APPLICATION

pavo: an R package for the analysis, visualization and organization of spectral data

Rafael Maia¹*, Chad M. Eliason¹, Pierre-Paul Bitton², Stéphanie M. Doucet² and Matthew D. Shawkey¹

¹Department of Biology, Integrated Bioscience Program, University of Akron, Akron, OH, 44325-3908, USA; and ²Department of Biological Sciences, University of Windsor, 401 Sunset Avenue, Biology Building, Windsor, Ontario, N9B 3P4, Canada

Summary

1. Recent technical and methodological advances have led to a dramatic increase in the use of spectrometry to quantify reflectance properties of biological materials, as well as models to determine how these colours are perceived by animals, providing important insights into ecological and evolutionary aspects of animal visual communication.

2. Despite this growing interest, a unified cross-platform framework for analysing and visualizing spectral data has not been available. We introduce pavo, an R package that facilitates the organization, visualization and analysis of spectral data in a cohesive framework. pavo is highly flexible, allowing users to (a) organize and manipulate data from a variety of sources, (b) visualize data using R’s state-of-the-art graphics capabilities and (c) analyse data using spectral curve shape properties and visual system modelling for a broad range of taxa.

3. In this paper, we present a summary of the functions implemented in pavo and how they integrate in a workflow to explore and analyse spectral data. We also present an exact solution for the calculation of colour volume overlap in colourspace, thus expanding previously published methodologies.

4. As an example of pavo’s capabilities, we compare the colour patterns of three African glossy starling species, two of which have diverged very recently. We demonstrate how both colour vision models and direct spectral measurement analysis can be used to describe colour attributes and differences between these species. Different approaches to visual models and several plotting capabilities exemplify the package’s versatility and streamlined workflow.

5. pavo provides a cohesive environment for handling spectral data and addressing complex sensory ecology questions, while integrating with R’s modular core for a broader and comprehensive analytical framework, automated management of spectral data and reproducible workflows for colour analysis.

Key-words: animal communication, colour, colourspace, receptor noise, reflectance, sensory ecology, spectrometry, visual model

Introduction

The role of colouration and colour vision in animal communication has been a fundamental question in evolutionary biology for many decades (Darwin 1859, 1896; Poulton 1890; Bennett & Théry 2007). Studies on visual communication have shed light on various aspects of natural (Chittka & Menzel 1992) and sexual selection (Hill 2002), and how these interact (Kemp et al. 2009). It is also an ideal system for truly integrative biological research, spanning from the optical processes generating colour (Shawkey et al. 2009), hormonal and genetic mechanisms regulating phenotype (Muller & Eens 2009), physiological processes involved in perceiving the signal (Hart 2001), and its adaptive and evolutionary patterns (Badyaev & Hill 2003; Darst et al. 2006).

However, ‘colour’ refers to a sensory experience, not an objective quantity, and the realization that animals can vary quite considerably in their visual system and how they process this information prompted two important methodological advances. First, it highlighted the need for an objective quantification of the energy reflected at different wavelengths, as a first approximation of a ‘receiver-independent’ measure of an organism or object’s colour (Endler 1993; Eaton & Lanyon 2003; Bennett & Théry 2007). Over the last 20 years, the rising popularity of portable spectrometers has made objective quantification of the spectral properties of animal and plant integuments commonplace (Endler 1990; Eaton & Lanyon 2003; Andersson & Prager 2006). Second, advances in the understanding of perception and processing of colour have allowed analysis of reflectance data using visual models that estimate how animals see and differentiate these colours (Goldsmith 1990; Tovee 1995; Vorobyev & Osorio 1998; Vorobyev et al. 1998).
Because of these advances, a cohesive framework for working with and analysing colour from reflectance data is needed. Output file types from spectrometer manufacturers are not standardized, and though existing software programs have provided helpful implementations of several methodologies (Gomez 2006; Stoddard & Prum 2008; Montgomerie 2006), they are often limited in the number or types of methodologies implemented, the types of data they can import and process, or the platforms in which they are available. Moreover, many are proprietary and/or closed, hindering customization (for a review of available software, see Montgomerie 2006). Furthermore, once the relevant colour data are extracted, they often require additional conversion and export into statistical software for analyses, which precludes protocol standardization across laboratories, batch processing, and automation of workflows.

Here, we introduce pavo—a package for R (R Development Core Team 2013) that addresses these problems by providing a flexible, yet cohesive, environment in which researchers can organize, analyse and visualize colour data generated by spectrometry. R is open source and multi-platform and is rapidly becoming the working language for scientific programming and data analysis, particularly in ecology and evolution (e.g. Paradis et al. 2004; Bolker 2008). pavo incorporates R’s flexibility using object classes that can seamlessly interpret each other, providing functions that can be used to import, explore, process, and analyse spectral colour data under a variety of user-defined models. We propose that combining these procedures under a coherent framework not only streamlines workflow, but also allows data to be explored and manipulated in ways that can be used to visualize patterns, obtain information, and develop and test hypotheses (Fig. 1).

The pavo package

The stable release of pavo is available from CRAN (http://CRAN.R-project.org/package=pavo) for direct installation from R, and the development version is available from github (https://github.com/rmaia/pavo). pavo was developed with three main workflow stages in mind (Fig. 1): organization of spectral data by inputting raw files and processing their spectral content; visualization of the output, including exploratory capacities to identify further required manipulations and previously unconsidered patterns; and analysis of data from the spectral shape of reflectance curves or by incorporating receiver psychophysiology in visual models. As noted by Bennett & Théry (2007) and others (Andersson & Prager 2006; Montgomerie 2006), though spectral data have become commonplace in studies of animal colouration, it is easy to obtain poor-quality or inaccurate data. Therefore, a workflow for spectral

Fig. 1. Example pavo workflow, highlighting its main functions and plotting capabilities.
colour data analysis has to go beyond a ‘plug and chug’ implementation, requiring thorough exploratory investigation. With this in mind, pavo takes advantage of R’s object-oriented programming environment to implement modular functions that allow each of these steps to be explored and to work cohesively. Below we outline pavo’s organization, visualization and analysis capabilities, followed by a detailed worked example using African glossy starlings to demonstrate some of its functionalities.

**ORGANIZATION**

Spectral data are stored in pavo and recognized for its functions by use of a new object class, ‘rspec’, which inherits methods from data.frame. Objects of class rspec are characterized by having individual reflectance spectra as columns of the data frame, with a first column containing the associated wavelength values. Raw spectral data can be imported using the function getspec, which currently supports data from a variety of software (including Ocean Optics OOIBase and SpectraSuite files, Avantes AvaSpec and CRAIC). In addition, spectral data that have been previously compiled (into a spreadsheet, for example) can be imported into R and converted to rspec objects using the as.rspec function.

The use of dedicated R object classes allows generic functions such as plot and summary to identify the object as a particular type of data frame and interpret it accordingly (see below). The class ‘vismodel’ is used to interpret spectral data that have been processed through one of the visual models implemented and also stores information on how it was generated (e.g. the visual phenotype, background and illuminant used, and any transformations applied; see below). Additionally, the ‘tcs’ class refers to tetrahedral colourspace models (Endler & Mielke 2005; Stoddard & Prum 2008), and the summary function can be used to extract summary variables, such as the colour volume or hue span (Stoddard & Prum 2008) for groups and subsets of points.

It is common when collecting spectral data to take multiple measurements from the same sample, averaging these to avoid sampling error (Quesada & Senar 2006). pavo provides the aggspec function for this purpose, as well as the procspec function for noise removal via smoothing, and transformations to standardize and clean spectral data.

**VISUALIZATION**

With pavo installed and loaded, the plot function recognizes rspec objects and plots them accordingly—interpreting the first column as wavelengths (usually using it as the x axis) and the remaining columns as reflectance values (y axis) for individual spectra. Several plotting options for multiple spectra are implemented (Fig. 1). In addition, the aggplot function provides plotting capabilities for among-spectra summary statistics plotting (Fig. 1), with a similar syntax to aggspec. pavo also offers exploratory plotting capabilities that can be combined with data processing and formatting, such as explorespec (for visualizing groups of spectra) and smoothplot (for choosing smoothing parameters; Fig. 1).

Finally, pavo offers plotting capabilities for the avian tetra-chromatic colourspace model (Stoddard & Prum 2008; Endler & Mielke 2005) through the tcspplot and projplot functions (Fig. 1, see below).

**ANALYSIS**

The summary function can be applied to rspec objects to extract several objective (‘receiver-independent’) reflectance shape variables, relevant to specific or universal mechanisms of colour signalling (e.g. pigments, structures) or colour perception (spectral intensity, location and purity). Description and discussion of these variables can be found in Andersson & Prager (2006) and Montgomerie (2006), as well as in the package vignette. Additionally, the function peakshape provides descriptors of spectral peaks, such as the wavelength of maximum reflectance and the full width at half-maximum, and can be fine-tuned to extract information from specific areas of the curve. This implementation can be useful when the spectral curve has multiple peaks or a complex shape (e.g. the UV peak of carotenoid curves, Fig. 1).

pavo also allows the easy production of models that incorporate the visual system of the receiver through the vismodel function. Models can be calculated incorporating the visual phenotype (cone absorbance), background colour, and ambient illuminant (Vorobyev & Osorio 1998). Several avian receptor phenotypes (Hart 2001; Endler & Mielke 2005) are implemented as options, but user-defined receptor data from any taxon can be used as model input. Further, the sensmodel function implements the calculation of cone absorbance curves based on peak sensitivity information (available from the literature, for example Hart 2001) and can also include oil droplet and ocular transmission information in the calculations (Govardovskii et al. 2000; Hart & Vorobyev 2005).

Visual models can be calculated in terms of absolute photon catches, in which case the receptor noise model can be used to infer contrast between colours (implemented in the function coldist Vorobyev & Osorio 1998), or in relative cone stimulation, in which case the model reduces to a colourspace model represented in $n−1$ dimensions (where $n$ is the number of different receptors involved in colour vision; Goldsmith 1990; Endler & Mielke 2005; Stoddard & Prum 2008). Absolute or relative cone stimulation can be selected by the logical argument relative from the vismodel function. In the case of the avian tetrahedral colourspace, several additional variables can be calculated based on spherical coordinates which represent the hue angles and saturation (the distance from the achromatic centre; see Stoddard & Prum 2008) by calling the tcs function. This function generates an object of class tcs; a summary call from a tcs object will return summary statistics described in Stoddard & Prum (2008) for sets of points (see below).

pavo also builds upon previously described visual model methods. For example, Stoddard & Stevens (2011) presented the useful technique of calculating the overlap between the
volumes defined by two sets of points in colourspace. They used this metric to quantify mimicry (Stoddard & Stevens 2011; Stoddard 2012), such that a greater volume overlap would indicate greater overall colour similarity. Given the complexity of calculating the intersection of three-dimensional convex hulls, Stoddard & Stevens (2011) used a Monte Carlo approach to estimate the degree of volume overlap. *pavo*, instead, provides the exact solution for the calculation of the intersection of colour volumes using a method originally implemented to calculate the overlap between multidimensional niches (Villéger et al. 2011) through the computational geometry capabilities available from the *rcdd* package (Geyer et al. 2012) (For performance and precision comparison, see Supporting Information).

**Worked example: colour divergence in glossy starlings (Sturnidae)**

Glossy starlings from the African clade have bright and diverse iridescent colours likely used in courtship displays and social competition (Rubenstein & Lovette 2009). Here, we demonstrate some of *pavo‘*s capabilities through a worked example to compare the colours of a monophyletic clade of three glossy starling species, two of which (*Lamprotornis chloropterus* and *L. elisabeth*) have recently diverged and whose status as full species is debated (Gill & Donsker 2012; Craig et al. 1998; Lovette & Rubenstein 2007). The data used for this example have been deposited online and is available from the Dryad Database (Maia et al. 2013). Further information and a ‘how-to’ of *pavo‘*s main functions, including the formulae used for the calculations, can be found in detail in the package vignette – available online from CRAN or directly from R using the command vignette(’pavo‘).

**STEP 1: ORGANIZATION AND PROCESSING**

The data consist of reflectance spectra in Avantes ‘.ttt’ output format, which in this example are located in the folder ‘/Desktop/glossystarlings’ (by default, the current working directory and the ‘.txt’ extension are used). The files are named to indicate the species, bird ID, plumage patch and measurement ID (for example, ‘LAAC.14423.belly.001.ttt’ would be the first measurement taken from the belly of individual 14423, a *Lamprotornis acuticauda* specimen). This is important because *pavo* uses the file name (minus the file extension) to label the columns of the *rspec* object with the imported reflectance spectra. We recommend that users carefully consider and adhere to a rigorous file naming scheme to make subsetting and data manipulation easier (see Andersson & Prager 2006 for a useful discussion and suggestions in this regard). Additionally, we suggest that sample identity labels should have the same number of characters, which simplifies character string manipulation and subsetting based on partial matching.

We measured reflectance spectra from 11 plumage patches (see Fig. 2 for list) of four males from museum specimens of each of the three species. Three measurements from different locations within each patch were collected. We used the *getspec* function to load these 396 raw spectral data files, then used the *aggspec* function to average the spectra within patches (as determined by the *by* argument) and the *procspec* function to remove electrical noise arising from the spectrometer (using local polynomial regression fitting, or *loess*) with the following annotated lines of code:

```r
# get raw data
> specs <- getspec(where = ‘~/Desktop/glossystarlings’, ext = ‘.ttt’, lim = c(300, 700))
396 files found; importing spectra
=================================================================
# average by groups of 3 spectral curves
> specs <- aggspec(specs, by = 3, FUN = “mean”)
# remove electrical noise using
# Gaussian smoothing
> specs <- procspec(specs, opt = “smooth”)
processing options applied:
  smoothing spectra with a span of 0.25
```

**STEP 2: VISUALIZATION**

Next, we plotted spectra contained in the resulting *rspec* object. We used *aggplot* to visualize the mean reflectance curves for each body part from each species, as can be seen in the example below for the ‘belly’ body patch (Fig. 2b):

```r
# subset wavelength column and the
# 12 spectra from the belly patch
> specs.belly <- subset(specs, “belly”)
# extract first 4 characters from column names
# (species labels)
> spp <- substr(names(specs.belly), 1, 4)
# average and plot spectral data by species
> aggplot(specs.belly, by = spp)
```

**STEP 3: ANALYSIS**

**Colour distances**

To explore how these colours may be perceived by birds, we first used the *vismodel* function, which takes into account avian visual sensitivities (sensory phenotype), to calculate the quantum catches for each photoreceptor. *pavo* allows for user-defined cone sensitivities, but also incorporates available data from several taxa, including the European starling (*Sturnus vulgaris*, visual = “star”, Hart et al. 1998), which we used for our example. We used the *relative* = FALSE option from the *vismodel* function to obtain raw photon catch values for the four avian photoreceptor classes (*usml*), suitable for calculating chromatic distances ΔS, Vorobyev et al. (1998).

*vismodel* can return either the calculated quantum catches ($Q_i$, the default) or values transformed according to Fechner’s law ($f$; the signal being proportional to the logarithm of the quantum catch), as determined by the argument *qcatch* (not used in the code below, hence the default, *Qi*, is returned). It can also apply the von Kries transformation (normalizing by
the stimulus’ background to account for receptor adaptation) through the vonkries argument (which defaults to FALSE). These settings are stored as arguments in the vismodel object, are referenced by functions downstream in the workflow (e.g. coldist, below), and can be called using the summary function:

```r
>vm.star <- vismodel(specs, visual="star", relative = FALSE)
>summary(vm.star)
```

The five visual model options are stored as arguments in the vismodel object, are referenced by functions downstream in the workflow (e.g. coldist, below), and can be called using the summary function:

```r
visual model options:
* Quantal catch: Qi
* Visual system: star bt.dc
* Illuminant: ideal, scale = 1 (von Kries color correction not applied)
* Background: ideal
* Relative: FALSE
```

Second, we used the coldist function to calculate colour distances with receptor noise based on the relative photoreceptor densities (Vorobyev & Osorio 1998). We used relative cone abundances (arguments n1, n2, n3 and n4) for the European starling (Hart et al. 1998), and the ν argument was set to 0.1 to give a 1/D-cone Weber fraction (calculated as \( \sqrt[3]{n} \)) of approximately 0.05 (Vorobyev & Osorio 1998; Vorobyev et al. 1998). The results of coldist give colour distances (chromatic, ΔS, shown in Fig. 2; and achromatic, ΔL) between all possible combinations of plumage patches in a vismodel object and can be further subsetted (using the subset argument) to focus on comparisons of interest (e.g. colour difference between homologous patches of two species, or between colour patches and a given background).

This argument employs regular expressions functionality and thus allows for partial string matching of row names to the rule specified in the argument. In this case, if we want comparisons between a single type of patch, the subset argument will have two values to match both patch1 and patch2 columns:

```r
>deltas <- coldist(vm.star, n1=1, n2=1.38, n3=3.34, n4=3.46, v=0.1)
>subset only comparisons between wing spectra
>deltas.wing <- coldist(vm.star, n1=1, n2=1.38, n3=3.34, n4=3.46, v=0.1, subset=c("Wing", "Wing"))
>head(deltas.wing, 3)
```

We can see from Fig. 2a that the recently divergent *L. chloropterus* and *L. elisabeth* have accumulated similar levels of colour disparity as they have to their sister species, *L. acuticaudus*. Considering a value of 1 as a threshold for Just Noticeable Differences (JNDs, Fig. 2a dashed line), nearly all plumage patch comparisons yield discernible colours, both within the *L. chloropterus – L. elisabeth* subclade as well as compared with the *L. acuticaudus* outgroup.

### Spectral analysis

Another strength of pavo is its ability to compare different colour metrics (Butler et al. 2011), such as those extracted directly from the spectral curves (Fig. 2b) and thus do not consider
receiver visual system phenotype (see Andersson & Prager (2006) for discussion). As many as 23 objective colourimetric variables are calculated by the summary of an rspec object (for a complete description and summary of formulae, see the package vignette and Montgomerie 2006), but the subset argument can be used to filter these variables. If subset = TRUE, the variables described in Andersson & Prager (2006) are returned, but a string of variable names can also be used as a filter. For our example, we extracted and plotted an objective metric of hue (wavelength of peak reflectance, Fig. 2c) for the most contrasting body patch (‘belly”, Fig. 2a) in the visual model analysis:

```r
# extract colour variables
> colr <- summary(specs.belly, subset=TRUE)
> head(colr, 3)

   B2     S8    H1
LAAC.260046.Belly 18.92453 1.518254 520
LAAC.264607.Belly 12.31255 1.257920 481
LAAC.347959.Belly 11.20595 1.233001 486

# extract hue (H1)
> hue <- colr$H1

# extract species names
> spp <- substr(rownames(colr), 1, 4)

# make box plots of hue (H1) by species
> boxplot(hue ~ spp)
```

Avian tetrahedral colourspace

To analyse these spectra in avian tetrahedral colourspace, we used vismodel again, but now specifying relative=TRUE to indicate that {usml} values should be scaled to sum of unity for each observation. We then used the voloverlap function to calculate the volumes occupied by each species’ plumage patches, as well as their overlap. The plot=TRUE option provides a useful graphical representation of the overlap in colourspace (Fig. 3).

```r
> vm.star.rel <- vismodel(specs, visual="star", relative=TRUE)
> tcs.star <- tcs(vm.star.rel)

#subset points from the two species
> tcs.lael <- subset(tcs.star, "LAELE")
> tcs.lacl <- subset(tcs.star, "LACL")
> voloverlap(tcs.lael, tcs.lacl, plot=TRUE)
```

Conclusions and future directions

pavo implements colour analyses for reflectance data based on both spectral properties and visual system receptor stimulation, as well as a modular framework that allows the assumptions and parameters of such models to be tuned as necessary. As such, it is particularly suited to compare and validate the use of different colourimetrics (Butler et al. 2011), determine the influence of various assumptions in visual models, or help choose parameter values that would introduce the least bias in visual models (Hanley 2013). pavo can also be used to investigate the information content of colour signals using receiver-specific visual systems (Pike et al. 2011) and answer questions pertaining to signal production mechanisms (e.g. comparing the relative contribution of pigments and structural colours in various integuments; D’Alba et al. 2012). Currently, the visual modelling functions implemented in pavo already allow for any number of photoreceptors to be considered, but colour distances can so far only be calculated for di-, tri- and tetrachromats. Implementing extensions of the Vorobyev & Osorio (1998) model will allow pavo to calculate colour distances considering the visual systems of organisms such as butterflies (e.g. six photoreceptors in Pierids and Papilio butterflies; Morehouse & Rutowski 2010, Arikawa 2003) and mantis shrimps (Order Stomatopoda; 16 photoreceptors; Cronin et al. 1993).

Furthermore, as an R package, pavo provides a direct integration with the growing number of statistical procedures that the language incorporates, facilitating a streamlined workflow from initial data input to final analysis. It therefore allows the inclusion of colour analysis from spectral data in open and reproducible workflows, which can be directly automated and adjusted for consistency. Finally, pavo will continue to incorporate reference data for illuminants, backgrounds, photoreceptors, and other components of visual systems (e.g. ocular media; Siebeck & Marshall 2001) as they become available, providing a repository of spectral and colour data for physiological models across a broad range of taxa.

Citation of methods implemented in pavo

Most of the methods implemented in pavo have been thoroughly described in their original publications, to which users should refer for details and interpretation. For reflectance shape variables (‘objective colourimetrics’) and their particular relation to signal production and perception, see Andersson & Prager (2006) and Montgomerie (2006). Visual models based on photon catches and receptor noise are detailed in the study by Vorobyev & Osorio (1998) and Vorobyev et al. (1998), and photoreceptor sensitivity curve estimation in the study by Govardovskii et al. (2000) and Hart & Vorobyev (2005). For tetra-
hedral colourspace model implementations and variable calculations, see Endler & Mielke (2005) and Stoddard & Prum (2008), and for colour volume overlap, see Stoddard & Stevens (2011) and Stoddard (2012). Users of the functions that apply these methods should cite the original sources as appropriate, along with pavo.

Data accessibility

Data for the examples have been deposited in the Dryad repository: http://dx.doi.org/10.5061/dryad.298b1

Acknowledgements

We would like to thank two anonymous reviewers for comments to a previous version of this manuscript, and Jarrod D. Hadfield and Mary Caswell Stoddard for sharing code that helped us develop some of pavo’s capabilities. This work was supported by the NSF grant DEB-1210650, AMNH Chapman research grant and Sigma Xi GIAR (R.M.), NSF grant EAR-1251895, HSF grant RGY0083, AFOSR grant FA9550-06-1-0159 (M.D.S.), the University of Akron (M.D.S., R.M., C.M.E.), NSERC Discovery grant and Equipment grant (S.M.D), and NSERC Graduate Scholarship (P.-P.B.).

References


Received 3 January 2013; accepted 13 May 2013
Handling Editor: Andrew Tatem

**Supporting Information**

Additional Supporting Information may be found in the online version of this article.

**Appendix S1.** Exact determination of volume overlap.

**Figure S1.** Results of Monte Carlo simulations of volume overlaps for various 3-dimensional shapes (a: cube, b: regular tetrahedron, c: cuboid) compared to the exact result.

**Figure S2.** Results of Monte Carlo simulations of proportional volume overlaps for arbitrary polyhedra with varying overlap volumes (a: 40%, b: 16%, c: 3%) compared to the exact calculations.