Hole or grain? A Section Pursuit Index for Finding Hidden Structure in Multiple Dimensions

Ursula Laa[a,b,c], Dianne Cook[a], Andreas Buja[d], German Valencia[b]

[a]Department of Econometrics and Business Statistics, Monash University; [b]School of Physics and Astronomy, Monash University; [c]Institute of Statistics, University of Natural Resources and Life Sciences; [d]Statistics Department, The Wharton School, University of Pennsylvania

ABSTRACT

Multivariate data is often visualized using linear projections, produced by techniques such as principal component analysis, linear discriminant analysis, and projection pursuit. A problem with projections is that they obscure low and high density regions near the center of the distribution. Sections, or slices, can help to reveal them. This paper develops a section pursuit method, building on the extensive work in projection pursuit, to search for interesting slices of the data. Linear projections are used to define sections of the parameter space, and to calculate interestingness by comparing the distribution of observations, inside and outside a section. By optimizing this index, it is possible to reveal features such as holes (low density) or grains (high density). The optimization is incorporated into a guided tour so that the search for structure can be dynamic. The approach can be useful for problems when data distributions depart from uniform or normal, as in visually exploring nonlinear manifolds, and functions in multivariate space. Two applications of section pursuit are shown: exploring decision boundaries from classification models, and exploring subspaces induced by complex inequality conditions from a multiple parameter model. The new methods are available in R, in the `tourr` package.

CONTACT Ursula Laa. Email: `ursula.laa@boku.ac.at`, Dianne Cook. Email: `dicook@monash.edu`, Andreas Buja. Email: `andreasbuja@gmail.com`, German Valencia. Email: `german.valencia@monash.edu`

data visualization, exploratory data analysis, data science

Contents

1. Introduction

The visualization of high-dimensional data often utilizes linear projection. For example, in principal component analysis plotting the principal components is effectively a projection of the data. Projections form the basis for a grand tour (Asimov 1985) of high-dimensional data. Projection pursuit (Kruskal 1969, Friedman and Tukey (1974)) is used to find interesting low-dimensional views of the data by optimizing an index function over all possible projections. Projections can obscure non-uniform patterns near the data center, or hollowness. These features may be visible in non-linear mappings (such as multidimensional scaling (Kruskal and Wish 1978)), but these methods often lack interpretability. Laa, Cook, and Valencia (2019) introduced a slice tour that shows sections through high-dimensions instead of projections, and helps to reveal hidden structure. Showing sections can also be useful in combination with conditional function or model visualization (O'Connell, Hurley, and Domijan 2017, Torsney-Weir, Sedlmair, and Möller (2017)).

The space of sections of high-dimensional space is larger than the space of projections. The slice tour (Laa, Cook, and Valencia 2019) introduces a special type of section, based on projections and the grand tour, thus simplifying the space to be explored. This type of section is called a slice, and we can visually compare the projected distributions inside and outside the slice. This is an unguided process, and providing a method to find interesting slices would be beneficial.

Here we propose "section pursuit", which searches the space of projections for the most interesting slices of the data. The "interestingness" of each slice is computed as a measure of dissimilarity between the distribution of projected points inside and outside the slice. Finding conditional features in large data sets will provide better understanding of the data, and can help improve modelling.

This paper is organized as follows. The next section provides background on tours, slicing and projection pursuit. Section 3 describes an index designed to detect concavities, and Section 4 discusses the associated practical considerations and explores the behavior of the index on simulated examples. The application of section pursuit to exploring high-dimensional classification models and constrained high-dimensional

spaces is illustrateded in Section 5.

2. Background

2.1. *Grand tour and slice tour*

The grand tour shows a geodesically interpolated sequence of randomly selected projections in an animation. The interpolation allows the viewer to interpret each view in the context of the previously seen projection, thus providing additional insights compared to static projections. By observing how the data distribution changes under a rotation of the low-dimensional projection, the viewer can extrapolate from the low-dimensional shapes to the distribution in high dimensions. The underlying computational methods were described in Buja et al. (2005), and an implementation in R (R Core Team 2018) is available in the `tourr` package (Wickham et al. 2011).

The information available in projections can be complemented using *sectioning* Furnas and Buja (1994) , i.e. highlighting or dropping subsets of the data based on inequality conditions defined for the variables. A special type of sectioning is *slicing*, which is defining the sectioning condition based on the orthogonal distance of data points from a projection plane. Points are considered to fall into the slice if this distance is below some cutoff value h. Figure 1 shows diagrams illustrating slicing through a 3D sphere (left) and the general concept of the orthogonal distance (right) that is used to define slices in more than three dimensions. Recent work in Laa, Cook, and Valencia (2019) implemented a display of interpolated slices, which has been added to the `tourr` package and can be used to show slices for projection planes obtained when running a grand tour.

A slice is defined using Euclidean distance in the orthogonal space to ensure rotation invariance. This means that in general slices are spherical, and a *flat slice*, matching the common picture of a slice (as shown in the left plot of Fig. 1), is only obtained in the case of a single orthogonal direction on the projection plane.

The first two columns of Figure 2 show scatterplots of the points in a 4D data set inside an informative (S1) and an uninformative (S2) slice, defined by two different projection planes centered at the origin, for two simulated data sets (A, B). Note that
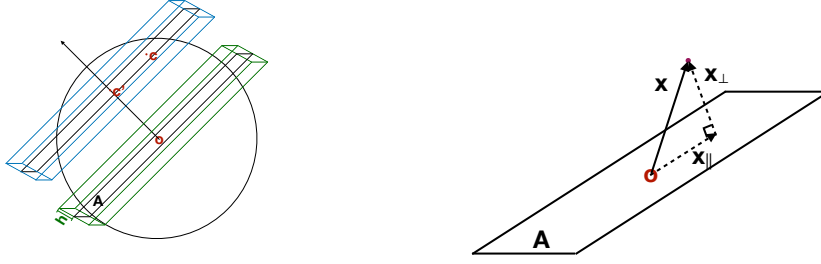
4

Figure 1. Illustrations of slicing, through a 3D sphere (left), and demonstrating the calculation of the orthogonal distance (right). A slice can be done centered at the origin (green) or off-center at **c** (blue). In more than three dimensions it is best to use orthogonal distance between point and projection plane for computing the slice. Diagrams adapted from Laa, Cook, and Valencia (2019).
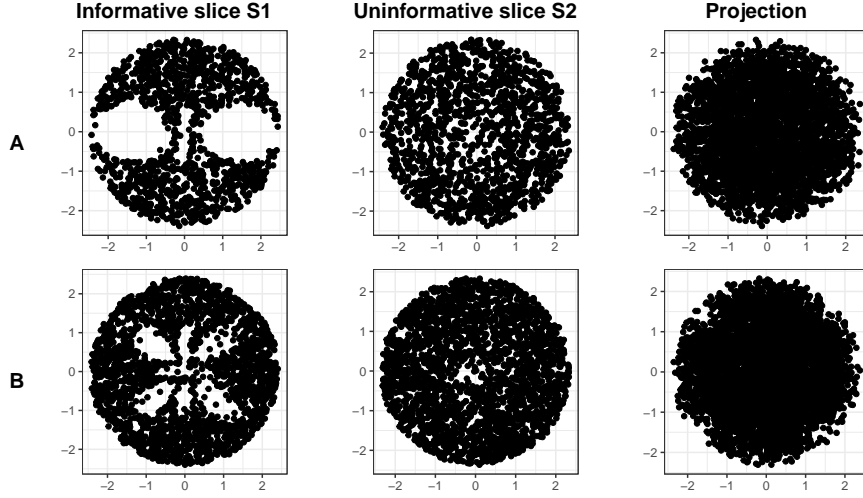

Figure 2. Slices of two 4D data sets (A, B) shown using scatterplots of points inside the slice. The first slice (S1) is informative and reveals the hollow regions in the data, while the second slice (S2) is un-informative. The last column shows projections of the data onto a 2D plane, hiding the hollowness.

the data is defined in a 4D sphere, which generates the circular shape. The sampling results in hollow regions inside the sphere by rejecting points within shapes defined by selected hyperspherical harmonics (Meremianin 2009), as described in the Appendix. The hollowness is hidden in projections of the data, as shown in the final column of Figure 2. It is much simpler to define slices on high-dimensional spheres than cubes, and this is the approach used for all the data in this paper. Data defined in a cube is trimmed or supplemented to be contained in a sphere.

2.2. *Projection pursuit and guided tour*

Projection pursuit is the procedure of selecting interesting low-dimensional projections by optimizing a criterion function over the space of all possible projections. The criterion (or index) function is typically designed to take larger values for more "interesting" views of the data, and maximizing the index can reveal structure in the distribution.

The concepts of projection pursuit and the grand tour can be combined into the guided tour (Cook et al. 1995), which interpolates between projection planes selected through the optimization of a projection pursuit index. A guided tour presents the viewer with interesting views of the dataset in the context of the full distribution, moving from less to more informative projections.

Tour methods can also be used to understand the behavior of projection pursuit index functions (Laa and Cook 2020), for example by examining how the index value changes along an interpolated sequence of projections.

3. A new index for finding interesting sections

We define a new index function that is comparing the distribution of points inside a slice to the distribution outside the slice. This is achieved by comparing normalized counts in bins across the projection plane to find differences between the distributions. We first introduce the notation for slicing and binning (3.1), followed by the definition of the index function itself (3.2). In practice we need to make assumptions about the underlying distribution to get useful results when comparing the slice to a projection. These aspects are discussed in the next section.

3.1. *Taking a slice and binning*

Let $Y = X \cdot A$, where X is an $n \times p$ data matrix, A is a $p \times d$ (orthonormal) basis for the d-dimensional space onto which the data is being projected. To generate a 2-dimensional slice, following Laa, Cook, and Valencia (2019), compute the orthogonal distance between every point and the plane (defined by $A = (\mathbf{a}_1, \mathbf{a}_2)$) as the Euclidean

norm

$$h_i = ||\mathbf{x}_i - (\mathbf{x}_i \cdot \mathbf{a}_1)\mathbf{a}_1 - (\mathbf{x}_i \cdot \mathbf{a}_2)\mathbf{a}_2||. \tag{1}$$

Observations are considered inside the slice if $h_i < h$.

We denote the set of points inside the slice S, and the set of points outside the slice C. For the definition of the index we will further bin the projected data Y into K bins, $b_k, k = 1, ..., K$. The counts of observations for each bin are computed separately for S and C. We can thus write the counts in bin k via two indicator functions, $S_k = \sum_i I(Y_i \in b_k)I(h_i < h)$ inside the slice and $C_k = \sum_i I(Y_i \in b_k)I(h_i \geq h)$ outside the slice. The relative counts are thus $s_k = S_k / \sum_i S_i$ and $c_k = C_k / \sum_i C_i$.

Data points are therefore classified in two ways: information about the projected position onto the viewing plane is captured by the binning, while information about the position in the orthogonal space is captured by slicing. In practice the first step is projecting and slicing, followed by a separate binning of the projected points Y for the subsets S and C.

3.2. *Index definition*

Indexes for comparing the distribution of observations in the slice versus outside can be defined (building on Gous and Buja (2004)) as:

$$I_A^{low} = \sum_k [(c_k - s_k)]_{>\varepsilon}, \tag{2}$$

$$I_A^{up} = \sum_k [(s_k - c_k)]_{>\varepsilon} \tag{3}$$

where s_k and c_k are the relative counts of observations in bin b_k inside and outside the slice. The first definition I_A^{low} takes large values if there is a hollow region of low

density inside the slice, while the second definition I_A^{up} can be useful when looking for regions of higher density ("needles in a haystack"). The $[.]_{>\varepsilon}$ notation indicates that we drop all bins where the difference in counts is below some threshold ε,

$$[a - b]_{>\varepsilon} = \begin{cases} a - b, & \text{if } a - b > \varepsilon \\ 0, & \text{if } a - b \leq \varepsilon. \end{cases} \tag{4}$$

Only bins where the difference is positive are summed. The ε avoids summing noise and suppresses an artificial dependence on the number of bins. The value of ε should be estimated based on the expected size of sampling fluctuation, which may depend on assuming a distribution, the number of samples N, the number of bins K, the slice radius h and the dimension p. An estimate for ε assuming a uniform distribution inside a hypersphere is given in Section 4.4. If $\varepsilon = 0$ we get a symmetry in the definitions, $I_A^{low} = I_A^{up}$. In general, the choice of index should match the intention, searching for regions of high or low density in the slice.

Generalised index

The definitions in Eq.(2,3) can be generalized for example to emphasise selected bins or control the sensitivity.

$$I_A^{low} = \sum_k w_k \left(\left[c_k^{1/q} - s_k^{1/q} \right]_{>\varepsilon} \right)^q \tag{5}$$

$$I_A^{up} = \sum_k w_k \left(\left[s_k^{1/q} - c_k^{1/q} \right]_{>\varepsilon} \right)^q. \tag{6}$$

Here w_k can be used to (de)emphasize certain bins, e.g. to up-weight information in the center of the distribution. The exponent q can be used to tune the sensitivity, a small q will enhance sensitivity to small differences (and thus might over-emphasize fluctuations), a large q will suppress fluctuations and is mainly sensitive to large differences (and might thus miss features that are too similar to the background distribution). Selecting $q = 1$ corresponds to an L_1 type norm, and $q = 2$ to an L_2 type

norm. In addition we will also consider $q = 1/2$.

The overall range of plausible index values depends on q and in practice we use an estimate of the range to rescale the index value to fall in $[0, 1]$. Notice that, in this definition the index is no longer symmetric under exchanging the distributions inside and outside the section, even when $\varepsilon = 0$.

4. Practical issues

The application of the new index function should be informed by an expected underlying distribution. As explained in 4.1 these should be rotation invariant, and the concrete assumption can guide practical choices that need to be made: reweighting bin counts to account for expected differences between the projected distribution and the distribution in a slice (4.2), estimates of sufficient sample size (4.3), expected noise and corresponding threshold ϵ (4.4). We also include visualizations to better understand the index behavior and how it depends on its parameters (4.5) and guidance on using the index in practice (4.6).

4.1. *Rotation invariance*

A desirable property for projection pursuit, into d-dimensions, and thus also desirable for section pursuit, is that the index be rotational invariant. That is, regardless of the basis in the d-D plane defining the projection, the index value should be identical. With slices, this is more complicated, because interior and exterior distributions need to be comparable in the absence of structure. To account for expected differences we need to choose an assumed (spherically symmetric) distribution of the data.

Thus a restriction is imposed on the observed data: that it falls within a p-dimensional hypersphere. Data is typically observed in a hypercube, so this prescription requires a departure from the norm which, however,is still practical. In cases of simulated data, such as when looking at multivariate models, the sampling scheme can be adjusted. Observed data, on the other hand, requires shaving off the corners. Note here that most data has a tendency to be elliptical or spherical, and that we are interested in interior structures for which the corners are not important. By sphering
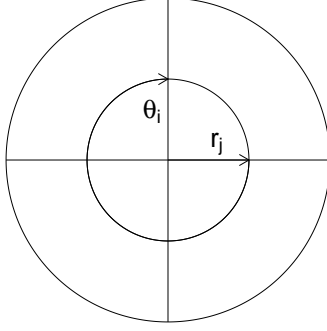
Figure 3. Diagram illustrating the notation for polar binning in angle θ and radius r. Angular bins $\theta_i, i = 1, 2, 3, 4$, split the angular component into four bins, $r_j, j = 1, 2$ splits the radial component into two bins, resulting in a total of 8 bins.

the data during pre-processing, shaving off any remaining corners can be kept to a minimum. In addition, this approach is designed for reasonably small p so that the vast gap between spheres and cubes in high dimensions is not a concern. For very high-dimensional data, some dimension reduction is expected in pre-processing.

Here we will work with the assumption of a uniform distribution inside a hypersphere because it naturally captures how the relative slice volume changes with dimensionality. Other rotation invariant distributions could also be of interest, for example a multivariate normal distribution.

4.2. *Polar binning and reweighting*

Requiring a rotation invariant range suggests a preference for binning in polar coordinates. In general we can decompose an expected underlying distribution in a radial and a directional component, the latter being parametrized by $d - 1$ angles. The expected reference distribution can then guide the concrete choices in how to bin in these parameters.

In our case we are interested in binning the projected data points in $d = 2$ dimensions and will use the radius r and a single angle θ in the projection plane to parametrize the binning, as illustrated in the diagram in Fig. 3. As a reference distribution we consider points that are uniformly distributed in a hypersphere, thus the angular distribution will be uniform across all values of θ, suggesting the use of K_θ equidistant angular bins. This would also hold in the case of a spherical normal reference distribution.
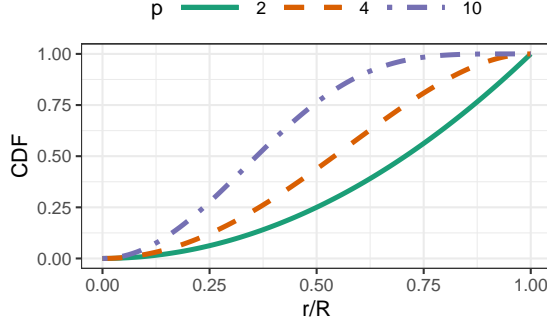
Figure 4. CDF of the 2D projected radial distribution of a p-dimensional hypersphere, for selected values of p. As p increases, a larger fraction of the points will be found at small values of the radius.

The radial binning is more complicated. This is because the marginal radial distribution of the projected data depends on the assumed reference distribution as well as the data dimenson p. As p increases, the projected data piles more in the center. A varying radial bin size could be used to offset this effect, where the bounds of the K_r radial bins take into account the expected distribution of points given p. However, the expected radial distribution will also differ between the projected data in a slice and a projection of the full data, so the effect is accounted for instead by reweighting bin counts. The aim is to adjust for the expected difference in distribution: after the reweighting, the expected counts in all radial bins is the same, and this needs to be adjusted separately for the data in the slice and the projected data. The calculations are as follows.

Consider the expected distribution to be a uniform distribution within a hypersphere in p dimensions projected onto a 2D plane. The cumulative distribution function (CDF) for the radial distribution in the projection, derived in the Appendix, is given by:

$$F(r; p, R) = 1 - \left(1 - \left(\frac{r}{R}\right)^2\right)^{p/2}, \tag{7}$$

and depends on the hypersphere radius R and on p. For illustration we show the CDF dependence on r/R (where $0 < r \leq R$) for different values of p in Fig. 4. For large values of p the majority of points is found within a small relative radius r/R, e.g. for $p = 10$ we see that 75% of points are within $r/R < 1/2$.

11

While the radial distribution of the points projected from a full p dimensional hypersphere follows the CDF in Eq.(7), the distribution in the slice is approximately uniform in the disk (as long as $h \ll R$). Within the slice, the adjustment only accounts for relative areas in radial bins for a 2D uniform distribution. All weights can be calculated from the CDF.

The fraction of points in the radial bin r_i with lower boundary r_i^l and upper boundary r_i^u is

$$F_i(p, R) = F(r_i^u; p, R) - F(r_i^l; p, R). \qquad (8)$$

Consider K_r radial bins, with total bin count N_i in bin i, $i = 1, ..., K_r$, and relative counts are $n_i = N_i / \sum_j N_j$. We define the bin-wise weights as

$$w_i(p, R, K_r) = \frac{1}{K_r F_i(p, R)}, \qquad (9)$$

and the reweighed bin count as

$$s_i(p, R, K_r) = n_i w_i(p, R, K_r) \qquad (10)$$

corresponding to the weighted relative number of points in each bin. After reweighting, the relative expected count in each radial bin is thus $1/K_r$ for a uniform hypersphere. We calculate the outside weights as $w_i(p, R, K_r)$ and the inside weights as $w_i(2, R, K_r)$. Recall that we are assuming that the slice thickness can be neglected, $h \ll R$.

Figure 5 illustrates the effect of the adjustment using polar histograms for the simulated example data B, with and without re-weighting. The rows show raw and weighted bin counts, respectively, and the columns contain plots of the distributions of points inside, outside the slice and the difference between the two. The re-weighting has the desired effect of focusing the attention more effectively on the central structure.

The index needs to be optimized over the set of all possible projections to find interesting concavities. This requires the binned data to be comparable, thus *the bins are fixed for all planes*. For each slice we first center the projected data before binning
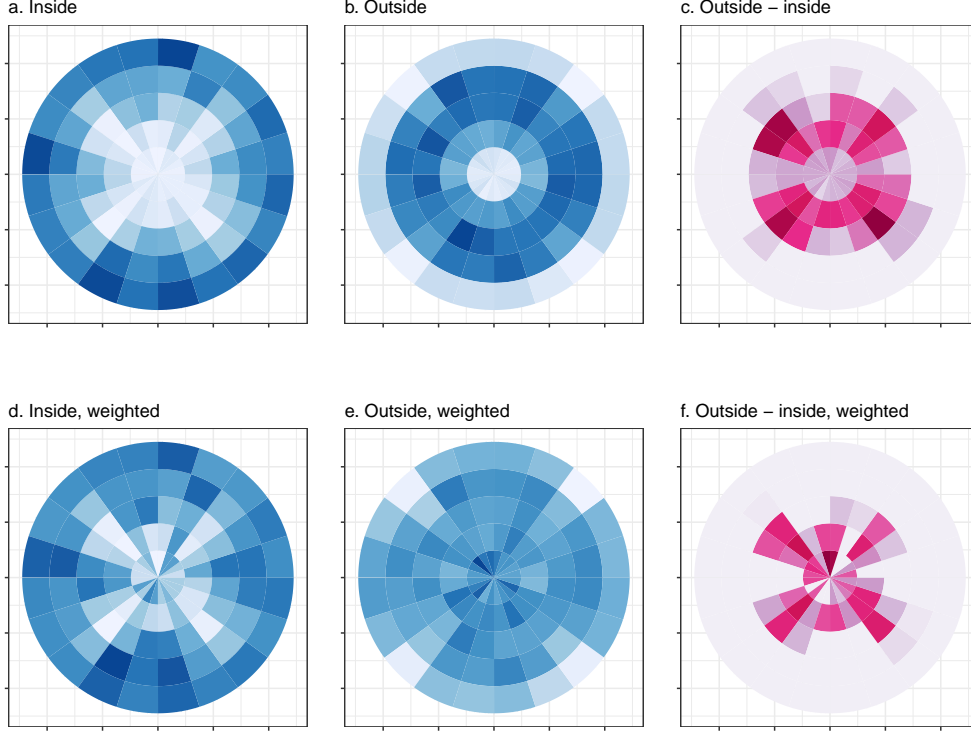
Figure 5. Illustration of the polar binning, effect of re-weighting bin counts in the index function computation, on sample data set B and for the informative slice S1. Rows show raw and weighted bin counts, respectively. Columns show the distributions of inside, outside and the difference. The effect of the re-weighting focuses the attention more effectively on the cavity structure.

it.

In the simulated examples in this paper the radius of the hypersphere, R, is known, and is used to define the radial boundary. In practice, the radius of a random projection can be used to estimate R. When observed points for any new projection, or slice, fall outside of the sphere, they might be grouped into the outer bin.

Note that, the fixed angular binning can marginally affect rotational invariance of the index. To mitigate this, combining index values from several small rotations within a single angular bin window might be done at each step.

4.3. *Sufficient sample size*

As dimensionality increases the number of sample points required to resolve features in a thin slice of the data increases exponentially. Care must be taken that the sample is large enough. We estimate the number of required sample points as a function of the chosen parameters, starting again from the hyperspherical distribution in p dimensions.

Following Laa, Cook, and Valencia (2019), given a sample of N points in p dimensions, distributed uniformly in a hypersphere of radius R, the number of points inside a slice through the center of the distribution is

$$N_S = \frac{N}{2} \left(\frac{h}{R} \right)^{p-2} \left(p - (p-2) \left(\frac{h}{R} \right)^2 \right).$$

(11)

The calculation is based on the relative volume of the slice, derived in the Appendix. We denote $x = h/R$ the resolution. It determines the minimum relative size of features that can be seen in a slice. When adding noise dimensions, in order to keep the resolution x fixed, the sample size needs to increase approximately as $x^{\Delta p}$, where Δp is the increase in number of dimensions.

As an explicit example consider the expected fraction of points N_S/N inside a slice of resolution $x = \frac{h}{R} = 0.1$. This fraction is about 0.15 or 15% when $p = 3$, and quickly drops to 0.02 when $p = 4$ and 0.002 when $p = 5$. Thus, even for moderate dimensionality p the original sample size needs to be large to have enough points to resolve features in a slice.

4.4. *Estimating the magnitude of noise, ε*

We can estimate the expected sampling variability based on N. The dominant uncertainty will be on the bin count inside the slice which typically will have much smaller statistics than the outside distribution (notice that this may not be true for bins at large radius which, depending on the bin size, can have a very low number of observations). We estimate the number of points in a bin i inside the slice as

$$N_S^i = \frac{N_S}{K_\theta} \cdot F_i(2, R),$$

(12)

with N_S given by Eq.(11), K_θ the number of angular bins and F_i defined in Eq.(8). The relative Poisson error on this count is

$$\delta_S^i = \frac{\sqrt{N_S^i}}{N_S^i} = \frac{R}{\sqrt{(r_i^u)^2 - (r_i^l)^2}} \sqrt{\frac{2K_\theta}{N}} x^{(2-p)/2} \frac{1}{\sqrt{(p - (p-2)x^2)}}, \qquad (13)$$

where r_i^l and r_i^u are the lower and upper radial boundary of bin i and the expected error will be different depending on the radial position of the bin. By its definition, the expected reweighted count is $1/K$ in all bins, where $K = K_r K_\theta$. We therefore expect sampling fluctuations of order

$$\delta^i = \delta_S^i / K. \qquad (14)$$

To suppress index fluctuations to below one standard deviation, we set a bin-wise $\varepsilon^i = \delta^i$. Figure 6 compares the index behavior for two values of ε^i, 0 and δ^i. Three 4D data sets are used: examples A and B (shown in Figure 2), and a reference set C consisting of observations sampled uniformly within a p-dimensional sphere. Color indicates the number of bins used to calculate the index. The difference in index value when $\varepsilon^i = 0$ for all bins clearly indicates a dependence on the number of bins, which is undesirable. Setting $\varepsilon^i = \delta^i$ mostly removes these differences.

Dependence on q

In the generalized definition of the index, the parameter q can be used to tune the sensitivity to small differences. It will therefore also need to be considered when choosing ε^i. To understand the effect we consider a simple one-dimensional setup: uniformly draw N samples and bin them in K bins, such that the expected bin count is N/K in all bins (or $1/K$ after normalization). Using the Poisson approximation, the standard deviation for the normalized counts is $\delta = 1/\sqrt{NK}$. By drawing two independent samples and calculating the index value 100 times we get an estimate of the expected index value and its variance depending on q. Note that since we are assuming pure noise distributions for both samples, $I_A^{low} = I_A^{up} = I_A$. This is shown in Figure 7 for $N = 10000$ and $K = 100$. We see that for values of $q < 1$ the noise becomes
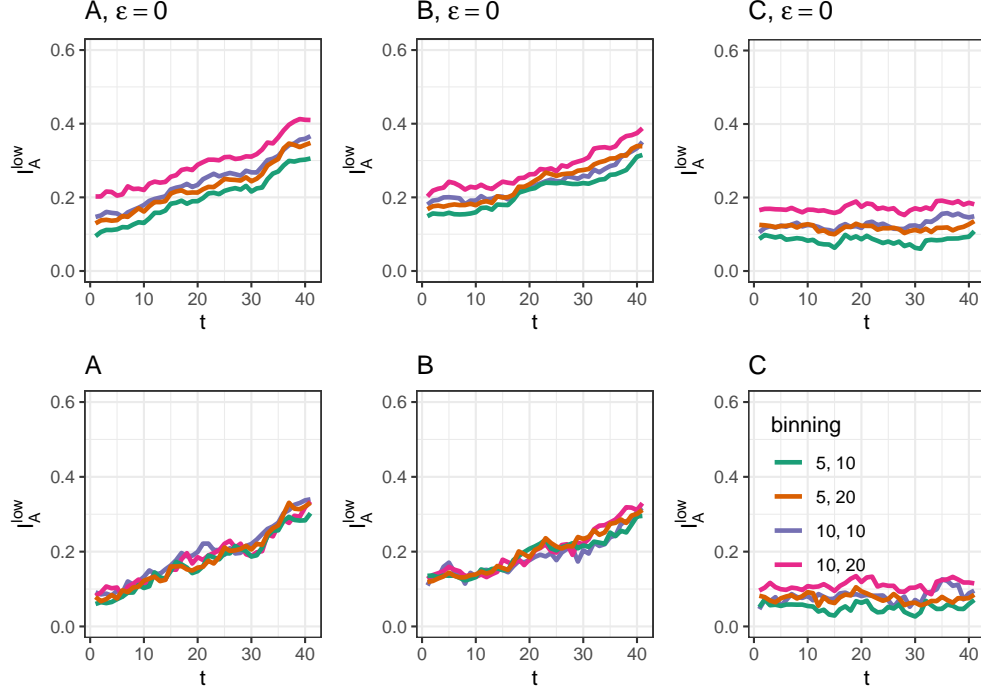
15

Figure 6. Examining the effect of the noise cutoff ε^i on the index value for varying number of bins. Three simulated data sets are used: A, B and C. The horizontal axis traces the index from an uninteresting to interesting sliced projection (defined by slice S2 and S1 respectively), for $\varepsilon^i = 0, \delta^i$. Colour indicates number of bins. Ideally, the index value should be the same, regardless of choice of bins, which is achieved in practice by $\varepsilon^i = \delta^i$.

inflated, especially when we do not use an appropriate ε cutoff. For the default choices of $q = 1$ and $\varepsilon = \delta$ we find that the expected index value for a noise distribution is about 0.05. Note here that in this simplified setup there is no bin-dependence of ε and we have dropped the index i.

4.5. *Index behavior*

To examine the behavior of the generalized index for choices of q, K and ε, a re-parametrization is used. Consider an outside distribution that is uniform across all K bins,

$$c_k = \frac{1}{K}, \tag{15}$$

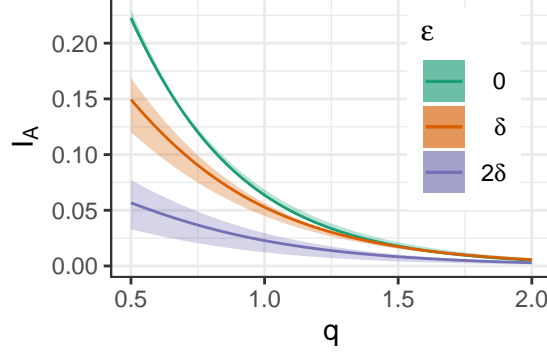**Figure 7.** Examining the effect of the power parameter q on the variability of the index value, when a noise cutoff (ε) is used. Rescaled values of I_A shown as a function of q, for fixed values of ε, for $N = 10000$ and $K = 100$. Smaller values of q inflate the variability in the index, exaggerating the noise.

and a distribution inside the slice to be uniform across $K' < K$ bins,

$$
s_k = \begin{cases} \frac{1}{K'}, & \text{if } k \leq K' \\ 0, & \text{if } k > K' \end{cases} \tag{16}
$$

i.e. there are $K - K'$ empty bins inside the slice. We can then parametrize the relation between K and K' as $K' = \gamma K$ with $\gamma \in [0,1]$.

To focus on the dependence on q, drop the bin-dependent weights w_k from the description and set $\varepsilon = 0$. Using the parametrization in terms of γ, the index can then be written as

$$
I_A^{up}(q,\gamma) = ([1 - \gamma^{1/q}]_+)^q, \tag{17}
$$

while

$$
I_A^{low}(\gamma) = 1 - \gamma. \tag{18}
$$

(In this latter form the dependence on q drops out because only bins where $s_k = 0$ are counted.)

This allows us to estimate the typical range depending on q. The range of $I_A^{up}(q,\gamma)$ will be different for different values of q, and this range difference needs to be ignored
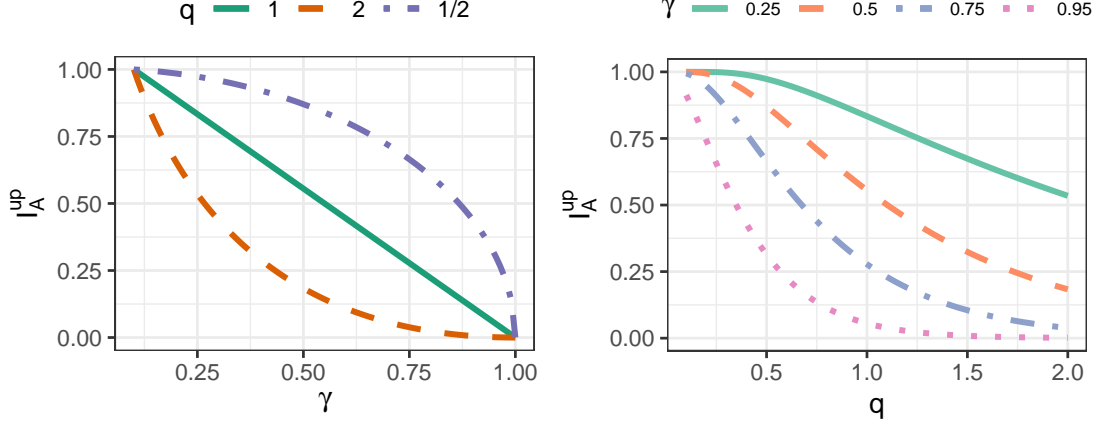
17

Figure 8. Examining the sensitivity of the index to the extcolorredpower parameter q. Rescaled values of I_A^{up} as a function of γ, extcolorredthe fraction of non-empty bins, for fixed values of q (left), and as a function of q for fixed values of γ (right). The parameter q shifts emphasis between small regions and large regions of difference.

for comparison purposes. For $q = 1$ the index range is $[0, 0.9]$. For $q = 2$ the index range is approximately $[0, 0.5]$. We rescale the index depending on the selected value of q against $I_A^{up}(q, \gamma = 0.1)$, resulting in a common range $[0, 1]$.

Fig. 8 shows the effect of q for $\gamma \in [0.1, 1]$, i.e. between 10% and 100% overlap in the distributions, and the effect of γ for a range of $q \in [0.1, 1]$. When $q = 1$ the index is linearly dependent on γ. For $q = 2$ most of the sensitivity lies towards small values of γ. For $q = 1/2$ most of the sensitivity lies towards large values of γ. The parameter q enables the index to be made more sensitive to small regions of difference as opposed to large areas.

4.6. *Visualising the index*

A Huber plot (Huber 1985) (available in the R package, PPtreeViz (Lee 2018)) is a useful illustration of a 1D projection pursuit index. It shows the index values for a 2D distribution across all possible 1D projections. This idea is generalized for a 2D projection pursuit index, to illustrate the behavior over a higher dimensional dataset, and called a *topotrace plot*. The approach is:

1. Select a starting plane, one of particular interest, A_0.
2. Randomly generate a large number (m) of directions to move away from the starting plane, $A_i, i = 1, ..., m$.

3. Generate the geodesic interpolation with a fixed length or angle (α) in each direction, $A_{ij}, i = 1, ..., m; j = -\alpha, ..., 0, ..., \alpha$.

4. Calculate the index values along each path, $I_{A_{ij}}$.

5. Plot $I_{A_{ij}}$ against j, with a separate trace for each i.

The purpose of making these plots is to examine the nature of the function, in terms of local maxima and ridges, and also characteristics such as smoothness and squint angle (Laa and Cook 2020). The squint angle describes the distance from the optimal projection where a structure can be seen – if it is large then the function should be easier to optimize. If the optimal projection is known, then using this as the starting plane, provides views analogous to standing on top of the mountain and looking down in all directions. If a random starting plane is used, most likely this will be a low point from which to view mountains.

Fig. 9 shows these plots for the two example datasets A and B, with $\alpha = \pi/2$, $m = 100$ randomly selected directions. The left panels show how the index value changes when moving away from the optimal viewing slice S1, the right panel shows paths moving away from the uninformative slice S2. We see that for the dataset A there is a large variability between the index behavior along the different directions. Moving away from S1 we find some flat directions along which the index value remains large. This suggests that the function has ridges. This is expected for this data, as a result of a symmetry in the simulation distribution, the structure remains visible so long as the first variable is dominant along one direction in the plane. This also makes the structure easier to detect, and we find that among the 100 random directions, several traces reach index values close to that of the ideal view. On the other hand, the 100 traces shown for set B have much less variability, and all result in an approximately linear decay of the index value as we move away from the S1 slice. Similarly, they all show approximately linear increase in index value when moving away from S2. We can also see that the index is noisy, sometimes producing jumps in index values under small rotations of the slicing projection.

We can also use this visualization to better understand the generalizations of the index. In Fig. 10 we show the topotrace plots for the same settings as Fig. 9, but with
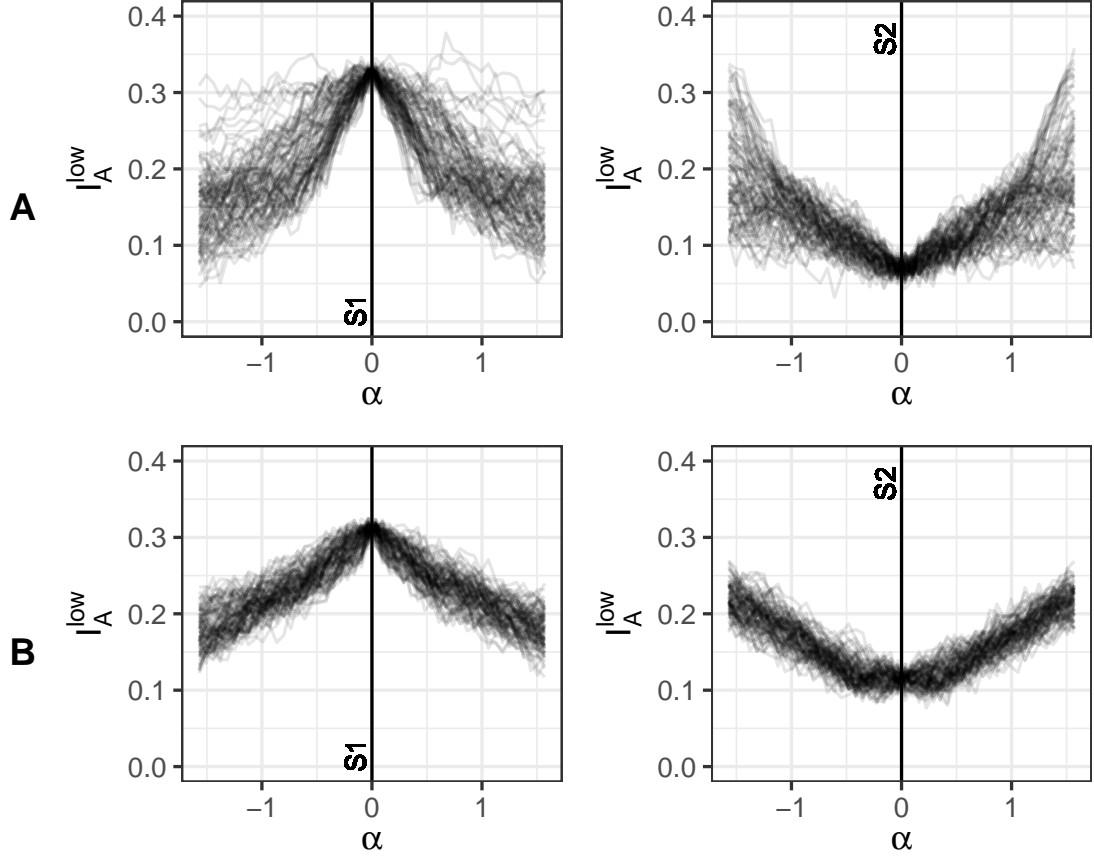
Figure 9. Topotrace plots showing index function characteristics for the index with $q = 1$, $\alpha = \pi/2$, $m = 100$, for example data sets A (top) and B (bottom) from the starting planes, the informative slice S1 (left) and the uninformative slice S2 (right). Example A indicates ridges in the function because for several traces the index remains high when varying α. Example B has a more gradual decline from the peak. The dependence of the index on α is not smooth.

$q = 2$. We see that this results in a smaller squint angle, and steeper change of the index near the optimal S1 slice, while the index is flatter away from this view.

4.7.   *Index in practice*

When using the index in practice we follow these steps:

1. Check the underlying assumption that the data is inside a hypersphere. This can be ensured by first centering and scaling all variables, and then dropping points that have a radius above the maximum r_{max}.

2. Choose the number of bins. Setting the bin size for polar binning uses the maximum radius and defines n_r equidistant radial bins and n_θ angular bins. The preferred values will depend on the sample size and slice thickness.
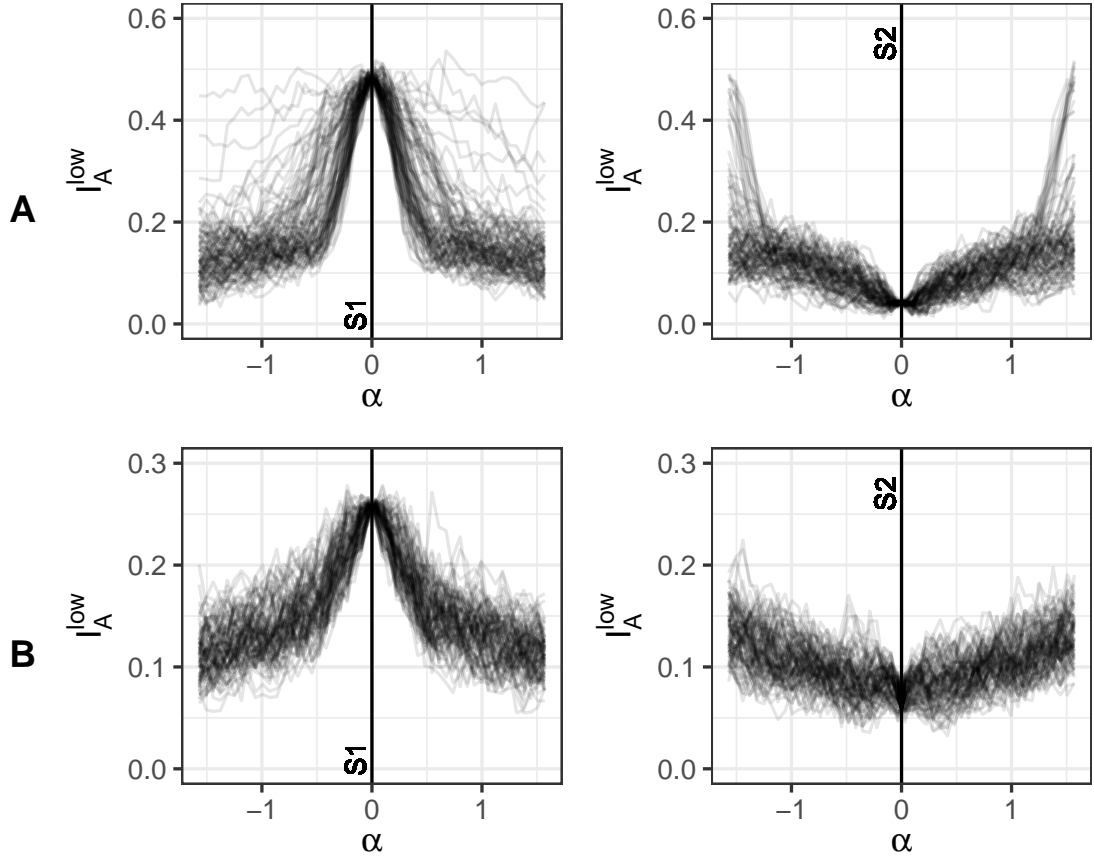
20

Figure 10. Topotrace plots for the index with $q = 2$, $\alpha = \pi/2$, $m = 100$ for set A (top) and set B (bottom) from the informative slice S1 (left) and the uniformative slice S2 (right). The choice of $q = 2$ results in a smaller squint angle, and steeper change of the index.

3. Decide the slice radius h to use. We can think of it in terms of a relative resolution h/r_{max}.

4. The binning and resolution, together with the number of variables and number of observations are needed to estimate the uncertainties following Eq.14.

5. We can now define the slice index, which takes the binning and uncertainty estimates as input. We define the index such that for each slice it reweights the counts according to Eq. 10.

6. For the optimization we use a modified version of the guided tour that passes the projected points, the distance vector and the radius h into the index function. This can directly be used together with the optimization routines from the `tourr` package.

5. Applications

5.1. *Index settings*

For the application considered we find that the following parameters work well:

- We use the index as defined in Eq. 2.
- When binning we set $n_r = 5$ and $n_\theta = 8$ or 10.
- For our examples, a slice radius $h = 0.25$ worked well for scaled data (standard deviation = 1).

5.2. *Classification boundaries*

The slice display can be used to understand non-linear classification boundaries (Laa, Cook, and Valencia 2019), visualized following Wickham, Cook, and Hofmann (2015) and using the classifly package (Wickham 2020). Classifly samples the design space and calculates model predictions. We can use section pursuit to identify slices that reveal the decision boundaries in the design space, by dropping sample points based on the assigned prediction. By selecting a class that is only predicted in a small region we generate the hollow features to be found by section pursuit. The resulting slice can then be viewed showing all assigned classes to resolve the boundary. An example using

Olives data

We first consider the classical olives data (Forina, Armanino, and Lanteri 1983). This data set contains measurements of 8 fatty acids for 572 Italian olive oils, collected in 9 different areas, and is available in the classifly package.

We fit a support vector machine (svm) classification model using the implementation in the e1071 R package (Meyer et al. 2019), to predict the area from the fatty acid measurements. For this example we consider 4 of the variables: palmitoleic, stearic, linoleic and arachidic, and we use a radial kernel for the svm model. We use classifly for automated sampling of the design space and evaluation of the predictions. For the visualization we first center and scale the data to have standard deviation one. We then select only those points that are inside a 4D hypersphere.

As an example we select as the area of interest West-Liguria and drop all samples with this predicted class before performing section pursuit. The model predictions are shown in slices and projections in the first two columns of Fig. 11, the last column shows the projected data. The views in the first row are defined by the projection onto palmitoleic-stearic and in the second row by the projection onto linoleic-arachidic. The color indicates if the predicted area is West-Liguria (orange) or not (green). We see that the parameters palmitoleic and stearic do not allow us to distinguish the West-Liguria region in the projected data. The region where the svm model predicts this class is hidden in the projection, and it is not predicted anywhere inside the thin slice. On the contrary, linoleic and arachidic can separate the West-Liguria area from the others. Looking at the model predictions, we find that projections can partly reveal the part of the model space resulting in this prediction, while the slice can also resolve the non-linear decision boundary.

Next we run section pursuit on the reduced svm sample. We define the section pursuit index with polar binning, with 5 equidistant radial bins and 10 angular bins. The bin counts are reweighed according to Eq. 10. We set $q = 1$ and then optimize the index using the geodesic search available in the tourr package, starting form the slice defined by the projection onto linoleic-arachidic (see Fig. 11, second row). The
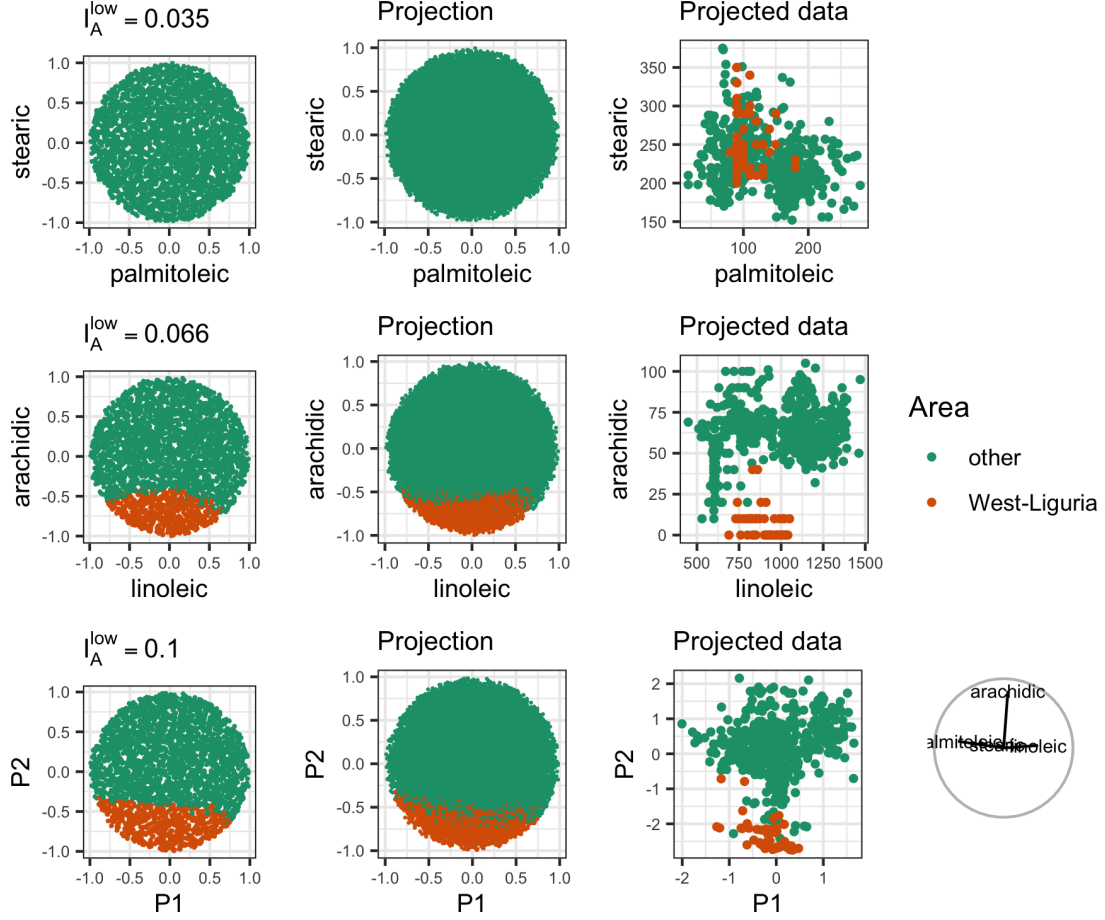
Figure 11. SVM classification of the West-Liguria region (orange) against all other regions (green) in the olives data. The first and second row are projections onto pairs of variables, the last row shows the final result obtained via section pursuit. The first and second column show the svm classification in a thin slice and as a full projection, the third column shows the projected data. The last column shows guides, including the axes for the final view at the bottom.

final views obtained in the optimization are shown in the bottom row of Fig. 11.

Looking at the data projected onto linoleic and arachidic, and the corresponding slice view of the classifier, we see a non-linear decision boundary that allows to separate the West-Liguria area from the other regions. This non-linearity is hidden in the projected view of the classifier. The section pursuit has identified a slice that has a higher index value and shows a linear decision boundary and with a larger area of the section predicting the selected class. The projected model predictions in this plane do not allow us to resolve this boundary. Interestingly, to maximize the area in the sliced view of the predictor, section pursuit has found a plane that leads to a linear decision boundary on the projected data.

24

5.3. *Inequality condition*

Similar to the decision boundaries of classification models, inequality conditions can induce non-linear boundaries in the parameter space. The inequalities may be complicated functions of multiple parameters, and we can use section pursuit to find visualizations that illustrate the boundaries. These are of interest in physics models for which we often have inequality constraints defining the allowed region of parameter space. By understanding the shape of this region we can gain physics insights guiding theoretical and experimental work. For example, the constraints might result in preferred parameter combinations hinting at a simpler underlying model, or we can reparametrise the allowed region to design targeted experiments to test the model further.

THDM

A certain type of particle that occurs in particle physics models is the so-called scalar (or Higgs) boson, with properties specified by a "potential" which depends on a set of parameters. These are the basic parameters of the model and they allow one to calculate physical quantities such as the masses of the scalar particles and their interaction strengths. A common problem that arises in this context consists of finding the parameter space that leads to acceptable predictions, that satisfy known theoretical or experimental constraints.

For example, a two-Higgs-doublet model can be written under certain assumptions in terms of the parameter set $\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5, \alpha, \beta$, where the last two are angles. These angles are commonly discussed in terms of $\tan(\beta)$ and $\cos(\beta-\alpha)$, as these quantities are closely related to experimental observations. This model contains five scalar particles (types of "Higgs" bosons), dubbed $h, H, A, H^{\pm}$, the first one corresponding to the famous Higgs boson found at CERN in 2012. The squared masses of these five particles are predicted in terms of the parameters of the model, and we use a form from Gunion and Haber (2003) (given in the Appendix).

For the model to be viable it needs to satisfy a number of conditions that restrict the parameter space that is allowed. The simplest of these restrictions is that the masses must be real numbers: only parameters for which all squared masses are greater than 0 are viable. We will use this as our example, sampling $\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5, \tan(\beta), \cos(\beta-\alpha)$

within a 7D hypersphere. We then evaluate the predicted mass spectrum for each sample and flag all points that result in non-physical predictions for (some of) the masses. These points are dropped from the dataset and we use section pursuit to find regions that lead to a non-physical mass spectrum.

For the visualization we first standardize each parameter and then drop all sample points for which the condition is not met. We then apply section pursuit to the reduced sample to find sections that are associated with real masses. We use the index with polar binning, with 5 equidistant radial bins up to the maximum radius, and 10 angular bins. The ε cutoff is calculated according to Eq. 14 and we reweight the bins according to Eq. 10. We set $q = 1$ and use the search_better optimization to find the view with the maximum index value.

Two of the resulting views are shown in Fig. 12. The samples that violate the condition are shown in red, the remaining samples (those used for the section pursuit) are shown in black. The top row shows a view encountered along the optimization path, i.e. a section with relatively large index value. The slice is shown on the left and reveals the non-linear boundary defined by the conditions. The corresponding projection is shown in the middle and cannot resolve the feature. The bottom row shows the final view obtained via section pursuit.

Both slices show interesting aspects of the boundary. In the first slice the condition is only violated in a small but well-defined region of the plane. Section pursuit has identified a slice where the condition is often violated, and the final view shows a complex non-linear boundary between the two regions. The boundaries are hidden in the corresponding projections. Looking at the axes representation of this projection, we find that the optimal slice is defined by a combination of all the input parameters, making direct interpretation challenging.

6. Conclusion and discussion

This paper introduces a section pursuit index that can be used to detect hollow or dense features that are visible in slices but hidden in projections. The index uses the distribution outside each current slice as a reference, and computes its difference
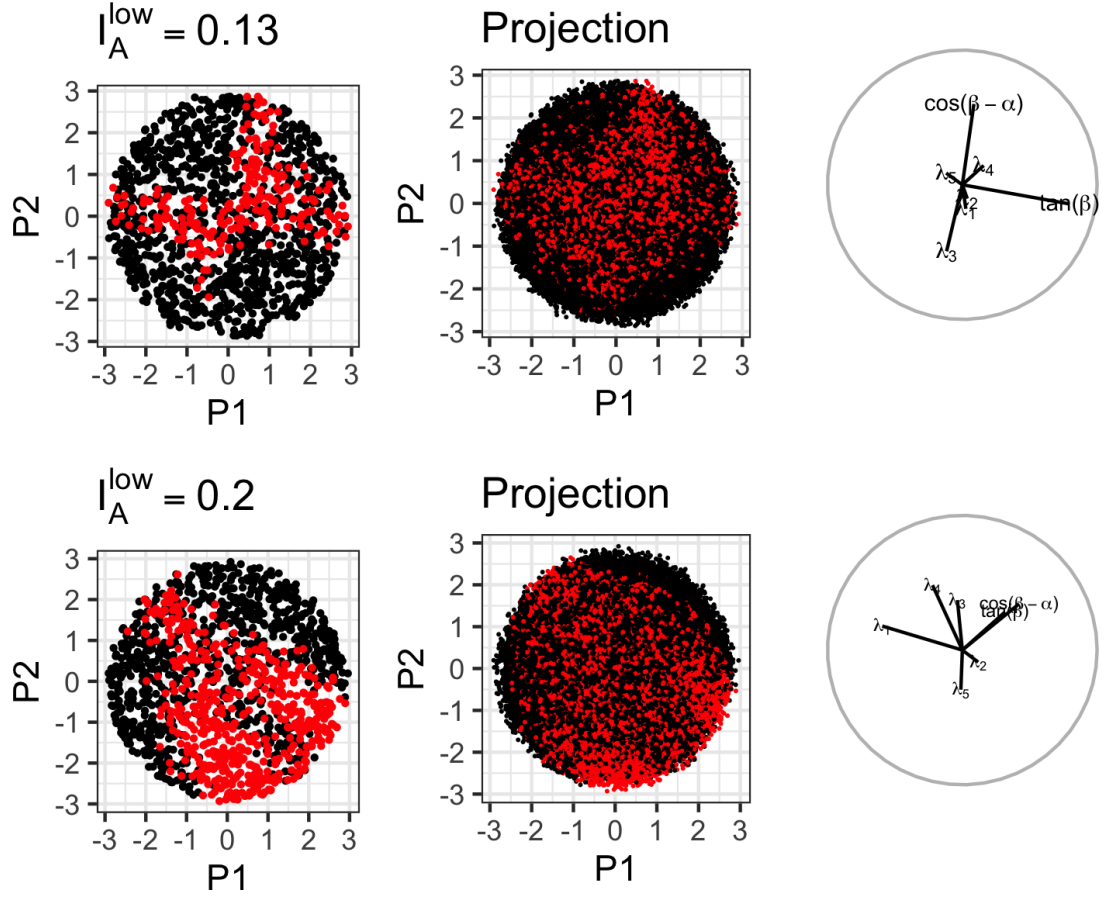
Figure 12. Slice (left) and projection (middle) views found when optimising the section pursuit index for the THDM sample. Black points satisfy the inequality conditions, red points do not and are dropped in the optimisation. The first row shows a view along the optimisation path, and the last row the final slice selected by section pursuit. The corresponding axes are shown on the right. The selected slices show clear separation of points not satifsying the conditions, which is hidden in the corresponding projections.

from the slice distribution. The comparison of the two distributions is calculated as the positive sum of differences of normalized bin counts. To avoid differences arising from the overall shape we consider spherical multivariate distributions. In addition, we use the expected cumulative distribution function to reweight the inside and outside distributions separately, such that the expected counts are uniform across all bins.

The section pursuit index can be used together with the slice display in the tourr package to define a guided slice tour. This means that we can use the available optimization routines, and we can further look at the interpolated optimization path. In practice, to conform to the assumption of an underlying spherical distribution, we can shave off points outside the hypersphere. Since the focus is on detecting hidden features in the center of the distribution, we do not expect to lose important information doing so. In case of simulated data, augmenting the data to be of this form is also an option.

We have shown how to use the section pursuit index to explore non-linear decision boundaries of classification models, or similarly, to explore complex inequality conditions that depend on multiple parameters. In all examples, we have sampled the parameter space and evaluated the classification or inequality conditions for all points. We then dropped points assigned to one particular class, or in conflict with the inequality condition to generate the hollow features in the distribution. For all examples considered, the optimization resulted in slice views that illustrate interesting aspects of the boundaries. Additional potential applications for section pursuit include the exploration of non-standard multivariate confidence regions or Bayesian credible regions.

Note that large samples are required to obtain useful insights using the section pursuit method. As discussed in the context of sufficient sample size, the number of data points quickly grows with the number of dimensions. The applications we have studied here are considering simulated data in up to seven dimensions. In the case of larger parameter space we recommend to first reduce dimensionality: in the olives-data example we have selected variables based on prior knowledge, in the PDFSense example in the Appendix we have used principal component analysis prior to the application of section pursuit. The method is not appropriate for small sample size

data, in those cases interactive methods such as linked brushing will be preferred.

Depending on the application, different index definitions would be recommended. Several variations on the index would be produced different choices of weights, and parameters in the generalized index definition. This paper hasn't fully explored the impact of all the choices, but the same type of diagnostics, in particular, the index visualization presented in Section 4.6, could be used decide.

Given the compositional nature of the normalized bin-count data, considering e.g. log-ratios in the definition, could also define an interesting index. It may also be reasonable to consider using a kernel, instead of discrete bins, to produce some spatial smoothing. The effect would be to focus attention on large differences in specific regions rather than small differences anywhere. A completely different approach to producing a section index, would be to apply any existing projection pursuit index on a sliced projection instead of the full projected data. Only observations inside the slice are then used for the index calculation.

Acknowledgements

Supplementary material

- Code and data is available at `https://github.com/uschiLaa/paper-section-pursuit`.
- The Appendix contains the derivation of the radial CDF of a hypersphere projected onto a 2D plane and the equations used to calculate the masses in the two-Higgs-doublet model, and an additional example from physics.

References

Asimov, D. 1985. "The Grand Tour: A Tool for Viewing Multidimensional Data." *SIAM Journal of Scientific and Statistical Computing* 6 (1): 128–143.

Buja, A., D. Cook, D. Asimov, and C. Hurley. 2005. "Computational Methods for High-Dimensional Rotations in Data Visualization." 391–413.

Cook, D., A. Buja, J. Cabrera, and C. Hurley. 1995. "Grand Tour and Projection Pursuit." *Journal of Computational and Graphical Statistics* 4 (3): 155–172.

Forina, Michele, C. Armanino, and Sergio Lanteri. 1983. "Classification of olive oils from their fatty acid composition." *Food Research and Data Analysis* 189–214.

Friedman, J. H., and J. W. Tukey. 1974. "A Projection Pursuit Algorithm for Exploratory Data Analysis." *IEEE Transactionson Computers* 23: 881–889.

Furnas, George W., and Andreas Buja. 1994. "Prosection Views: Dimensional Inference through Sections and Projections." *Journal of Computational and Graphical Statistics* 3 (4): 323–353. `http://www.jstor.org/stable/1390897`.

Gous, Alan, and Andreas Buja. 2004. "Visual Comparison of Datasets Using Mixture Decompositions." *Journal of Computational and Graphical Statistics* 13 (1): 1–19. `https://doi.org/10.1198/1061860043119`.

Gunion, John F., and Howard E. Haber. 2003. "The CP conserving two Higgs doublet model: The Approach to the decoupling limit." *Phys. Rev.* D67: 075019.

Huber, P. J. 1985. "Projection Pursuit." *The Annals of Statistics* 13 (2): 435–475.

Kruskal, J. B. 1969. "Toward a Practical Method Which Helps Uncover the Structure of a Set of Observations By Finding the Line Transformation Which Optimizes a New 'Index of Condensation'." In *Statistical Computation*, edited by R. C. Milton and J. A. Nelder, 427–440. New York: Academic Press.

Kruskal, J.B., and M. Wish. 1978. "Multidimensional Scaling." *Sage University Paper Series on Quantitative Applications in the Social Sciences* No. 07-011.

Laa, Ursula, and Dianne Cook. 2020. "Using tours to visually investigate properties of new projection pursuit indexes with application to problems in physics." *Computational Statistics* `https://doi.org/10.1007/s00180-020-00954-8`.

Laa, Ursula, Dianne Cook, and German Valencia. 2019. "A slice tour for finding hollowness in high-dimensional data." `https://arxiv.org/abs/1910.10854`.

Lee, Eun-Kyung. 2018. "PPtreeViz: An R Package for Visualizing Projection Pursuit Classification Trees." *Journal of Statistical Software* 83 (8): 1–30.

Meremianin, A. V. 2009. "Hyperspherical harmonics with arbitrary arguments." *Journal of Mathematical Physics* 50 (1): 013526. `https://doi.org/10.1063/1.3054274`.

Meyer, David, Evgenia Dimitriadou, Kurt Hornik, Andreas Weingessel, and Friedrich Leisch. 2019. *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien*. R package version 1.7-3, `https://CRAN.R-project.org/package=e1071`.

O'Connell, Mark, Catherine Hurley, and Katarina Domijan. 2017. "Conditional Visualization for Statistical Models: An Introduction to the condvis Package in R." *Journal of Statistical Software, Articles* 81 (5): 1–20. `https://www.jstatsoft.org/v081/i05`.

R Core Team. 2018. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. `https://www.R-project.org/`.

Torsney-Weir, T., M. Sedlmair, and T. Möller. 2017. "Sliceplorer: 1D slices for multi-dimensional continuous functions." *Computer Graphics Forum* 36 (3): 167–177. `https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13177`.

Wickham, Hadley. 2020. *classifly: Explore classification models in high dimensions*. R package version 0.4.0.99, `http://had.co.nz/classifly`.

Wickham, Hadley, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, et al. 2019. "Welcome to the tidyverse." *Journal of Open Source Software* 4 (43): 1686.

Wickham, Hadley, Dianne Cook, and Heike Hofmann. 2015. "Visualizing statistical models: Removing the blindfold." *Statistical Analysis and Data Mining: The ASA Data Science Journal* 8 (4): 203–225. `https://onlinelibrary.wiley.com/doi/abs/10.1002/sam.11271`.

Wickham, Hadley, Dianne Cook, Heike Hofmann, and Andreas Buja. 2011. "tourr: An R Package for Exploring Multivariate Data with Projections." *Journal of Statistical Software* 40 (2): 1–18. `http://www.jstatsoft.org/v40/i02/`.

Xie, Yihui. 2015. *Dynamic Documents with R and knitr*. 2nd ed. Boca Raton, Florida: Chapman and Hall/CRC. ISBN 978-1498716963, `https://yihui.name/knitr/`.

Xie, Yihui, J.J. Allaire, and Garrett Grolemund. 2018. *R Markdown: The Definitive Guide*. Boca Raton, Florida: Chapman and Hall/CRC. ISBN 9781138359338, `https://bookdown.org/yihui/rmarkdown`.