Harpy: user manual for version 0.1.4 (Windows)

Anthony Labarre

April 26, 2010

1 What `harpy` is

In a nutshell, `harpy` is a piece of software that reads in MIDI files, extracts the channel of your choice and produces the corresponding tabs for the diatonic harmonica of your choice.

`harpy` knows about (low) major, natural minor, harmonic minor, Country tuned and Melody Maker blues harps. It also tells you "how difficult" a song is, based on the advanced techniques you need to master to play the song. If you want to practice those techniques, just choose the harp that suits your needs. If you're not good enough yet, you can select harps that avoid them.

`harpy` is available at

`https://sourceforge.net/projects/harpy/`

It is free (as in free beer, and as in free speech), but if you think I deserve something, feel free to donate whatever amount you find appropriate – follow this link or drop me a line to do that. I also welcome comments and suggestions, which can be mailed to me or posted at

`https://sourceforge.net/projects/harpy/forums`

2 Requirements

In its most basic usage, `harpy` does not require anything special. However, if you want to be able to produce LaTeX tabs[1], you will need to install a LaTeX distribution that features `pdflatex`, and a PDF viewer to see the results.

---

[1] Don't worry: you don't need to know a thing about LaTeX.

Although binaries are only available for Windows at this time, `harpy` is intended to and should run on any platform. If binaries for your platform are missing, or if for some reason they do not work, get the source package and install Python (and please, report your problems! I cannot solve them if I don't know about them).

3 Basic usage

`harpy` is a command-line program. Under Windows, type `cmd` in the "Run" dialog that can be selected in the "Start" menu to open the command-line prompt, then use the `cd` command to reach the directory where `harpy` is located. The most basic way of running `harpy` is:

```
harpy.exe [options] midifile
```

(options are described in Section 4) where `midifile` can be a file with extension `.mid` or `.kar`. If the file contains several channels, `harpy` will list them all and ask you which one you're interested in. `harpy` also tries to guess which instrument is used on a given channel. The channel you select is then extracted to another MIDI file, which only has a single channel this time, and on which subsequent actions are carried out.

`harpy` then lists which harps (if any) you can use to play the wanted song or channel. If there are none, it will tell you why, and will try to find transpositions[2] that will allow you to play the song. Either way, it will also tell you how difficult the song will be to play on each harmonica (e.g. whether you need to be able to bend notes or overblow/overdraw them).

Once you got a list of harmonicas you can use, `harpy` will ask you if you want tabs for any of them. Pick your favourite, and `harpy` will write the tab in the format of your choice – simple text file by default, but there is an option to set the desired format: see Section 4 for information on that, and Section 5 for an idea of how your tab will look like.

In case `harpy` is unable to guess channel names, you can always use MidiSplitter (see Section 6) to extract all channels to separate files and hear what they sound like before deciding on which ones `harpy` should be run. The following information may also help you make up your mind:

- drums are always on channel number 9;
- most files I've played around with so far use channel number 1 for bass.

---

[2] I.e. moving every note up or down by the same amount of semitones.

4 `harpy` options

`harpy`'s general behaviour is to stop as soon as it finds satisfying harps: it will stop searching if your song is playable on at least one harp, and if it has to search for satisfying transpositions, it will stop as soon as it finds one. You can, however, use some of its options to force it to produce different results – for example when a song is playable on a bunch of harps, but you'd rather avoid bending.

4.1 Hardness-related options

The following options allow you to control the output of `harpy` and to select advanced techniques that you would like to avoid.

- `-s`, `--sort`: sorts usable harps by increasing hardness: harps that require no special techniques are listed first, then harps that require bending, and finally those that (additionnally) require overblows or overdraws.
- `--avoid-bends`: discard harps that require bending; using this option will force `harpy` to look for satisfying transpositions if bends cannot be avoided on the original song.
- `--avoid-overblows`: discard harps that require overblows; using this option will force `harpy` to look for satisfying transpositions if overblows cannot be avoided on the original song.
- `--avoid-overdraws`: discard harps that require overdraws; using this option will force `harpy` to look for satisfying transpositions if overdraws cannot be avoided on the original song.

4.2 Harp selection options

By default, `harpy` takes every kind of harp it knows about into account, but not many people own that many harps – especially not beginners. The following options allow you to get rid of some of them:

- `--no-country-harp`: discard country blues harps.
- `--no-low-harp`: discard low key harps.
- `--no-major-harp`: discard major blues harps.

- `--no-harmonic-minor-harp`: discard harmonic minor harps.
- `--no-natural-minor-harp`: discard natural minor harps.
- `--no-melody-maker-harp`: discard Melody Maker harps.

And the corresponding options for restricting your search to one kind of harp are also available:

- `--only-country-harps`: list only country blues harps.
- `--only-low-harps`: list only low key harps.
- `--only-harmonic-minor-harps`: list only harmonic minor harps.
- `--only-major-harps`: list only major blues harps.
- `--only-melody-maker-harps`: list only Melody Maker blues harps.
- `--only-natural-minor-harps`: list only natural minor harps.

Obviously, `--only` options are exclusive: you can select at most one of them.

4.3 Tab format options

The `-f FORMAT` (or `--format=FORMAT`) option allows you to select a format from several choices, by setting `FORMAT` to one of the following:

- `text`: this is the default, lightest and most basic format, and also the most straightforward to edit.
- `latex`: use LaTeX to produce tabs and compile them. Tabs look much better that way than in text format. On the other hand, you need to install a LaTeX distribution, and if you don't like the result, editing the tab will be harder if you're not familiar with this format.

Tabs are written to `.\tabs\FORMAT\`. When using `-f latex`, LaTeX source files are compiled and stored as PDF files in `.\tabs\pdf\`. Have a look at Section 5 to have a better idea of how the format you select will affect the resulting output.

4.4 Miscellaneous options

The following options do not fit in the other categories:

- `--version`: shows program's version number and exits.
- `-h, --help`: show a help message summarising all options and exits.

5 Tabs

Notation varies from author to author as far as tabs are concerned, and it will also vary here according to the format you ask `harpy` for.

5.1 LaTeX tabs

LaTeX tabs follow the notation shown in Table 1, a mix of what I found in several books, including [1] and [2].

Action	**Symbol**
blow hole x	$x\uparrow$
draw hole x	$x\downarrow$
bend hole x by one semitone	$\bar{x}$
bend hole x by two semitones	$\bar{\bar{x}}$
bend hole x by three semitones	$\bar{\bar{\bar{x}}}$
chord on x, y	$\{x, y\}$
overdraw on x	$\overset{\circ}{x}\downarrow$
overblow on x	$\underset{\circ}{x}\uparrow$

Table 1: Harmonica tab notation in LaTeX format

5.2 Text tabs

Text tabs follow the notation shown in Table 2, introduced by Will [3].

6 MidiSplitter

As of now, `harpy` is not able to guess which track(s) you are interested in. I strongly doubt this will change, because most MIDI files I came accross do not contain instrument names and even if they did, there could be several

Action	**Symbol**
blow hole x	`x>`
draw hole x	`x`
bend hole x by one semitone	`x'`
bend hole x by two semitones	`x''`
bend hole x by three semitones	`x'''`
chord on x, y	`x&y`
overdraw on x	`x#`
overblow on x	`x>#`

Table 2: Harmonica tab notation in text format

tracks of interest. Therefore, I wrote MidiSplitter in order to split a given MIDI file into k separate files, where k takes its values in the set of channels that are in use in the original midi file. You can then try and run `harpy` on those files. Its usage is very simple:

```
MidiSplitter.exe [options] midifile
```

The resulting files are stored in `.\basename\basename-i-instr.ext`, where `basename` is the name of the input midi file stripped of its path and extension `ext`, `i` is the channel, and `instr` is the (guessed) name of the instrument used on that channel.

Only one option (besides `--version` and `-h` or `--help`, which behave exactly the same as for `harpy`) is available: the `-d OUT_DIR` or `--directory= OUT_DIR` option, which changes the default output directory from `.` to `OUT_DIR`.

7 Acknowledgements

I want to thank Max M. for his Python Midi package, on which `harpy` heavily relies. You can get the original package there:

`http://www.mxm.dk/products/public/pythonmidi/`

References

[1] D. Baker, *Beginning Blues Harp*, Music Sales America, 1996.

[2] J.-J. Milteau, G. Szlapczyncki, and T. Crommen, *Méthode complète harmonica diatonique & chromatique*, Paul Beuscher, 1997.

[3] M. B. Will, *The diatonic harmonica reference.* `http://www.angelfire.com/tx/myquill/`.