Interoperability and licensing

Kris Shaffer, Charleston Southern University

Computational approaches to music theory and analysis: Overview and hands-on tutorials
Society for Music Theory Annual Meeting
27 October 2011

Interoperability and licensing

- Researcher makes code available to be used by other researchers
- Researcher makes code available to be modified (for private use and/or distribution) by other researchers

Interoperability and licensing

- Ease of use
- Ease of modification
- Ease of interoperation
- Licensing

Ease of use

- Clear names for variables, subroutines, databases, files, etc.
- Detailed, direct comments in code

```
#!/usr/bin/perl
# Takes an analysis file, and extracts aggregate data from it.
# See www.theory.esm.rochester.edu/rock_corpus for more info.
#
# START END ROMNUM CROOT DROOT KEY AROOT
# 0     1   2      3     4     5   6
#
# Run it like this:
# ./tally.pl [input-file]


for($c=0; $c<12; $c++) {
    $chord_count[$c] = 0;
    $chord_time[$c] = 0;
    for($c2=0; $c2<12; $c2++) {
     $trans[$c][$c2] = 0;
    }
}

$misc_count = $misc_time = $ma = $mi = $dim = $aug = $inv = 0;

open(INFILE, $ARGV[0]);

while(<INFILE>) {   # or use <STDIN> instead of <INFILE>, then "pipe" in input from another
process,
                    # e.g. "... | ./tally.pl"
```

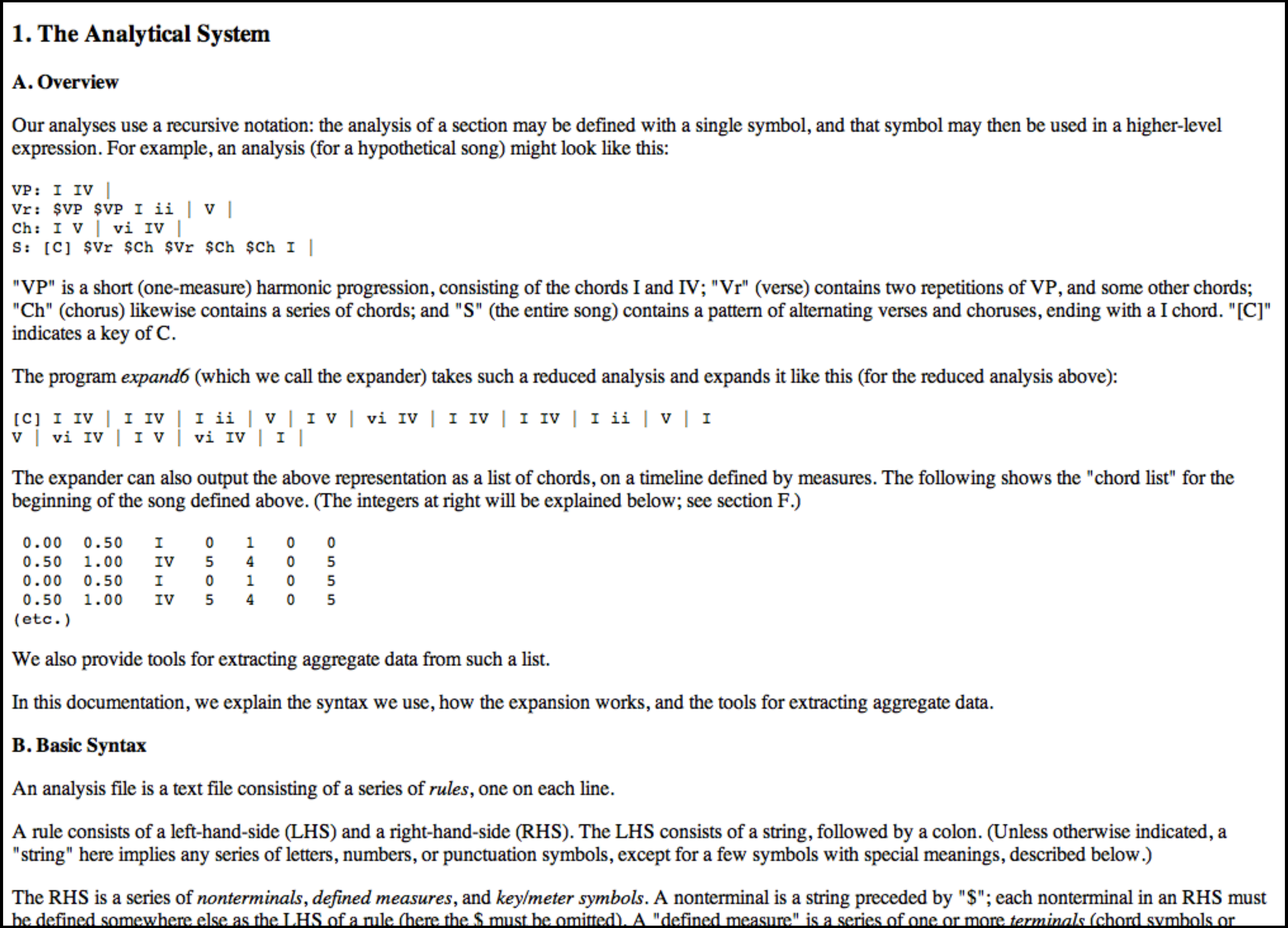
Trevor de Clercq & David Temperley, beginning of *tally.pl* script from tools for “A Corpus Analysis of Rock Harmony” (*Popular Music* 2011).

```
#!/usr/bin/perl

use File::Basename qw/ /;
my %chordProfile;

# create directory for profiles
     mkdir "ChordDistributionProfiles", 0755 or warn "Could not create directory for chord 
distribution profiles: $!";

foreach $movement (@ARGV) {

# create filename for profile
     my $tallyFile = File::Basename::basename $movement;
     $tallyFile =~ s/(\.csv$|\.txt$)//i;
     $tallyFile = $tallyFile . "-profile.csv";

# open source and destination files
     open SOURCE, "<$movement" or die "Could not open $movement: $!";
     open DESTINATION, ">./ChordDistributionProfiles/$tallyFile" or die "Could not create 
$tallyFile: $!";

# set profile & probability values for each root to zero
     foreach (0..11) {
          $chordProfile{$_} = 0;
          }
     $totalchords = 0;
```

Kris Shaffer, beginning of *chordsToProfile.pl* script from Profiler, a set of scripts for processing harmonic analysis data (see http://kris.shaffermusic.com).

Ease of use

- Clear names for variables, subroutines, databases, files, etc.
- Detailed, direct comments in code
- Standard data and file formats—match existing analyses & software
- Cross-platform, freely available language
- Clear, complete documentation



1. The Analytical System

A. Overview

Our analyses use a recursive notation: the analysis of a section may be defined with a single symbol, and that symbol may then be used in a higher-level expression. For example, an analysis (for a hypothetical song) might look like this:

```
VP: I IV |
Vr: $VP $VP I ii | V |
Ch: I V | vi IV |
S: [C] $Vr $Ch $Vr $Ch $Ch I |
```

"VP" is a short (one-measure) harmonic progression, consisting of the chords I and IV; "Vr" (verse) contains two repetitions of VP, and some other chords; "Ch" (chorus) likewise contains a series of chords; and "S" (the entire song) contains a pattern of alternating verses and choruses, ending with a I chord. "[C]" indicates a key of C.

The program *expand6* (which we call the expander) takes such a reduced analysis and expands it like this (for the reduced analysis above):

```
[C] I IV | I IV | I ii | V | I V | vi IV | I IV | I IV | I ii | V | I
V | vi IV | I V | vi IV | I |
```

The expander can also output the above representation as a list of chords, on a timeline defined by measures. The following shows the "chord list" for the beginning of the song defined above. (The integers at right will be explained below; see section F.)

```
 0.00  0.50   I    0   1   0   0
 0.50  1.00   IV   5   4   0   5
 0.00  0.50   I    0   1   0   5
 0.50  1.00   IV   5   4   0   5
(etc.)
```

We also provide tools for extracting aggregate data from such a list.

In this documentation, we explain the syntax we use, how the expansion works, and the tools for extracting aggregate data.

B. Basic Syntax

An analysis file is a text file consisting of a series of *rules*, one on each line.

A rule consists of a left-hand-side (LHS) and a right-hand-side (RHS). The LHS consists of a string, followed by a colon. (Unless otherwise indicated, a "string" here implies any series of letters, numbers, or punctuation symbols, except for a few symbols with special meanings, described below.)

The RHS is a series of *nonterminals*, *defined measures*, and *key/meter symbols*. A nonterminal is a string preceded by "$"; each nonterminal in an RHS must be defined somewhere else as the LHS of a rule (here the $ must be omitted). A "defined measure" is a series of one or more *terminals* (chord symbols or

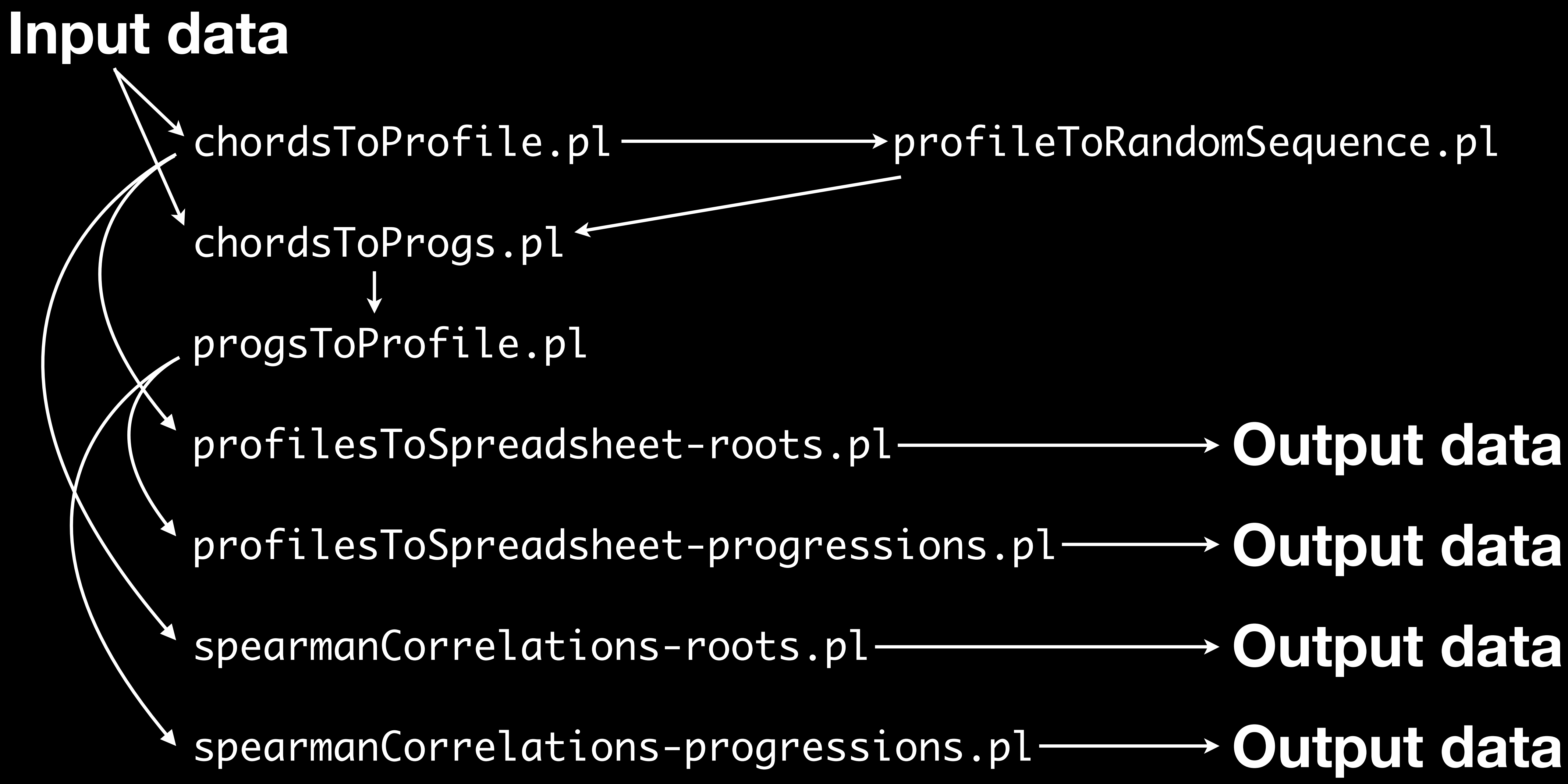
Ease of modification

- All of the above apply—
  the easier to use, the easier to modify for other uses

Interoperability

- Modularity—
  using multiple short scripts rather than a single long application
- UNIX pipeline

```
programOne input.file | programTwo | programThree > output.file
```



Input data

chordsToProfile.pl

profileToRandomSequence.pl

chordsToProgs.pl

progsToProfile.pl

profilesToSpreadsheet-roots.pl

Output data

profilesToSpreadsheet-progressions.pl

Output data

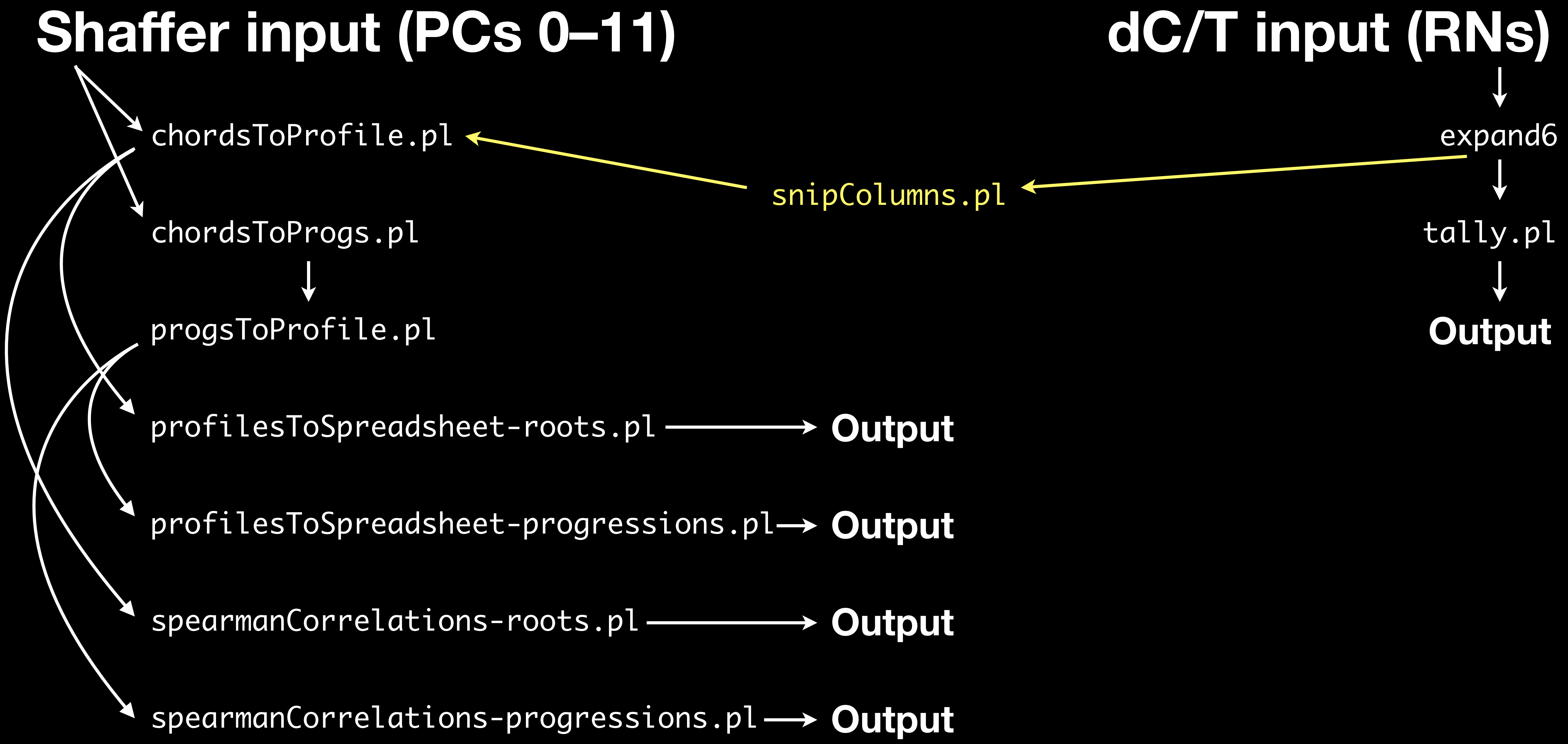
spearmanCorrelations-roots.pl

Output data

spearmanCorrelations-progressions.pl

Output data

Kris Shaffer, Profiler, a set of scripts for processing
harmonic analysis data (see http://kris.shaffermusic.com).



Shaffer input (PCs 0–11)

dC/T input (RNs)

chordsToProfile.pl

chordsToProgs.pl

progsToProfile.pl

profilesToSpreadsheet-roots.pl

profilesToSpreadsheet-progressions.pl

spearmanCorrelations-roots.pl

spearmanCorrelations-progressions.pl

Output

Output

Output

Output

snipColumns.pl

expand6

tally.pl

Output

Data processing streams for Shaffer’s Profiler scripts and
de Clercq/Temperley’s rock harmony analysis scripts.

Licensing

- Making clear what rights *are* and *are not* passed on to other researchers when sharing code

Public domain

- Expired or waived copyright
- Anyone can use public domain material for any purpose
- CC0 license—http://creativecommons.org/about/cc0

Freeware

- Free to use (no cost), but with specific limitations—many rights reserved
- Copyright notice with clear statement of rights granted to user
  (e.g., use but not modification; academic but not commercial use; etc.)

Free software

- Typically grants permission to use, modify, distribute, sell, etc., but with restrictions
- Restrictions protect freedom of users
- Two main types: BSD-style, GPL-style

Free software

- BSD- and GPL-style licenses both preserve freedoms of users for the original program
- GPL-style licenses preserve same freedoms for users of derivative programs, as well

see fsf.org, opensource.org, and gnu.org/licenses/ for more information on free/open-source licenses

Licenses

- Free to use only?
  Freeware license
- Free to modify (and poss. distribute)?
  BSD-style, GPL-style, CC0 (public domain)
- Ensure same freedoms in derivative works?
  GPL-style license

Interoperability and licensing

Kris Shaffer, Charleston Southern University

http://kris.shaffermusic.com