectorMatrix[w, 1] <- vectorMatrix[w, 1] *
  vectorMatrix[w, 4]
  vectorMatrix[w, 2] <- vectorMatrix[w, 2] *
  vectorMatrix[w, 4]
  vectorMatrix[w, 3] <- vectorMatrix[w, 3] *
  vectorMatrix[w, 4]
}
```

```
    netVector <- colSums(vectorMatrix[,c(1,2,3)]) #the
    columns are summed to get the net vector, the mangnitude of
    the vector is calculated on the next line
    magnitudesMatrix[q,] <- sqrt(netVector[1]^2 +
    netVector[2]^2 + netVector[3]^2)
  }
  close(pb)
  multipliedMatrix <- newMat[ , c(4:columns)] *
  c(magnitudesMatrix) #takes the matrix with 1 and 0
  representing whether or not a voxel belongs to a structure
  and multiplies the 1 by the dose gradient magnitude
  DSCtable <- matrix(0, nrow = columns - 3, ncol = columns -
  3, dimnames=list(names(structures), names(structures)))
  for (i in 1:(columns-3)) {
    for (j in 1:(columns-3)) {
      if (i == j) {
        DSCtable[i, j] <- 1      #DSC for identical
        structures is 1
        next()
      }
      alpha <- colSums(multipliedMatrix)[i] #represents |A|
      sum all gradient magnitudes for structure A
      beta <-  colSums(multipliedMatrix)[j] #represents |B|
      sum all gradient magnitudes for structure B
      gamma <- matrix(0, ncol = 1, nrow =
      nrow(multipliedMatrix))
      for (k in 1:nrow(multipliedMatrix)) {
        if(multipliedMatrix[k,i] == multipliedMatrix[k,j]){
          gamma[k] <- multipliedMatrix[k,i] #if voxel is in
          both structures, add the gradient magnitude to gamma
        }else{
          gamma[k] <- 0
        }
      }
      gamma <- sum(gamma, na.rm = TRUE) #represnts |A ∩ B|
      DSCtable[i, j] <- 2*gamma/(alpha+beta)
    }
  }
  print(DSCtable)
}


pointInPoly2D <- function (points, poly) {
  poly <- matrix(unique(poly), ncol=2)
  n <- dim(poly)[1]
  x <- diff(poly[c(1:n,1),1])
  y <- poly[,2] + poly[c(2:n,1),2]
```

```
  if (sum(x*y/2) >= 0) {
    #clockwise poly
    return(pip2d(poly[n:1,], points) >= 0)
  }
  else {
    #anti-clockwise poly
    return(pip2d(poly, points) >= 0)
  }
}
```