Dynamic visualization of high-dimensional data via low-dimension projections and sectioning across 2D and 3D display devices

Candidature confirmation report

Nicholas S Spyrison

Supervisors:

Prof. Kimbal Marriott,

Prof. Dianne Cook,

Prof. German Valencia

Faculty of Information Technology

Monash University

Australia

February 2020

Contents

Chapter 1

spinifex: manual control of dynamic linear projections of high-dimensional data

1.1 Abstract

The class of dynamic linear projections that are collectively known as 'tours' provide a unique dynamic visualization of numeric multivariate data. Tours are particularly useful for understanding the structure held within multivariate data, and in association with techniques for dimension reduction, supervised, and unsupervised classification. The *R* package *tourr* offers a variety of path generators and geometric displays for conducting tours. This paper discusses an extension package, *spinifex*, that adds support for the path generation of manual tours and extends the display of tours to use with the contemporary animation packages, *plotly* and *gganimate*. Manual tours are used to explore the sensitivity of structure as the contributions of a manipulation variable are changed. This particularly useful after identifying a feature of interest.

A recent paper {Wang et al. (2018)} visualizes the sensitivey of the hadronic experiments to nucleon structure. Sensitivity was characterized in non-linear 3D embeddings of the first 10 principal components. This research applies manual tours to this data showing

that manual tours resolves more structrual information that is orthogonal to the original viewing plane.

Keywords: manual tour, guided tour, grand tour, projection pursuit, high dimensional data, multivariate data, data visualization, statistical graphics, data science.

1.2 Introduction

A tour is a multivariate data analysis technique in which a sequence of linear (orthogonal) projections are viewed as an animation while the orientation of the projection basis is rotated across time. Each frame of the sequence corresponds to a small change in the projection for a smooth transition that perseveres continuity.

While there are numerous methods that generate tour paths, this research focuses on the manual tour. The manual tour was described in Cook and Buja (1997) and allows a user to control the projection coefficients of a select variable has in a 2D projection. The manipulation of these coefficients allows the analyst to explore how sensitive the projections structure is to these changes. This makes manual tours particularly useful once a feature of interest has been identified, for example, with the use of a guided tour (Cook et al., 1995). The path of a guided tour is selected via projection pursuit, the optimization of an index function on the projection via a hill climbing algorithm. This allows guided tours to identify interesting projection features rapidly given the relatively large parameter-space. Once the given projection has been provided, it is time to define the path of rotation.

Ideally, the path would be intuitively user-generated from physical movement, be it through mouse or motion capture. Unfortunately, this type of dynamic control has proven difficult to capture for in R. Because of this, manual tours were not implemented in *tourr*. This research allows for the consumption, but not the generation, of such dynamic input. After the capture of an oblique user motion, the rotation needs to be applied to step 3 (rotation sequence) of the algorithm discussed below. In the section below we stick with a radial rotation where, θ, the angle of in-projection-plane rotation is held constant.

Spinifex utilizes two new animation packages, *plotly* (Sievert, 2018) and *gganimate* (Pedersen and Robinson, 2019), to display tours, manual or other saved tours. From a given projection, the user can choose which variable to control, and the animation sequence is generated to remove the variable from the projection, and then extend its contribution to be the sole variable in one direction. This allows the viewer to assess the change in structure induced in the projection by the variable's contribution.

The paper is organized as follows. Section 1.3 explains the algorithm using a toy dataset. Section 1.4 discussed the display of the animation after the path has been generated. Section 1.5 illustrates how this can be used for sensitivity analysis applied to contemporary high energy physics. The last section, 1.8 summarizes the work and discusses future research.

1.3 Algorithm

The section below describes the algorithm for performing a 2D radial manual tour:

1. Provided with a 2D projection, choose a variable to explore. This is called the "manip" variable.
2. Create a 3D manipulation space, where the manip variable has the full contribution.
3. Generate a rotation sequence which increases the norm of the coefficient to 1 and zeros it.

The steps are described in more detail below. The R functions used below mentioned briefly, but more complete code example can be found in section 1.7

1.3.1 Notation

This section describes the notation used in the algorithm for a 2D radial manual tour.

- $\mathbf{X}$, the data, an $n \times p$ numeric matrix to be embedded in two dimensions.
- $\mathbf{B} = (B_1,\ B_2)$, any of orthonormal projection basis set, $p \times 2$ matrix, describing the projection from p to two dimensions

- **e**, a zero column vector of length p with the $k-$th element set to one, where k is the number of the variable to manipulate.
- θ, the angle of in-projection-plane rotation, for example, on the reference axes.
- ϕ, the angle of out-of-projection-plane rotation, coming into the manipulation space.

The algorithm primarily operates on the projection basis and utilizes the data only when making a display. The projection space can be viewed at any point in the process by pre-multiplying the data and plotting the first 2 variables.

1.3.2 Toy data set

The flea data, originally from Lubischew (1962), available in the R package *tourr* (Wickham et al., 2011) is used to illustrate the algorithm. The data contains 74 observations across 6 variables, physical measurements of the flea beetles. Each observation belonging to one of three species.

The data is defined. A basis set (ideally that views an interesting feature) should be provided to explore the sensitivity of the variables to the structure. To identify a projection containing an interesting feature, apply a guided tour(Cook, Swayne, and Buja, 2007) on the flea data. In a guided tour the projection sequence is selected by optimizing an index via hill-climbing. In this case, the holes index is selected. The holes index is maximized by when the projected observations are furthest from the center. Figure 1.1 shows a locally optimized projection for this data. The left plot displays the reference axes of the projection basis, a visual indication of the magnitude and direction each variable contributed to the projections. The right plot shows the projection of the data through the basis set described by the reference axes (left). Data points are colored and given point characters according to the species of the flea (the guided tour was unsupervised with this information).

Call `view_basis()` on a basis to produce a *ggplot2* graphic similar to 1.1. Projection space is always available for display via the matrix multiplication $\mathbf{X}_{[n,\ p]} * \mathbf{B}_{[p,\ d]} = \mathbf{P}_{[n,\ d]}$.

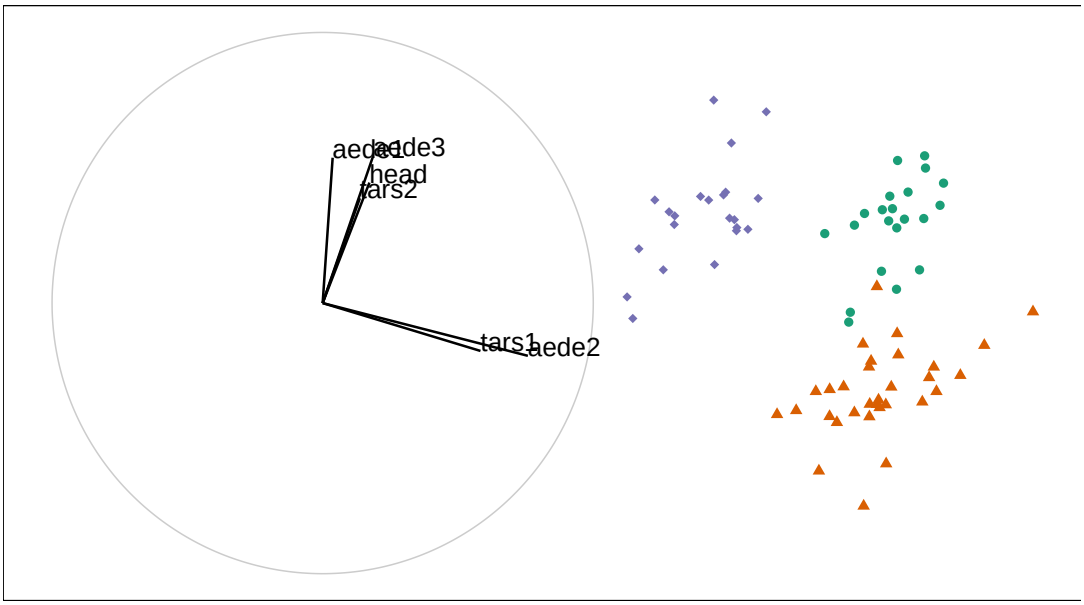

Figure 1.1: *Basis reference axes (left) and projected data (right) of standardized flea data. Data points color and shape are mapped to beetle species. Basis identified by a holes-index guided tour. The variables* `aede2` *and* `tars1` *contribute mostly orthogonal to the other variables. We'll select* `aede2` *as our manipulation variable to see how the structure of the projection changes as we rotate* `aede2` *into and out of the projection.*

1.3.3 Step 1) Choose variable of interest

In figure 1.1, above, the contributions of the variables `tars1` and `aede2` are mostly orthogonal to the contributions of the other four variables. These two variables explain the variation of the data between the purple and green species. We select `aede2` as the manip var, the variable to be manipulated as it typically has a larger contribution after the optimizing the holes index. The question that will be explored in the explanation of the algorithm is how important the variable `aede2` is to the separation of the clusters.

1.3.4 Step 2) Create the manip space

Initialize a zero vector e of p elements. Because `aede2` is the fifth variable in the data, set the $k = 5$-th element to one giving the manip var a full contribution in this dimension. Use the Gram-Schmidt process to orthonormalize the zero vector onto the basis yielding the 3D manipulation space, $\mathbf{M}$.

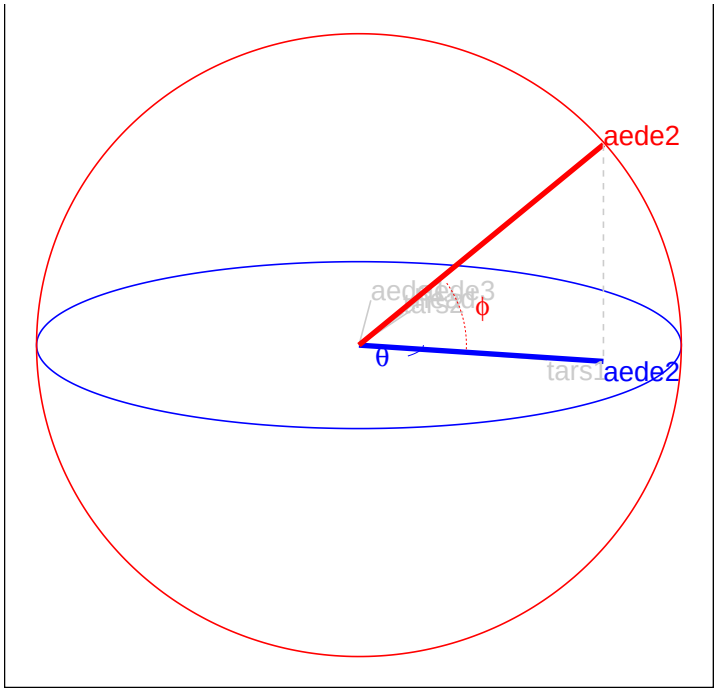

Figure 1.2: *Manipulation space for controlling the contribution of* `aede2` *of standardized flea data. Basis selected by a holes-index guided tour. The Projection plane is shown in blue. The manipulation axis, in red, allows the coefficients of the manip var to be changed.*

$$\begin{aligned}\mathbf{e} &\leftarrow Orthonormalize_{GS}(\mathbf{e}) w.r.t. Basis \\ &= \mathbf{e} - \langle \mathbf{e}, \mathbf{B}_1 \rangle \mathbf{B}_1 - \langle \mathbf{e}, \mathbf{B}_2 \rangle \mathbf{B}_2\end{aligned}$$

$$\mathbf{M}_{[p,\,3]} = (\mathbf{B}_1, \mathbf{B}_2, \mathbf{e})$$

Adding this extra dimension to our basis plane allows for the coefficients of the specified variable to be changed. For example, the ability to lift a piece of paper, rather than being constrained to the motion on a table top. Orthonormalizing rescales the new depth vector while the projection down to 2D is the original basis, that is the first d vectors remain constant. Imagine the reference axes (and projection plane) laying flat on a table, while a new dimension exists with axes projecting back onto the reference axes. An illustration of such can be seen below in figure 1.2. The manip var is highlighted, while the depths of the other variables are not depicted.

The representation in 1.2 can be duplicated by calling the function `view_manip_space()`.

1.3.5 Step 3) Generate rotation

Imagine holding the red axis it is fixed to the origin. As it is manipulated the projection back onto the projection plane correspondingly moves. This is what happens in a manual tour. For a radial tour, fix θ, the angle within the blue plane, and vary the sequence of ϕ, the angle coming out of the projection plane. Conceptually, live manipulation on a 2D plane allows the user to dynamically control these angles, effectively changing the coefficients of the manip var, which then performs a constrained rotation on the remaining variables.

For the demonstration of the radial tour, we define a sequence for ϕ that brings the initial contribution of the manip var to be maximized and then zeroed before returning to the initial position.

For i in 1 to n_slides:

Post-multiply the manipulation space by the pre-defined rotation matrix producing **RM**, the rotated manip space.

Let:

c_θ be the cosine of θ

c_ϕ be the cosine of ϕ

s_θ be the sine of θ

s_ϕ be the sine of ϕ

then

$$\mathbf{RM}_{[p,\,3,\,i]} = \mathbf{M}_{[p,\,3]} \;*\; \mathbf{R}_{[3,\,3]}$$

$$= \begin{bmatrix} M_{1,1} & M_{1,2} & M_{1,3} \\ M_{2,1} & M_{2,2} & M_{2,3} \\ \vdots & \vdots & \\ M_{p,1} & M_{p,2} & M_{p,3} \end{bmatrix}_{[p,3]} * \begin{bmatrix} c_\theta^2 c_\phi s_\theta^2 & -c_\theta s_\theta (1 - c_\phi) & -c_\theta s_\phi \\ -c_\theta s_\theta (1 - c_\phi) & s_\theta^2 c_\phi + c_\theta^2 & -s_\theta s_\phi \\ c_\theta s_\phi & s_\theta s_\phi & c_\phi \end{bmatrix}_{[3,3]}$$

A note on application: compile the sequence of ϕ_i and create an array/long table for each rotated manipulation space. ϕ is the angle relative to the initial value of ϕ, we find the transformation ϕ_i - ϕ_1 useful to think about ϕ relative to the basis plane. Additionally, the value of ϕ may be offset by a factor of pi. If the manip variable doesn't move as expected these are the first places to check.

```
for (phi in seq(seq_start, seq_end, phi_inc_sign)) {
  slide <- slide + 1
  tour[,, slide] <- rotate_manip_space(manip_space, theta, phi)[, 1:2]
}
```

Figure 1.3 illustrates a sequence with 15 projected bases and highlight the manip variable on top while showing the corresponding projected data points on the bottom. Take note of how the changes in the manip var change the distance between the purple and green cluster of points, `aede2` is crucial in distinguishing between these groups. Tours are typically viewed as an animation such a dynamic version of this tour can be viewed online at https://nspyrison.netlify.com/thesis/flea_manualtour_mvar5/. The page may take a moment to load. The format of this figure and linking to an HTML animation will be used again in the Application, section 1.5.

Animations can be produced using the function `play_manual_tour()`. This function defaults to an HTML5 widget produced from *plotly*.

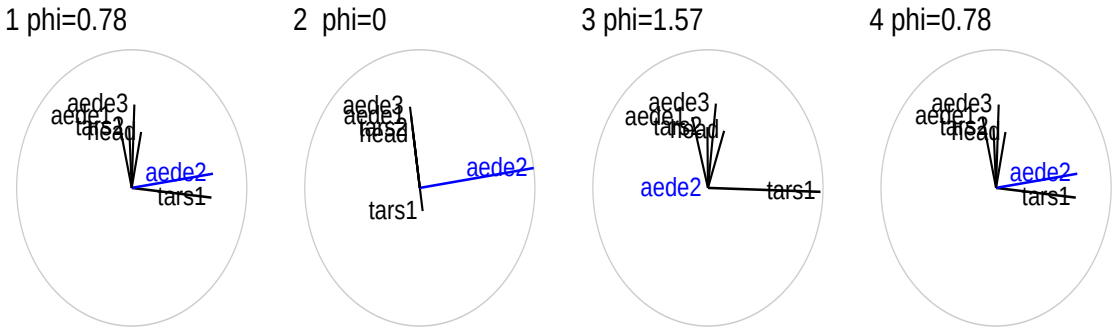

Figure 1.3: *Radial manual tour changing the contributions from* `aede2` *of standardized flea data. The contributions increase from its initial contribution to a full contribution to the projection before decreasing to zero and then returning to its initial value. The change in the projected data shows that* `aede2` *is important for distinguishing between the purple and green clusters. An animated version can be viewed at* `https://nspyrison.netlify.com/thesis/flea_manualtour_mvar5/`.

1.4 Data in projection-space

In light of performance, the above operations are performed on the bases without the use of the larger datasets. After the bases are brought into the projection-space, however, it is helpful to observe them with data in the same space. Pre-multiply the data by basis frame bringing the data into the projection space.

$$\mathbf{P}_{[n,\,3]} = \mathbf{X}_{[n,\,p]} \;*\; \mathbf{RM}_{[p,\,3]} \tag{1.1}$$

$$= \begin{bmatrix} X_{1,\,1} & \dots & X_{1,\,p} \\ X_{2,\,1} & \dots & X_{2,\,p} \\ \vdots & \vdots & \vdots \\ X_{n,\,1} & \dots & X_{n,\,p} \end{bmatrix}_{[n,\,p]} * \begin{bmatrix} RM_{1,\,1} & RM_{1,\,2} & RM_{1,\,3} \\ RM_{2,\,1} & RM_{2,\,2} & RM_{2,\,3} \\ \vdots & \vdots & \vdots \\ RM_{p,\,1} & RM_{p,\,2} & RM_{p,\,3} \end{bmatrix}_{[p,\,3]} \tag{1.2}$$

For a 2D scatterplot, plot the first two variables from each frame statically as in the previous figure, or in sequence, producing an animated scatterplot. The remaining variable is sometimes linked to a data point aesthetic (such as size or color) to produce depth cues used in conjunction with the XY scatterplot.

1.4.1 Rendering and sharing

The *tourr* package utilizes R's base graphics for the display of tours. *spinifex* allows tours to be used in rendered in *plotly* Sievert (2018) as an HTML5 object or *gganimate* Pedersen and Robinson (2019) as .gif or .mp4 objects. Both of which build off *ggplot2* objects in internal functions. Sharing of animations is not trivial especially in print and static formats such as .pdf. Even with the use of computers and dynamic file formats capturing the correct resolution, aspect, and display is challenging and many formats quickly bloat file sizes. Keep in mind hosting options and exporting functions from *plotly*, *gganimate* and *tourr*.

1.4.2 Storage

Storing each data point for every frame of the animation is very inefficient. Just as operations are performed on the bases, so too should tour paths be stored as bases. Consider a radial manual tour, we can store the salient features in 3 bases, where ϕ is at its starting, minimum, and maximum values. The frames in between can be interpolated by supplying angular speed. With the use of the `tourr::save_history()` function, the target bases can be saved. From there geodesic interpolation can be used to populate the intermittent frames. This type of interpolation should not be used on manual tours, which have already been initialized into a 3D manipulation space where direct linear interpolation is appropriate.

1.5 Application

In a recent paper, Wang et al. (2018), the authors aggregate and visualize the sensitivity of hadronic experiments to ncleon structure. The authors introduce a new tool, PDFSense, to aid in the visualization of Parton distribution functions (PDF). The parameter-space of these experiments lies in 56 dimensions, $\delta \in \mathbb{R}^{56}$, and are visualized as 3D subspaces of the 10 first principal components in linear (PCA) and non-linear (t-SNE) embeddings.

Using the same data, another study, Cook, Laa, and Valencia (2018), applies grand tours to the same subspaces. Grand tours are able to better resolve the distribution shape of clusters, intra-cluter detail, better outlier detection, and exonerate a claim persened from

TFEP (TensorFlow embedded projections). Table 1 of Cook et al. summarizes the key findings of observations made with PDFSense & TFEP and those from grand tours.

Without getting too domain-specific the data has three primary groupings; DIS, VBP, and jet Each group is a particular class of experiments and each with many experimental datasets which inturn have many observation. Inconsideration of data density and business of the data We conduct manual tours on a subsets of the DIS and jet clusters. This explores the sensitivity of the structue to each of the variables in turn, and we present the subjectively best and worst manaul tour identifying structure in the respective data sets.

1.5.1 Jet cluster

The jet cluster is of interest as it contains the largest data sets and is found to be important in Wang et al. (2018) The jet cluster resides in a smaller dimensionality than the full set of experiments with 4 principal components explaining 95% of the variation in the jet cluster (Cook, Laa, and Valencia, 2018). The data is subset down to ATLAS7old and ATLAS7new to focus in on two groups with a reasonable number of observations that occupy different parts of the subspace. Below, we perform radial manual tours all four principal components within this scope. Visualizing PC3 and PC4 in figure 1.4 (more sturcturally insightful) and figure 1.5 (less sturcturally insightful) respectively, and list links to dynamic animation of all variables.

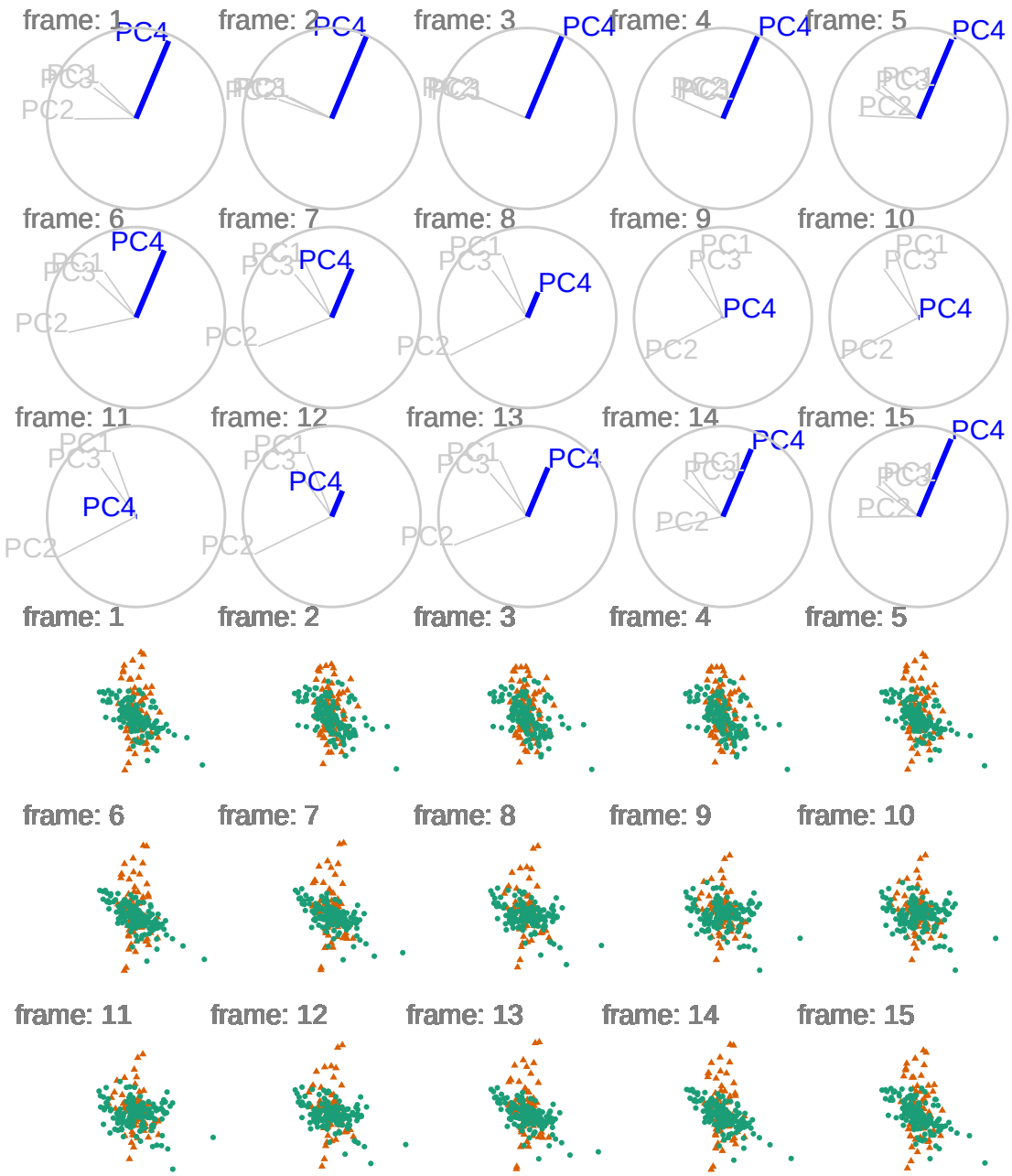

Figure 1.4: *Jet cluster, a radial manual tour of PC3. Colored by experiment type: 'ATLAS7new' in green and 'ATLAS7old' in orange. When PC3 fully contributes to the projection ATLAS7new (green) occupies unique space and several outliers are identifiable. Zeroing the contribution from PC3 to the projection hides the outliers and indeed all observations with ATLAS7new are contained within ATLAS7old (orange). A dynamic version can be viewed at* `https://nspyrison.netlify.com/thesis/jetcluster_manualtour_pc3/`.

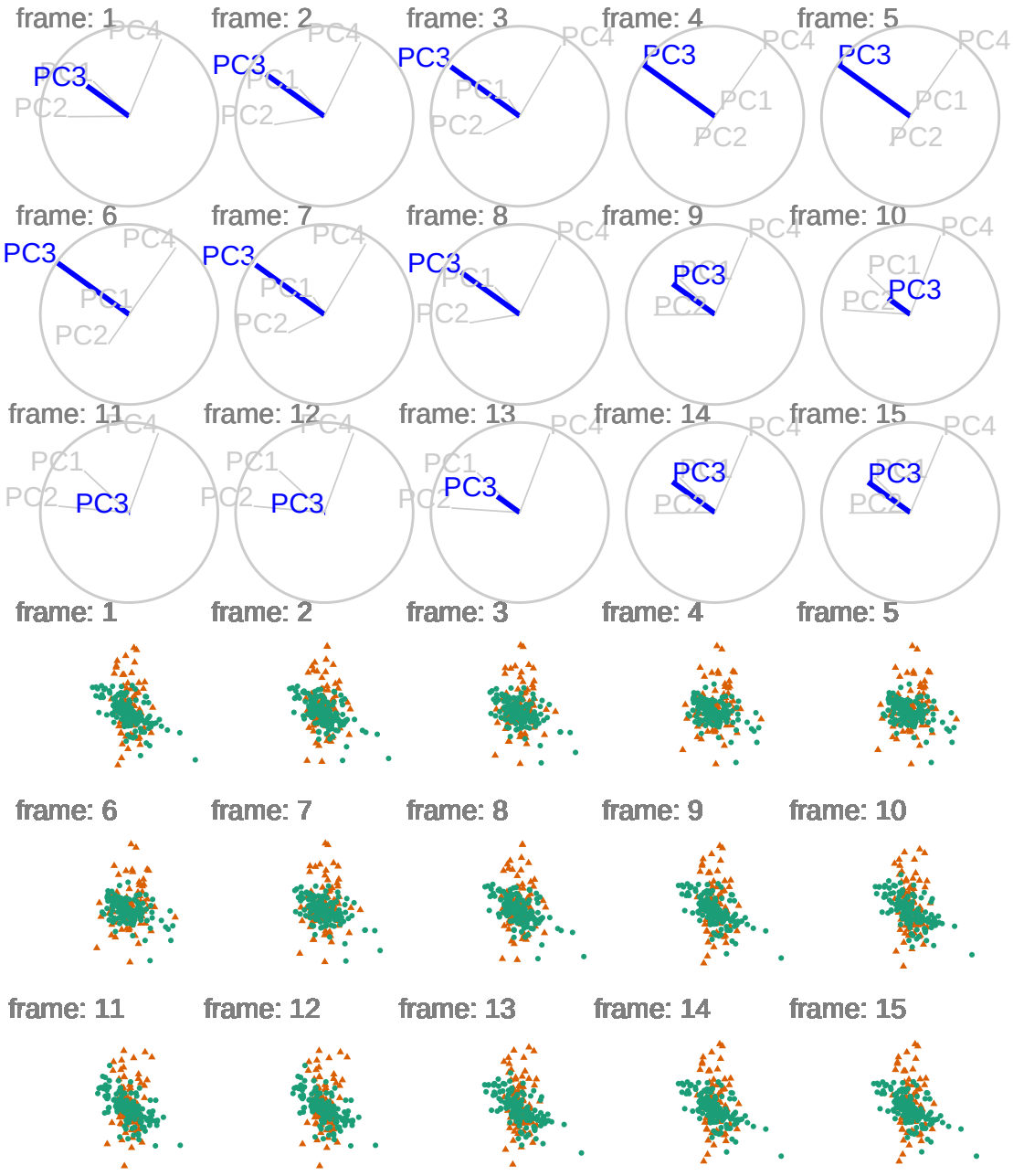

Figure 1.5: *Jet cluster, a radial manual tour of PC4. Colored by experiment type: ATLAS7new in green and ATLAS7old in orange. This manual tour contains less interesting information ATLAS7new (green) has points that are right and left of ATLAS7old, while most points occupy the same projection space, regardless of the contribution of PC4. A dynamic version can be viewed at* `https://nspyrison.netlify.com/thesis/jetcluster_manualtour_pc3/`.

1.6 TODO: edit application, Grammarly and word checks.

Manipulating PC3, where varying the angle of rotation brings interesting features into and out of the center mass of the data, is more interesting than the manipulation of PC4, where the features are mostly independent of the contribution of PC4.

Jet cluster manual tours manipulating each of the principal components can be viewed from the links: PC1, PC2, PC3, and PC4.

1.6.1 DIS cluster

We perform a manual tour on this data, manipulating PC6 as depicted in figure 1.6. Looking at several frames we see that DIS HERA data lies mostly on a plane. When PC6 has full contributions, we see the dimuon SIDIS in purple is almost orthogonal to the DIS HERA (green). Yet the contribution of PC6 has zeroed the dimuon SIDIS data occupy the same space as the DIS HERA data. A dynamic version of this manual tour can be found at: `https://nspyrison.netlify.com/thesis/discluster_manualtour_pc6/`. The page may some time to load, as the animation is several megabytes.

The selection of the correct manip variable is important as the manipulation spaces convey different information. For example, in figure 1.7 we select PC2 as the manip variable finding it to be less insightful than PC6.

DIS cluster manual tours manipulating each of the principal components can be viewed from the links: PC1, PC2, PC3, PC4, PC5, and PC6.

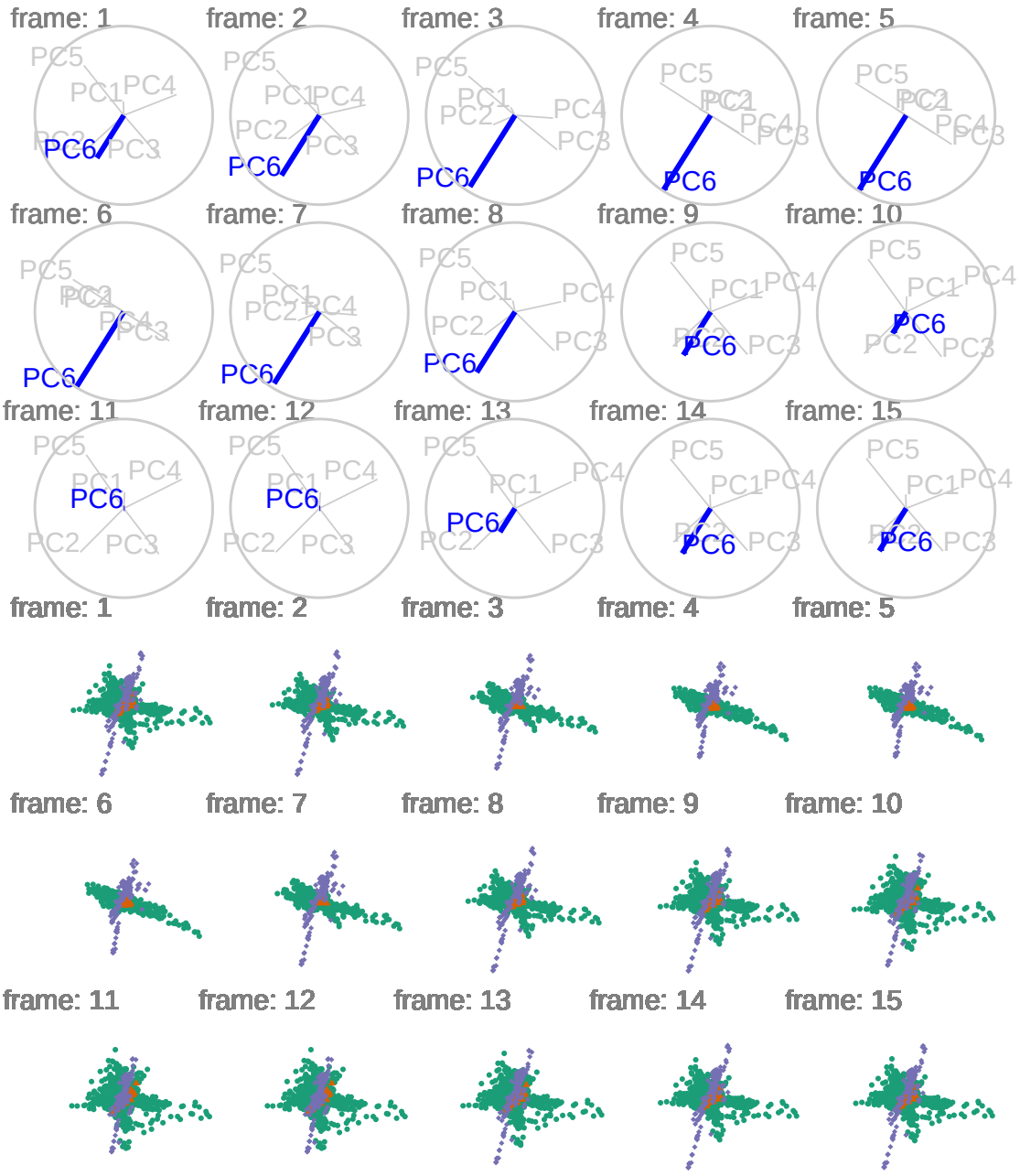

Figure 1.6: *DIS cluster, a radial manual tour of PC6. colored by experiment type: 'DIS HERA1+2' in green, 'dimuon SIDIS' in purple, and 'charm SIDIS' in orange. When the contribution PC 6 is large we see that dimuon SIDIS (purple) data are nearly orthogonal to DIS HERA (green) data. As the projection is rotated, we can also see that DIS HERA (green) practically lies on a plane in this 6D subspace. When the contribution of PC6 is near zero, dimonSIDIS (purple) occupies the same space as the DIS HERA data. A dynamic version can be viewed at* `https://nspyrison.netlify.com/thesis/discluster_manualtour_pc6/`.

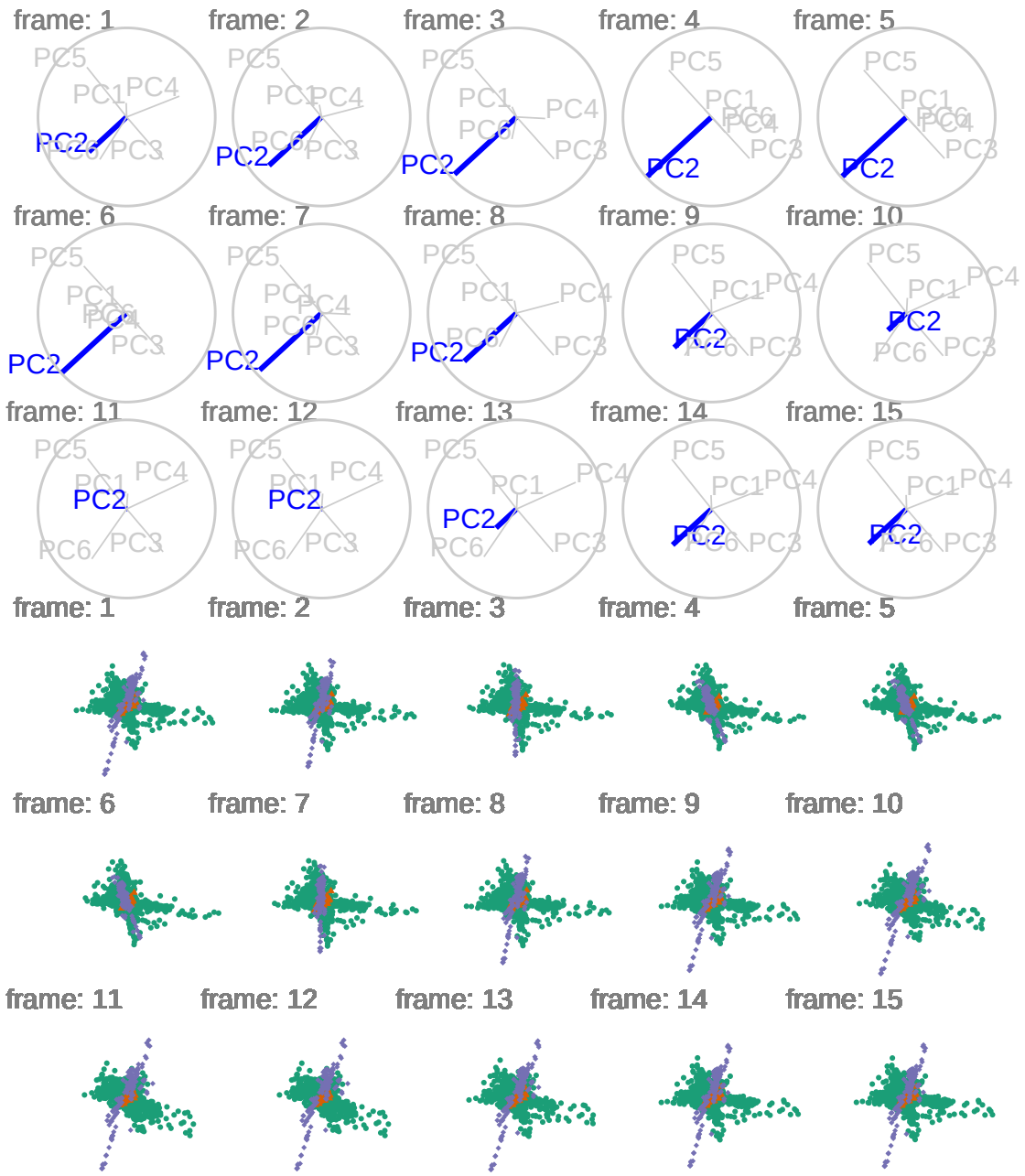

Figure 1.7: *DIS cluster, a radial manual tour of PC2. Colored by experiment type: 'DIS HERA1+2' in green, 'dimuon SIDIS' in purple, and 'charm SIDIS' in orange. The structure of previously described plane of DIS HERA (green) and nearly orthogonal dimuon SIDIS (purple) is present, however, the manipulating PC2 does not give a head-on view of either, a less useful manual tour than that of PC6. A dynamic version can be viewed at* `https://nspyrison.netlify.com/thesis/discluster_manualtour_pc2/`.

1.7 Source code and usage

Use the below code as a guide for installation and finding the vignette. The vignette offers a less technical discussion opting to focus on code usage and goes through a couple more use cases. If you prefer to follow along with the example in the algorithm then simplified code is also listed below.

```
# devtools::install_github("nspyrison/spinifex") # Development version
install.package("spinifex")

# Also see vignette:
vignette("spinifex") # vignette 'spinifex' not found

## manual tour of std flea from holes-index:
library(spinifex)
f_dat  <- tourr::rescale(flea[,1:6])
f_cat  <- factor(flea$species)
f_path <- save_history(f_dat, guided_tour(holes()))
f_bas  <- matrix(f_path[,, max(dim(f_path)[3])], ncol=2)
f_mvar <- 5
f_lab  <- colnames(f_dat)

# View the basis
view_basis(f_bas, data = f_dat, lab = f_lab)
# View the manip space
view_manip_space(basis = f_bas, manip_var = f_mvar, lab = f_lab)
# Play animation as HTML5 widget using plotly
play_manual_tour(data = f_dat, basis = f_bas, manip_var = f_mvar,
                 col = f_cat, angle = f_angle)
```

1.7.1 Acknowledgments

This article was created in *R* (R Core Team, 2018), using *bookdown* (Xie, 2016) and *rmarkdown* (Xie, Allaire, and Grolemund, 2018), with code generating the examples inline. The source files for this article be found at github.com/nspyrison/confirmation/. The source code for the *spinifex* package can be found at github.com/nspyrison/spinifex/.

1.8 Discussion

Tours, the dynamic linear projection of multivariate data, is an important aspect of data visualization extending the display of data-space as data dimensionality increases. This research has modified the algorithm producing manual tours, applied this functionality in *R* and offers extends the graphics offerings that can be used to display tours. The paragraphs below explore how this work might be extended.

Future research on the algorithm would include extending it for use in 3D projections. The addition of another dimension theoretically allows for improved perception. This could explore interactions in immersive virtual reality or mixed reality, which may further allow for a better perception of structure and aid in higher-dimensional function visualization. Functions with many parameters suffer from the same dimensionality problem as data while their possible values lie on a plane of values rather than discrete points. Occulation, or the closer surface blocking further surfaces, will likely be an issue that may be alleviated by the use of wire mesh, changing opacity, or looking at sections of the projections (Furnas and Buja, 1994).

The *tourr* package provides many other geometric displays with the `tourr::display_*()` family. These geometric options could be integrated into the *ggplot2* framework for display on *plotly* and *gganimate*. Additionally, the *animation* package Xie et al. (2018) could be implemented for another graphics framework. However, *animation* builds from base graphs while *spinifex* utilizes *ggplot2* graphics.

The Givens rotations and Householder reflections as outlined in Buja et al. (2005) could also be added. Currently, Gram-Schmidt is the only form of frame interpolation used (not used in manual tours). In a Givens rotation, the x and y components (for example

$\theta = 0, pi/2$) of the in-plane rotation are calculated separately and would be applied sequentially to produce the radial rotation. Householder reflections define reflection axes to project points on to the axes and generate rotations.

Having a script only interaction with tours causes a significant barrier to entry. To a lesser extent, *plotly* offers some static interactions with the contained object, such as tooltips, brushing, and linking without communicating back to the R console. The development of a dynamic graphical user interface, perhaps with the use of a *shiny* (Chang et al., 2018) application, would mitigate the barrier to entry, allow for more rapid analysis, and offer an approachable demo tool. The user could easily switch between variables to control, adjust interpolation step angle, or flag/save specific frame basis sets.

Bibliography

Buja, A, D Cook, D Asimov, and C Hurley (2005). "Computational Methods for High-Dimensional Rotations in Data Visualization". en. In: *Handbook of Statistics*. Vol. 24. Elsevier, pp.391–413. http://linkinghub.elsevier.com/retrieve/pii/S0169716104240147 (visited on 04/15/2018).

Chang, W, J Cheng, JJ Allaire, Y Xie, and J McPherson (2018). *shiny: Web Application Framework for R*. https://CRAN.R-project.org/package=shiny.

Cook, D and A Buja (1997). Manual Controls for High-Dimensional Data Projections. *Journal of Computational and Graphical Statistics* **6**(4), 464–480. (Visited on 04/15/2018).

Cook, D, A Buja, J Cabrera, and C Hurley (1995). Grand Tour and Projection Pursuit. en. *Journal of Computational and Graphical Statistics* **4**(3), 155. (Visited on 05/27/2018).

Cook, D, U Laa, and G Valencia (2018). Dynamical projections for the visualization of PDFSense data. *Eur. Phys. J. C* **78**(9), 742.

Cook, D, DF Swayne, and A Buja (2007). *Interactive and Dynamic Graphics for Data Analysis: With R and GGobi*. en. Google-Books-ID: 34DL7lR_4CoC. Springer Science & Business Media.

Furnas, GW and A Buja (1994). Prosection Views: Dimensional Inference through Sections and Projections. *Journal of Computational and Graphical Statistics* **3**(4), 323–353. (Visited on 04/15/2018).

Lubischew, AA (1962). On the use of discriminant functions in taxonomy. *Biometrics*, 455–477.

Pedersen, TL and D Robinson (2019). *gganimate: A Grammar of Animated Graphics*. http://github.com/thomasp85/gganimate.

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

Sievert, C (2018). *plotly for R*. https://plotly-book.cpsievert.me.

Wang, BT, TJ Hobbs, S Doyle, J Gao, TJ Hou, PM Nadolsky, and FI Olness (2018). Mapping the sensitivity of hadronic experiments to nucleon structure. *Physical Review D* **98**(9), 094030.

Wickham, H, D Cook, H Hofmann, and A Buja (2011). **tourr** : An *R* Package for Exploring Multivariate Data with Projections. en. *Journal of Statistical Software* **40**(2). (Visited on 11/23/2018).

Xie, Y (2016). *bookdown: Authoring Books and Technical Documents with R Markdown*. Boca Raton, Florida: Chapman and Hall/CRC. https://github.com/rstudio/bookdown.

Xie, Y, JJ Allaire, and G Grolemund (2018). *R Markdown: The Definitive Guide*. Boca Raton, Florida: Chapman and Hall/CRC. https://bookdown.org/yihui/rmarkdown.

Xie, Y, C Mueller, L Yu, and W Zhu (2018). *animation: A Gallery of Animations in Statistics and Utilities to Create Animations*. https://yihui.name/animation.