Mathematica Graphics

and general review

You are always smarter than the computer!
Mathematica interface

Package loading statements
Menus
Cell indicators
Notebook

Useful menus:
Insert
Format
Evaluation
Palettes
Basic Mathematica expressions

**Functions** - entities that take in input and return output. Functions always have the form of a name (all one word, capitalized first letters of words) and one pair of square brackets into which the input is entered. If a function is active, its name will appear in black type.

Function[input]

**Variables** - named holders that you create. You can put nearly anything into a variable using the equals sign (=). If a variable has not yet been used in a session, its name will appear in blue. Once it has been filled with something its name will appear in black. The values of variables can be changed at any time.

\[a = 20000\]
\[b = \text{Mean}[\text{proc}]\]
Basic Mathematica expressions

**Constants** - special “variables” that are already assigned by Mathematica.

Pi, Infinity, Degree, E

**Operators** - symbols for mathematical operations

+, =, *, /,.

**Comparative operators** - symbols for mathematical comparisons

== (are two things equal?)
> (greater than)
< (less than)
<= (greater than or equal)
Basic Mathematica expressions

**Numbers versus strings** - Numbers are entities (often stored in a variable) that are treated as mathematical numbers. Strings are collections of characters (some of which could be numerals) that are not treated as numbers. To create a string, the characters must be put in quotes.

```
1    (this is a number)

"1"  (this is a string)

a    (this is a variable)

"a"  (this is a string)
```

**Options** - Options are information given to functions, entered between commas after the required input for a function. Often options have a name and a value separated by a right-pointing arrow: `PlotRange -> {0, 1}`
Kinds of brackets and punctuation

Square brackets [ ] - functions (single pair of square brackets around the input part of the function). variables that contain a list (double square brackets around numbers that tell which part of the list to show)

Parentheses ( ) - comments when used with asterisks (* comment here *). ordinary mathematical operations. (a + b) * 3

Curly brackets {} - lists (each set defines beginning and end of list, where list elements are separated by commas), and nested define rows or rows within rows within a list. iterators in table or loop function.

Commas , - separate items in a list, or input pieces in a function

Semicolons ; - when placed at end of command causes output not to be printed to screen. two semicolons together ;; means full range (scores[[1;;10,3]])

Slashes / - one slash is operator for division. // at end of command specifies formatting command to follow.
Formatting statements

Statements placed at the end of a line of code that affect how the output is displayed. Formatting statements always have a double-slash at the start and never have brackets.

//N    (forces output into numeric form instead of fractions or symbols)

//MatrixForm (forces matrices to be printed in neat rows and columns)

//TableForm (similar to MatrixForm).

Other items

<< - read something into Mathematica, usually an installed package

`  - Mathematica convention to indicate name of an installed package (goes at end of name).
Tables and other loops

Looping structures repeat things. Like other functions, looping functions have square brackets. They also have a specialized list called an “iterator” that comes at the end inside curly brackets:

\[
\text{Table}\left[\text{stuff to repeat}, \{\text{iterator}\}\right]
\]

The iterator usually has a variable that is used in the “stuff to repeat” and some numbers that tell it where to begin, where to end, and how big the steps should be.

\[
\text{Table}[x, \{x, 100\}] \quad \text{(* prints variable x, where x iterates from 1 to 100 in steps of 1 *)}
\]

\[
\text{Table}[x, \{x, 1, 100, 1\}] \quad \text{(* does exactly the same as above, 1 to 100 in steps of 1 *)}
\]

\[
\text{Table}[x, \{x, 50, 100, 10\}] \quad \text{(* prints variable x, where x iterates from 50 to 100 in steps of 10 *)}
\]

\[
\text{Table}[\text{scores}[\{x\}], \{x, \text{Length}[\text{scores}]\}] \quad \text{(* prints the xth row of scores for however many rows there are in scores. *)}
\]
Basic components of GMM

In plain language first....

Get coordinates

Procrustes superimposition

Principal components analysis

  calculate “residuals” (each set of coordinates - consensus)

  calculate covariance matrix

  get eigenvalues -- amount of variance (variation) associated with each PC

  get eigenvectors -- can be used to model shapes. (eigenvector * score + consensus)

  get scores -- address of each object in the PC plot
Basic components of GMM

In Mathematica-ese....

Get coordinates: \( \text{coords=tpsImport["filename"][[1;;,2;;]]} \)

Procrustes superimposition: \( \text{proc=Procrustes[coords,10,2]} \)

Principal components analysis: \( \text{PrincipalComponentsOfShape[proc, \{2,5\},labels]} \)

- calculate “residuals” (each set of coordinates - consensus): \( \text{resids = Table[proc[[x]]-consensus, \{x, Length[proc]\}]} \)

- calculate covariance matrix: \( \text{c=Covariance[resids]} \)

- get eigenvalues -- amount of variance (variation) associated with each PC: \( \text{EigenValues[c]} \)

- get eigenvectors -- can be used to model shapes. (eigenvector * score + consensus) \( \text{EigenVectors[c]} \)
Basic Graphics

Graphics can be constructed of Primitives, Directives, and Display functions

**Primitives** - Functions for basic shapes like points, lines, polygons, cubes, etc. Input usually includes coordinates for where their parts should appear in the graphic.

**Directives** - Instructions for how to display a primitive. Directives include color, size, orientation, opacity, fills, font faces, textures. These are added by creating a list of them joined the primitive by curly brackets.

**Display functions** - Functions that cause graphics primitives to be rendered, or displayed, on the screen. Common display functions include Graphics[], Graphics3D[], Animate[], ListAnimate[]