Badʾu-l-masīri fī taʾrīḫi-t-tafsīri–An Exploratory Data Analysis of 17 Tafāsīr From the Early 8th to the Late 14th Century and Across a Wide Geographical Area Using Corpus Linguistical Methods

Universities of Leipzig and Maryland

Franziska Blunck, Sarosh Bukhari, Dr. Maryam Foradi, Thomas Davis Keppen, Jonathan Raphael Schmid, BA; supervising: Dr. Matthew Thomas Miller and Dr. Maxim Romanov

20 December 2016

Abstract

Through web scraping, a large corpus of *tafsīr* (Quran exegetical work) was downloaded and initial exploratory data analysis was performed on it by identifying the number of words written by the different authors on each chunk or passage, as well as on each *sūra* (chapter of the Quran). Through network analysis, it was possible to draw first conclusions about the acquired data.

---

1 Introduction

How has Islamic interpretation of the Quran changed over time? Have certain sections of the Quran enjoyed more attention from religious scholars and if so, has preference shifted with time? Does geographic distance equate to divergent preference for scholars who are contemporaneous to one another? Is doctrinal affiliation the primary factor in a *mufassir* (Islamic exegete) giving a section of the Quran more consideration? To answer these questions, we sought out exegetical works that were in a manageable

and navigable digital format.

1.1 A note on transliteration

The transliteration system used is that of the German Oriental Society[1]. We have tried to be consistent in our transliteration of Arabic words (and sometimes Persian names). For the inconsistencies that remain[2] we ask the reader's digression. As regards diphtongs we have opted towards the variant employing the combination of consonant plus vowel which we feel best evokes a "closer-to-correct" pronounciation in readers not trained in Arabic. Proper names were transliterated starting with capital letters only at the beginning of a sentence or if they are a person's or a place's name. Words that have found their way into English[3], and which the reader can be assumed to be familiar with, were only transliterated if belonging to a group of other Arabic words.

1.2 A technical note regarding this document

The present document was prepared using the `rmarkdown`[4] package and consequently `knitr`'s[5] built-in devices are being used to include the source code relevant to this study in the appendix. Unfortunately, the underlying libraries have not yet evolved to a point where they know how to properly break-around longer lines as they often appear in source code files. The reader is kindly asked to open the source code files directly from the GitHub repository where they are being kept[6] if she wishes to read them in detail. Towards that end, the appendix subheadings provide direct links to

[1] *Deutsche Morgenländische Gesellschaft* in the original. A copy of the latest transliteration rules anno 1969 is available from http://www.dmg-web.de/pdf/Denkschrift.pdf.

[2] Mostly, vowelisation was sometimes added where it helps the flow of pronounciation.

[3] Such as Quran, Sunni, Shia, Fatimid, etc.

[4] http://rmarkdown.rstudio.com/

[5] http://yihui.name/knitr/

[6] https://github.com/Islamicate-DH/hw/tree/master/tafaseer_group

the respective files displayed beneath them.

2 The corpus

As part of an effort to make Islamic scholarship more accessible, the Jordanian-based *Royal Aal al-Bayt Institute for Islamic Thought* commissioned the creation of altafsir.com, an online repository of classical works of *tafsīr*. The format of the website allows a user to select a specific *āya* (verse from the Quran) and see the relevant section of *tafsīr* that provides commentary on the *āya*. This structure, along with the fact that the texts are typed in and not photos of scanned pages, made the website ideal for our research which required that the texts be in a digital format and segmented according to the verse that is being commented on. The ultimate goal of altafsir.com is to provide a diverse collection of texts originating from various traditions, sects, and schools of jurisprudence. That desired breadth has yet to be achieved. The website currently has seventeen complete tafāsīr that were all written by *sunnī* (one of the major two sects of Islam) scholars.

2.1 The tafāsīr

The following *tafāsīr* which ended up being the subject of our work are listed below. Short remarks on some of their authors' biographies will follow in section 3.2.

- *al-baḥr al-muḥīt* by *Abū Ḥayyān*
- *al-ǧāmiᶜ li-ʾaḥkām al-qurʾān* by *al-Qurṭubī*
- *al-kaššāf* by *az-Zamaḫšarī*
- *al-muḥarrar al-waǧīz fī tafsīr al-kitāb al-ʾazīz* by *Ibn ᶜAṭīya*
- *an-nukat wa-l-ᶜuyūn* by *al-Māwardī*
- *anwār at-tanzīl wa-ʾasrār at-taʾwīl* by *al-Bayḍāwī*

- *baḥr al-ᶜulūm* by *as-Samarqandī*
- *ǧāmiᶜ al-bayān fī tafsīr al-qurʾān* by *aṭ-Ṭabarī*
- *lubāb at-taʾwīl fī maᶜānī at-tanzīl* by *al-Ḫāzin*
- *maᶜlam at-tanzīl* by *al-Baġawī*
- *madārik at-tanzīl wa-ḥaqaʾiq at-taʾwīl* by *an-Nasafī*
- *mafātīḥ al-ġayb* by *ar-Rāzī*
- *tafsīr al-ǧalālayn* by *al-Maḥallī* and *as-Suyūṭī*
- *tafsīr al-qurʾān* by *al-Fayrūzābādī*
- *tafsīr al-qurʾān* by *Ibn ᶜAbd as-Salām*
- *tafsīr al-qurʾān al-karīm* by *Ibn Kaṯīr*
- *zād al-masīr fī ᶜilm at-tafsīr* by *Ibnu-l-Ǧawzī*

3 Methodology

Our attempt at determining which verses enjoyed more consideration from the *mufassirūn* began with calculating the word count of every section of the *tafsīr* that was commenting on a specific verse and dividing that number by the total word count of the *tafsīr*. It was our belief that increased attention on a verse was indicative of some sort of importance to the *mufassir*. Importance might not necessarily mean that the *mufassir* had an increased affinity for the values or instructions espoused in the verse. A *mufassir* might have believed that an obscure verse required elucidation to make its message comprehensible to the layman Muslim and therefore devoted more writing to explication and contextualizing.

When we sorted the verses from most written about to least, we observed some suspicious patterns. Consecutive verses were appearing next to one another on the list and they had identical word counts. When we inspected the files in our corpus,

we found that there were duplicate chunks of text. The problem was that most of the writers alternated between writing about singles verses and writing about groups of consecutive verses that dealt with a single narrative or theme. If a section of a *tafsīr* covered consecutive verses together, altafsir.com repeated that segment of text when any one of those verses were selected on the website. When the script to download the *tafāsīr* was run, it downloaded the repeated segments of texts and organized them as if they were unique to a single verse. Only three of the writers in the corpus consistently wrote about single verses and therefore did not succumb to the duplicate text problem. Of the 87 *tafsīr* works listed on the website, 69 turned out to be completely empty and one author, *Ibn ᶜArafa*, only provided commentary on *sūratān*. Both were removed from the corpus used for analysis, which consequently consists of 17 works comprising $i_{total} = 16444304$ words.

After trying a number of different data formats for working with the corpus, we settled on importing it into a relational database table which enabled us to perform queries either detailled or broad on it without necessarily having to write any logic. To give an example, the following SQL query would return *az-Zamaḫšarī*'s commentary on the fourth verse of the famous *sūrat al-iḫlāṣ* together with a simple character count:

```
SELECT text, COUNT(text) AS char_count
  FROM cts_units
  WHERE cts_urn LIKE '%alzmkhshry%' AND sura_id=112 AND aaya_id=4;
```

We built a chain of queries first excluding duplicate texts and subsequently any repeat segments. Appendix VI shows the full chain of queries which led to the CSV file that was then imported into a spreadsheet program to continue manually preparing it for network analysis. Because of the way most of the authors structure their commentary on the Quran, we were not able to get word counts for single verses/*āyāt*, and so decided to focus on the chapters/*suwar*. There is a huge range of *sūra* lengths,

with *sūrat al-baqara*, the longest, consisting of 286 verses and *sūrat al-kawṯar*, the shortest, consisting of 3 verses, so looking at how long the commentary was for a specific *sūra* relative to total *tafsīr* length would probably just leave us with something that resembles a list of *suwar*, from longest to shortest. We chose instead to look at length of *tafsīr* commentary divided by the length of the *sūra* it is commenting. We looked at the top twenty-five *suwar* for each *mufassir* and observed that the number of shared *suwar* among the writers of the corpus ranged from twelve to twenty-two.

The relationship between the *mufassirūn* was visualized using Gephi[7], a free/open-source program used for social network analysis. Each writer is represented as a "node" on the graph. An "edge" is a connection between nodes which for our research is a *sūra* from the list of twenty-five that is shared by writers. A connection is weighted for each additional, shared *sūra* so that the connection between writers who share numerous *suwar* is stronger than the connection between those who share few. Gephi has a modularity function that defines sub-networks which we believed could show which of the writers of the corpus were similar in the manner that they gave certain *suwar* more attention. What resulted from the calculation was the clustering of the *mufassirūn* into four groups.

3.1 The results

Gephi's algorithm calculated four distinct groups, represented as colors in figure 1. They are as follows:

1. Green: *Ibnu-l-Ǧawzī* (d. 597 AH), *aṭ-Ṭabarī* (d. 310 AH), *Ibn Kaṯīr* (d. 774 AH), *Ibn ᶜAṭīya* (d. 546 AH), and *al-Baġawī* (d. 516 AH)
2. Blue: *az-Zamaḫšarī* (d. 538 AH), *al-Bayḍāwī* (d. 685 AH), and *an-Nasafī* (d. 710 AH)

[7] https://gephi.org/

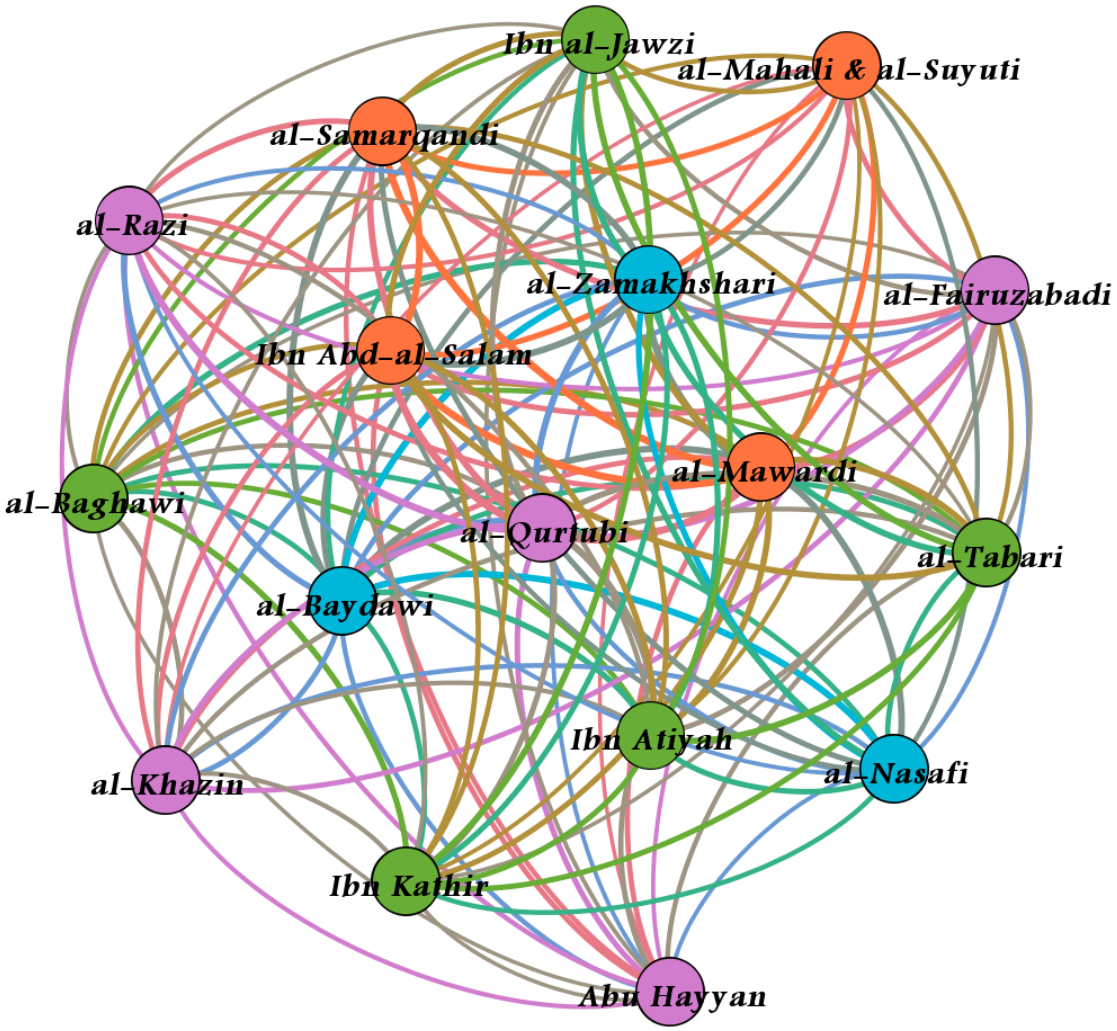

Figure 1: Gephi social network analysis result

3. Purple: *ar-Rāzī* (d. 606 AH), *al-Fayrūzābādī* (d. 817 AH), *al-Qurṭubī* (d. 671 AH), *al-Ḫāzin* (d. 741 AH), and *Abū Ḥayyān* (d. 754 AH)
4. Orange: *al-Maḥallī* (d. 864 AH) and *as-Suyūṭī* (d. 911 AH), *as-Samarqandī* (d. 375 AH), *al-Māwardī* (d. 450 AH), and *Ibn ᶜAbd as-Salām* (d. 660 AH)

The unifying characteristic for the exegetes of the first cluster is their reputation as traditionalists. *Aṭ-Ṭabarī* saw his method of *tafsīr bi-l-maṯūr*, or exclusive reliance on *ḥadīṯ* (accounts on the life of the prophet Muḥammad) in matters of Quranic interpretation, as essential to avoid the introduction of *bidᶜa* or heretical innovation (Bauer 2013). *Al-Baġawī* emphasized the importance of purity in Islamic doctrine and worked to exclude teachings that he believed could not be properly attributed to Muḥammad or the *ṣaḥāba*, the companions of Muḥammad (Robson 2012). *Ḥanbalī* adherent *Ibnu-l-Ǧawzī* campaigned against the Shia in Iraq and authored a paean[8] celebrating the end of *ismaᶜīlī* Fatimid rule in Egypt and the accession of a Sunni, *Ṣalāḥ ad-Dīn*, to power (Mallett ca. 2009). *Ibn ᶜAṭīya*'s exegesis received praise from *Ibn Taimīya*, the intellectual forefather of movements that are frequently labeled as Salafist, for its adherence to orthodoxy (Salahi 2002). The inclusion of *Ibn Taimīya*'s student *Ibn Kaṯīr* can be attributed to either his conservative leanings—he penned a defense of armed jihad against neighboring non-Muslim states (Laoust 2012)—or to his *tafsīr* being, as some scholars have observed, a summarized version of *aṭ-Ṭabarī* (McAuliffe 1991).

For the second cluster, geography seems to be the unifying factor. Arabic grammarian and *muᶜtazila* partisan *az-Zamaḫšarī* (McAuliffe 1991) differed from *ḥanafī* jurist *an-Nasafī* on matters of doctrine (Poonawala et al. 2012) but they both lived in the Central Asian region known as *mā warāʾu-n-nahr* (Transoxiana) while

[8] "[A] song, film, or piece of writing that praises someone or something very enthusiastically:" (Cambridge Dictionary of English)

the šāfiᶜī *al-Bayḍāwī* lived in what is now Iran (Krauss-Sánchez 2016). *Al-Bayḍāwī*'s exegesis *anwār at-tanzīl wa-ʾasrār at-taʾwīl* has been labeled an amended version of *az-Zamaḫšarī*'s *al-kaššāf*, with *muᶜtazila* philosophy and other perceived heresies expunged from the text (McAuliffe 1991). This grouping may reflect a unique Central Asian outlook to the Quran and the addition of *al-Bayḍāwī* may result from his reliance on the work of *az-Zamaḫšarī*.

If there is a latent, unifying factor to the *mufassirūn* of the third cluster, it is not immediately apparent to us. The exegetes were not contemporaneous to one another with *ar-Rāzī* preceding *al-Fayrūzābādī* by two centuries. Geography certainly did not connect the writers. *Abū Ḥayyān* (Glazer and Homerin 2008) and *al-Qurṭubī* were both born in Muslim Spain and ultimately settled in Egypt (Ruano, n.d.). *Al-Ḫāzin* lived in Syria which also was briefly the home of Persian-born *al-Fayrūzābādī* who also spent his life in Jerusalem, Mecca, and Yemen (Fleisch 2012). *ar-Rāzī* lived in what is now Iran and Afghanistan (McAuliffe 1991). In terms of doctrinal leanings, *Abū Ḥayyān* was a traditionalist and admirer of *Ibn Taimīya* (Glazer and Homerin 2008) while *ar-Rāzī*, though ostensibly a critic of *muᶜtazila* theology, is seen to have incorporated heterodox tenets emphasizing reason and *taʾwīl* or figurative reading in Quranic interpretation (Jaffer, 2015). Unfortunately, the inability to obtain reliable word counts at the verse-level (and therefore our inspection at the chapter-level) may have adversely affected the quality of the groupings calculated by Gephi.

The last cluster of our analysis resulted in a predictable grouping of exegetes. The primary characteristic that unified these authors was their choice of school of jurisprudence. These exegetes were all from different time periods and, for the most part, varying geographical locations yet still their analogous school of thought when approaching the interpretations of the Quran was similar enough to differentiate this group from the rest of our corpus. Amongst the four authors in this cluster three were

from the school of *aš-Šāfiᶜī* and one from the school of *al-Ḥanafī*.

Along with school of jurisprudence, a common factor amongst most of the authors was their interest in or relation to Islamic Law.

3.2 Remarks on noteworthy authors of works in the corpus after network analysis

Al-Māwardī was an Islamic jurist born in Basra, Iraq during 364 AH (Sharif 2002). During his time he was considered a high profile figure appointed with various significant responsibilities, including serving as a diplomat for the caliphate and eventually appointed chief Imam of Baghdad (Zahoor 1998) . The city where *al-Māwardī* lived was known to be a hub of the *muᶜtazila* school of thought, so some found it peculiar that its chief Imam was in fact associated with the šāfiᶜī school of thought. He was later condemned for his *muᶜtazila* sympathies. *Al-Māwardī* was known for his works in Islamic legal principles, in fact in 429 AH the caliph *al-Qādir* summoned four jurists from each school of jurisprudence to write a legal epitome (Sharif 2002). *Al-Māwardī* was chosen to represent the school of *aš-Šāfiᶜī*. Amongst the four jurists the Caliph favored *al-Māwardī* and appointed him chief Imam (Sharif 2002).

ᶜIzz ad-Dīn ibn ᶜAbd as-Salām was born in Damascus in the year 577 AH (Salahi 2016), long after *as-Samarqandī* and *al-Māwardī*. During his time he was the leading authority in the *šāfiᶜī* jurisprudence and most famous for his interpretations of Islamic legal principles. Unlike *al-Māwardī*, *ᶜAbd as-Salām* was known to be defiant of customs he deemed "unsanctioned". He even went as far as condemning those who follow these unsanctioned customs regardless of status; most notably the ruler of Damascus, *aṣ-Ṣāliḥ ʾIsmaᶜīl* (Jackson 1996). This condemning led to his imprisonment and eventual emigration to Cairo, Egypt. There, he was credited as the first jurist to teach

Quranic commentary in Egypt (Salahi 2016). Like *al-Māwardī*, *as-Salām* was a world renowned scholar of *fatāwa* (sg. *fatwa*, a document of legal opinion in Islamic Law). Indeed, *as-Salām* was so respected, Islamic jurists like *al-Ḥāfiẓ al-Munḏirī* stopped giving *fatāwa* stating that "[i]t does not behove any jurist to give a *fatwa* where *ᶜIzz ad-Dīn* [*Ibn ᶜAbd as-Salām*] happens to be present" (Salahi 2016). Both *Ibn ᶜAbd as-Salām* and *al-Māwardī* were highly respected scholars of their time especially in regards to Islamic Law. Both authors were advisors to respected Islamic governments of their time (Jackson 1996), another factor which may have attributed to the grouping of these authors in the same cluster.

Where *ᶜAbd as-Salām* died, much later *Ǧalāl ad-Dīn as-Suyūṭī* was born in the year 849 AH (Encyclopædia Britannica 2007), making his work the most recent here mentioned. Like most authors in the cluster he was an Islamic jurist from the school of *aš-Šāfiᶜī*. He was acknowledged as one of the more recent authorities of the *šāfiᶜī* school whose degree of *iǧtihād* was "accepted by most" (Meri 2005). *As-Suyūṭī* lacked a particular interest in Islamic Law and was not known to be a head advisor for any Islamic government like both *Ibn ᶜAbd as-Salām* and *al-Māwardī* had been. Although his father having been a judge of an Islamic state may have influenced any emphasis on the (Islamic) law within his writings and thus placing him within this cluster (Encyclopædia Britannica 2007). Unlike the other authors, *as-Suyūṭī* also studied the *ḥanafī* school of jurisprudence and was tutored by a Sufi, though his work drew primarily from the *šāfiᶜī* school of thought. *As-Suyūṭī*'s interest in the *ḥanafī* jurisprudence may account for the inclusion of *as-Samarqandī* in the cluster.

Abu-l-Layṯ as-Samarqandī, author of *baḥr al-ᶜulūm* was the single author in the cluster from the school of *al-Ḥanafī*. Apart from his jurisprudence another outlying factor for *as-Samarqandī* was his geographical location and time period. He was born in Afghanistan during 373 AH making him the oldest exegete in this cluster (Brannon

M. Wheeler 2002, p. 337). Since his work dated back so far we were unable to find much information as regards his lifestyle or any other factors that may have attributed to his placing in this cluster. We were however able to compile a few specific features of his exegesis: in his work on the Quran, *as-Samarqandī* was known to relate the stories of *ṣaḥāba* and of other *ḥadīṯ* from people not proven to be entirely reliable. He was also known for his lack of interest in the various *qirāʾāt*, the methods of recitation of the Quran (Wikipedia probably 2011). Unfortunately, due to lack of information on *as-Samarqandī* we were unable to pinpoint a specific factor that would explain his grouping in the fourth, or orange cluster. For the time being he remains an outlier.

4 Conclusion

From the results of the study we cannot definitively state that focusing on certain aspects of the Quran is determined principally by any of the factors mentioned at the beginning of this paper (chronology, geography or dogma). It is our contention that examining commentaries that are able to be divided up at the verse-level should yield clearer, more consistent clusters. There was a surprisingly large degree of overlap among the lists of top twenty-five *suwar*. Shorter *suwar* may require more commentary simply because their laconic nature demands contextualizing and therefore their placement at the top of the lists reveals little about the attention *mufassirūn* gave to certain sections of the Quran.

What this research did was attempt to explore a new method of charting the relationships of Quranic scholars that rely on an impartial, distant examination of *tafāsīr* that traditional categorization–according to sect or school of jurisprudence–may lack. The results were not as conclusive as we had hoped, however, we believe that examination of *sūra* and *āya* preferences of Quranic exegetes deserves further

exploration.

5 References

Bauer, K. 2013. 'A Study of Introductions to Classical Works of Tafsir', in *Aims, Methods and Contexts of Qur'anic Exegesis* (2nd/8th-9th/15th.) (New York, NY: Oxford University Press, 2013), 39–65.

Brannon M. Wheeler 2002. *Prophets in the Quran. An introduction to the Quran and Muslim exegesis.* (London: Continuum, 2002).

Encyclopædia Britannica 2007. 'Al-Suyuti', *Encyclopedia Britannica Online* (2007).

Fleisch, H. 2012. 'Al-Fīrūzābādī', ed. P. Bearman, T. Bianquis, C.E. Bosworth, E. van Donzel, and W.P. Heinrichs *Encyclopaedia of Islam* (2012).

Glazer, S. and T.E. Homerin 2008. 'Abū Ḥayyān al-Gharnāṭī', ed. K. Fleet, G. Krämer, D. Matringe, J. Nawas, and E. Rowson *Encyclopaedia of Islam* (2008).

Jackson, S. 1996. *Islamic law and the state. The constitutional jurisprudence of Shihāb al-Dīn al-Qarāfī.* (Leiden: E. J. Brill, 1996).

Krauss-Sánchez, H.R. 2016. 'Al-Baydāwī', ed. G. Dunphy and C. Bratu *Encyclopedia of the Medieval Chronicle* (2016).

Laoust, H. 2012. 'Ibn Kaṯ̲hīr', ed. P. Bearman, T. Bianquis, C.E. Bosworth, E. van Donzel, and W.P. Heinrichs *Encyclopaedia of Islam* (2012).

Mallett, A. ca. 2009. 'Ibn al-Jawzī', in *Christian-Muslim Relations 600 - 1500* (ca. 2009).

McAuliffe, J.D. 1991. *Qur'anic Christians: An Analysis of Classical and Modern Exegesis* (New York, NY: Cambridge University Press, 1991).

Meri, J.W. 2005. 'Suyuti, Al-, Abd al-Rahman', *Medieval Islamic Civilization. An Encyclopedia.* (Routledge, 2005), 784–786.

Poonawala, I., A.J. Wensinck, and W. Heffening 2012. 'Al-Nasafī', ed. I. Bearman, T. Bianquis, C.E. Bosworth, E. van Donzel, and W.P. Heinrichs *Encyclopaedia of Islam* (2012).

Robson, J. 2012. 'Al-Baġawī', ed. P. Bearman, T. Bianquis, C.E. Bosworth, E. van Donzel, and W.P. Heinrichs *Encyclopaedia of Islam* (2012).

Salahi, A. 2002. 'Scholar of Renown: Ibn Atiyyah', *Arab News*, 14 October 2002.

Salahi, A. 2016. 'Izz Al-Din ibn Abd Al-Salam', *Muslim Heritage*, 1 December 2016.

Sharif, M.M. 2002. 'Al-Māwardī', ed. P. Bearman, T. Bianquis, C.E. Bosworth, E. van Donzel, and W.P. Heinrichs *Encyclopaedia of Islam* (2002).

Wikipedia probably 2011. 'Tafsîr as samarqandî', *Wikipedia* (probably 2011).

Zahoor, A. 1998. 'Al-Mawardi, 972-1058 C.E.', (1998).

Appendix

I Script 01_retrieval/download.r

```
#!/usr/bin/env Rscript
# Only runs on R >= 3.3.1!
# Requires: install.packages(c('urltools', 'rvest', 'XML', 'magrittr', 'stringr', '
# Remember: every line of code one liability!

for (lib in c('urltools', 'curl', 'rvest', 'stringr', 'optparse', 'tools', 'methods')
  suppressPackageStartupMessages(library(lib, character.only = TRUE))
}

default_url = 'http://www.altafsir.com/Tafasir.asp?tMadhNo=0&tTafsirNo=0&tSoraNo=1&tA
parameters  = param_get(default_url, c('tMadhNo', 'tTafsirNo', 'tSoraNo', 'tAyahNo'))
save_path   = file.path('..', '..', 'corpora', 'altafsir_com', 'downloaded')

# Dropdown boxes 1 and 3
number_of_madahib = 10
number_of_suwar  = 114
# Dropdown boxes 2 and 4,
# thank you Christoph!
number_of_tafaseer_per_madhab = c(8, 20, 10, 2, 7, 7, 4, 3, 5, 2)
number_of_aayaat_per_sura     = c(7, 286, 200, 176, 120, 165, 206, 75, 129, 109, 123,

download_all <- function(url, path, start_pos=c(1,1,1,1), stop_pos=c(0,0,0,0))
{
  pos_set = FALSE
  t0 = proc.time()
  # The mother of all nested loops. Makes Ross' and Robert's hearts cringe,
  # but works. It would be more R-ish to use expand.grid on the `parameters'
  # data grid, unfortunately urltools does not seem to provide the necessary
  # functions to do so.
  # ~~~
  # Major thanks to Franziska for all the sweets that made this possible!
  # Still haven't finished all the Knoppers!
  for (madhab in 1:number_of_madahib) {
    if (!pos_set && madhab < start_pos[1]) {next}
    for (tafsir in 1:number_of_tafaseer_per_madhab[madhab]) {
      if (!pos_set && tafsir < start_pos[2]) {next}
      for (sura in 1:number_of_suwar) {
        if (!pos_set && sura < start_pos[3]) {next}
```

```
        for (aaya in 1:number_of_aayaat_per_sura[sura]) {
          if (!pos_set) {if (aaya < start_pos[4]) {next} else {pos_set = TRUE}}
          delim = rep('-', 115)
          message(
            c("\033[2J","\033[0;0H"), delim, '\n Working…\n', delim,
            sprintf('\n Madhab:\t%s/%s | ', madhab, number_of_madahib),
            sprintf('Tafsir:\t%s/%s | ',     tafsir, number_of_tafaseer_per_madhab[ma
            sprintf('Sura:\t%s/%s | ',       sura,   number_of_suwar),
            sprintf('Aaya:\t%s/%s | ',       aaya,   number_of_aayaat_per_sura[sura])
            sprintf('Time elapsed:\t%.0f min\n', (proc.time() - t0)[3] / 60),
            delim, '\n', url)
          download(url, path, sura, aaya, madhab, tafsir)
          # If this happens, we've hit the point where we're supposed to stop.
          if (identical(c(madhab,tafsir,sura,aaya), stop_pos)) {stop_execution()}
          sleep(0.1) # Sleep a little so we're not seen as a threat.
        }
      }
    }
  }
}

# Taken from http://stackoverflow.com/questions/17837289/break-exit-script
# I completely agree with the name of this function in any circumstance!
stop_execution <- function()
{
  cat("Reached end of requested range, stopping.")
    pid = Sys.getpid()
    pskill(pid, SIGINT)
    Sys.sleep(1)
}

# Taken from https://stat.ethz.ch/R-manual/R-devel/library/base/html/Sys.sleep.html
# Pretty much what has been lacking these last days...
sleep <- function(s)
{
  t0 = proc.time()
  Sys.sleep(s)
  proc.time() - t0
}

download <- function(url, root, sura, aaya, madhab, tafsir)
```

```
{
  url = param_set(url, 'tMadhNo',   madhab)
  url = param_set(url, 'tSoraNo',   sura)
  url = param_set(url, 'tTafsirNo', tafsir)
  url = param_set(url, 'tAyahNo',   aaya)
  # This is where we start
  page = 1
  no_pages = 1 # Might not be true, but we're assuming it for now
  # Will succeed at least once
  while (page <= no_pages) {
    path = file.path(root,
                     sprintf('quran_%03d', sura),
                     sprintf('aaya_%03d', aaya),
                     sprintf('madhab_%02d', madhab),
                     sprintf('tafsir_%02d', tafsir)) # Logical order
    file = file.path(path, sprintf('page_%02d.html', page))
    # The actual downloading and writing of the file,
    # unless it exists. Note that in that case it'll
    # be read from disk to figure out if there are
    # additional pages.
    if (!file.exists(file)) {
      url = param_set(url, 'Page', page)
      response = curl_fetch_memory(url)
      contents = iconv(rawToChar(response$content), from='CP1256', to='UTF-8')
      dir.create(path, showWarnings = FALSE, recursive = TRUE)
      write(contents, file)
      message(sprintf('\tSaved page %s to %s', page, file))
    }
    # Now let's find out the actual truth!
    if (page == 1 && file.info(file)$size > 0) {
      no_pages = extract_number_of_pages(read_html(file))}
    # Whatever it is, the show must go on...
    page = page + 1
  }
  # Position marker so we can pick up where we left off
  file = file.path(root, 'scraper_pos.dat')
  pos  = sprintf('%s,%s,%s,%s', madhab, tafsir, sura, aaya) # Website's order
  write(pos, file)
}

extract_number_of_pages <- function(raw_html)
```

```
{
  n <- raw_html %>%
    html_nodes('#DispFrame center u') %>%
    html_text() %>%
    as.numeric()
  # This is tricky in R: a NULL or NA value could result!
  if (length(n) < 1) {
    n = 1
  } else {
    n = n[length(n)]
  }
  message('\tNumber of pages: ', n)
  return(n)
}

# Figure stuff out...
option_list = list(
  make_option(
    c('-b', '--start'),
    action='store', default=NA, type='character',
    help='Where to start downloading: madhab,tafsir,sura,aaya'),
  make_option(
    c('-e', '--stop'),
    action='store', default=NA, type='character',
    help='Where to stop downloading: madhab,tafsir,sura,aaya'
  )
); o = parse_args(OptionParser(option_list=option_list))

# Let's rock'n'roll!
if (!is.na(o$start) && !is.na(o$stop)) {
  # Assume we're one of many processes and only download what we're told
  start_pos = strtoi(unlist(strsplit(o$start, split=',')))
  stop_pos  = strtoi(unlist(strsplit(o$stop, split=',')))
  download_all(default_url, save_path, start_pos, stop_pos)
} else {
  # Assume we're just one process and it's our job to do it all
  file = file.path(save_path, 'scraper_pos.dat')
  if (file.exists(file)) {
    pos = unlist(read.csv(file, header=FALSE))
    download_all(default_url, save_path, pos)
  } else {
```

```
    download_all(default_url, save_path)
  }
}
```

II Script 01_retrieval/extract.r

```
#!/usr/bin/env Rscript
# Only runs on R >= 3.3.1
# Requires: install.packages(c('rvest', 'XML', 'magrittr', 'stringr', 'jsonlite', '
# Remember: every line of code one liability!

for (lib in c('rvest', 'stringr', 'jsonlite', 'yaml', 'httr', 'optparse', 'stringi'))
  suppressPackageStartupMessages(library(lib, character.only = TRUE))
}

# Home-made libraries for the win!
source(file.path('..', 'lib', 'grepx.r'))

paths = list()
paths$downloaded <- file.path('..', '..', 'corpora', 'altafsir_com', 'downloaded')
paths$extracted  <- file.path('..', '..', 'corpora', 'altafsir_com', 'extracted')
paths$quran      <- file.path('..', '..', 'corpora', 'the_quran_by_aaya')

read_dirs <- function(paths, force=FALSE)
{
  t0 = proc.time()
  # Go through sura directories, save sura number
  for (p1 in Sys.glob(file.path(paths$downloaded, 'quran_???'))) {
    # Go through aaya directories, save aaya number
    for (p2 in Sys.glob(file.path(p1, 'aaya_???'))) {
      # Go through madhab directories, save madhab number
      for (p3 in Sys.glob(file.path(p2, 'madhab_??'))) {
        # Go through tafsir directories, save tafsir number
        for (path in Sys.glob(file.path(p3, 'tafsir_??'))) {
          message(path)
          paths$infile <- path
          data = list() # Whither to put our treasure, arrr!
          regx = 'quran_(?<sura>\\d{3})/aaya_(?<aaya>\\d{3})/madhab_(?<madhab>\\d{2})
          data$position = grepx(regx, path)[[1]] # See ../lib/grepx.r! # Attn: chara
          display_status_message(t0, data$position)
          # Go through page files
```

```
          paths$outpath = file.path(paths$extracted,
            sprintf('quran_%s',  data$position$sura  ),
            sprintf('aaya_%s',   data$position$aaya  ),
            sprintf('madhab_%s', data$position$madhab))
          paths$outfile <- file.path(paths$outpath, sprintf('tafsir_%s.yml', data$pos
          if (file.exists(paths$outfile) && !force) {
            message(paste(paths$outfile, 'exists, skipping infile dir...'))
          } else {
            read_files(paths, data)
          }
        } # path
      } # p3
    } # p2
  } # p1
}

read_files <- function(paths, data)
{
  for (infile in list.files(paths$infile, full.names=TRUE)) {
    message(paste('Processing', infile))
    raw_html = read_html(infile, encoding='utf8')
    regx = 'page_(?<page>\\d{2})\\.html'
    page = as.numeric(grepx(regx, infile)[[1]]$page)
    # Open first page file
    if (page == 1) {
      # Figure out meta data
      #   Save tafsir name
      #   Save mufassir name
      #   Save mufassir death date
      data$meta = extract_meta(raw_html)
      # Figure out first block of aayaat, ignore it
      #   Download the ayah given by directory numbers via GQ API
      data$aaya = gq_get_aaya(
        paths$quran,
        as.numeric(data$position$sura),
        as.numeric(data$position$aaya)
      )
      # Go through subsequent result blocks
      #   Figure out what each block is
      #     With an appropriate tag, add it to the tafsir text
      data$text = c(extract_text(raw_html))
```

```
    } else {
      message('\t(subsequent page)')
      # Keep going through page files, if any
      #   Figure out first block of aayaat, ignore it
      #   Go through subsequent result blocks
      #     Figure out what each block is
      #       With an appropriate tag, add it to the tafsir text
      text = extract_text(raw_html)
      if (length(text) > 0) data$text = c(data$text, text)
    }
  }
  write_file(paths, data) # If there was no tafsir text, that field will be missing,
                          # indicating that that particular aaya was of no concern
                          # to the mufassir.
}

write_file <- function(paths, data)
{
  # Join all pages together into one string, preserving the information of where th
  data$text = paste(paste('<section>', data$text, sep='', collapse='</section>'), '</
  # Save the whole shebang into quran_n/aaya_n/madhab_n/tafsir_n.yml
  message(paste('Writing', paths$outfile))
  dir.create(paths$outpath, showWarnings=FALSE, recursive=TRUE)
  write(as.yaml(data), paths$outfile)
}

display_status_message <- function(t0, position) {
  delim = rep('-', 115)
  message(
    c("\033[2J","\033[0;0H"),
    delim, '\n Working…\n',
    delim,
    sprintf('\nSura:\t%s | ',  position$sura),
    sprintf('Ayah:\t%s | ',    position$aaya),
    sprintf('Madhab:\t%s | ', position$madhab),
    sprintf('Tafsir:\t%s | ',  position$tafsir),
    sprintf('Time elapsed:\t%.0f min\n', (proc.time() - t0)[3] / 60),
    delim
  )
}
```

```
extract_text <- function(raw_html)
{
  text <- raw_html %>%
    html_nodes('#SearchResults') %>%
    xml_contents()
  trimws(gsub('[\r\n]', '', toString(text)))
}

extract_meta <- function(raw_html)
{
  meta <- raw_html %>%
    html_nodes('.TextArabic > .TextResultArabic:nth-child(1)') %>%
    html_text() %>%
    head(1) # Only selected nth-child(1), so 1 result max.
  if (!is.null(meta)) {
    regx = "\\*\\s*(?<title>.*?) ?\\/ ?(?<author>\\s?.*?)\\s?\\(\\D*(?<year>\\d{3,4})
    meta = grepx(regx, meta)
    if (length(meta) > 0) return(meta[[1]]) # Only return something if we were able
  }
}

gq_get_aaya <- function(path, sura, aaya)
{
  file = file.path(path, sprintf('%s,%s', sura, aaya))
  if (file.exists(file)) {
    return(paste(scan(file, what='character', quiet=TRUE), collapse=' ')) # No simpl
  } else {
    url = sprintf('http://api.globalquran.com/ayah/%s:%s/quran-simple', sura, aaya)
    json = fromJSON(url)
    verse = json$quran$`quran-simple`$`1`$verse
    if (!is.null(verse)) {
      write(verse, file)
      return(verse)
    }
  }
}

option_list = list(
  make_option(
    c('-f', '--force'),
    action='store_true', default=FALSE,
```

```
    help='Overwrite already existing files [default %default]'
  )
); o = parse_args(OptionParser(option_list=option_list))

read_dirs(paths, o$force)
```

III Script 01_retrieval/process.rb

```
#!/usr/bin/env ruby

require 'yaml'
require 'nokogiri'
require 'fileutils'

path = File.join('..', '..', 'corpora', 'altafsir_com', 'extracted')
files = Dir.glob(File.join(path, '**', '*.yml'))
testfile = File.join(path, 'quran_001/aaya_001/madhab_01/tafsir_01.yml')

class TafsirFile
  def initialize(file)
    @in_file = file
  end

  def self.convert(file)
    instance = self.new(file)
    instance.read
    instance.clean_yaml
    instance.clean_html
    instance.write
  end

  def read
    begin
      @yaml = YAML.load(File.open(@in_file))
      @html = Nokogiri::HTML.fragment(@yaml['text']) do |config|
        config.strict.nonet.noent.noblanks
      end
      print "#{@in_file} => "
    rescue
      puts "Problem parsing '#{@in_file}', aborting."
      abort
```

```
  end
end

def write
  @out_file = @in_file.gsub(/extracted/, 'processed')
  @out_path = @out_file.gsub(/\/tafsir_\d{2}\.yml/, '')
  begin
    FileUtils.mkdir_p(@out_path)
    File.open(@out_file, 'w') {|f| f.write(@yaml.to_yaml)}
    puts @out_file
  rescue
    puts "Problem writing '#{@out_file}', aborting."
    abort
  end
end

def clean_yaml
  %w{sura aaya madhab tafsir}.each {|p| @yaml['position'][p] = @yaml['position'][p]
  @yaml['text'] = String.new
end

def clean_html
  @doc = Nokogiri::HTML.fragment('') do |config|
    config.strict.nonet.noent.noblanks
  end
  builder = Nokogiri::HTML::Builder.with(@doc) do |doc|
    @html.css('section').each do |section|
      doc.section {
        nodes = section.css('div[align="right"][dir="rtl"] font[color]')
        nodes.each do |node|
          unless node.inner_text.empty?
            case node['color']
              when 'Olive'
                css_class = 'poetry_or_grammar'
              when 'Red'
                css_class = 'hadith'
              when 'ForestGreen'
                css_class = 'quran'
            end
            if css_class.nil?
              doc.p {
                doc.text node.inner_text
```

```
              }
            else
              doc.p(:class => css_class) {
                doc.text node.inner_text
              }
            end
          end
        end
      }
    end
  end
  @yaml['text'] << @doc.to_html
 end
end

# TafsirFile.convert(testfile)
files.each {|f| TafsirFile.convert(f)}
```

IV Script 01_retrieval/convert.rb

```
#!/usr/bin/env ruby

require 'bundler/setup'
require 'yaml'
require 'csv'
require 'pandoc-ruby'
require 'fileutils'
require 'sanitize'
require 'pp'
require 'sqlite3'
require 'active_record'
require 'awesome_print'
require 'pry'
require 'nokogiri'
require 'digest/md5'
require_relative '../lib/asciiarabic'
require_relative '../lib/flat_hash'
require_relative '../lib/numeric_to_hindi'

ActiveRecord::Base.establish_connection(
  adapter: 'sqlite3',
```

```
  database: '../../corpora/altafsir_com/processed/corpus.sqlite3'
)

class CTSUnit < ActiveRecord::Base
  # CREATE TABLE units(
  #   id          INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
  #   cts_urn     CHAR(255) NOT NULL,
  #   text        TEXT,
  #   label       TEXT,
  #   title       CHAR(255),
  #   author_name CHAR(255),
  #   author_era  INTEGER
  #   category_id INTEGER NOT NULL,
  #   author_id   INTEGER NOT NULL,
  #   sura_id     INTEGER NOT NULL,
  #   aaya_id     INTEGER NOT NULL
  # );
  default_scope {order('id ASC')}
end

class AlTafsirYAMLFiles
  def initialize
    @inpath  = File.join('..', '..', 'corpora', 'altafsir_com', 'processed', 'yaml')
    @outpath = File.join('..', '..', 'corpora', 'altafsir_com', 'processed')
    @number_of_madahib = 10
    @number_of_suwar  = 114
    @number_of_tafaseer_per_madhab = [8, 20, 10, 2, 7, 7, 4, 3, 5, 2]
    @number_of_aayaat_per_sura = [
      7, 286, 200, 176, 120, 165, 206, 75, 129, 109, 123, 111, 43, 52, 99, 128,
      111, 110, 98, 135, 112, 78, 118, 64, 77, 227, 93, 88, 69, 60, 34, 30, 73,
      54, 45, 83, 182, 88, 75, 85, 54, 53, 89, 59, 37, 35, 38, 29, 18, 45, 60,
      49, 62, 55, 78, 96, 29, 22, 24, 13, 14, 11, 11, 18, 12, 12, 30, 52, 52,
      44, 28, 28, 20, 56, 40, 31, 50, 40, 46, 42, 29, 19, 36, 25, 22, 17, 19,
      26, 30, 20, 15, 21, 11, 8, 8, 19, 5, 8, 8, 11, 11, 8, 3, 9, 5, 4, 7, 3,
      6, 3, 5, 4, 5, 6
    ]
    @hash = {}
    @yaml = ''
    @html = ''
    @header = %{<!doctype html>
<html lang="ar" dir="rtl" style="display:flex;justify-content:center;">
```

```
<head>
<meta charset="utf-8">
  <style type="text/css">
    h1 {font-size:120%;}
    h2 {font-size:100%;}
    p.quran {color:#0E4E00;}
    p.hadith {color:#225F6B;}
    p.poetry_or_grammar {color:#671F10;}
  </style>
</head>
<body style="width:50%;margin=1em 0;font-family:'Traditional Arabic';font-size:16pt;l
    @footer = %{</body>\n</html>}
  end

  def self.convert_to(formats)
    instance = self.new
    instance.convert_to(formats)
  end

  def convert_to(formats)
    walk_tree__by_book(formats)
  end

  def remove_specialchars(text)
    text = text.gsub(/[^ \p{Arabic}]/, '') # This is a *very* crude method which doe
    return text.squeeze
  end

  def urn(line_no)
    # CITE CTS URN example:
    # urn:cts:arabLit:tafsir.author.work:1.2.1234
    author = ASCIIArabic.translit(@hash['meta_author'])
    book   = ASCIIArabic.translit(@hash['meta_title'])
    sura   = @hash['position_sura'].to_i
    aaya   = @hash['position_aaya'].to_i
    # No need to continue if we don't have the full URN
    if (author.empty? || book.empty?)
      return false
    else
      return "urn:cts:arabLit:tafsir.#{author}.#{book}:#{sura}.#{aaya}.#{line_no}"
    end
  end
```

```ruby
# For purposes of the DH Leipzig/Maryland/etc. research groups (be
# able to use To Pan, etc.) the CSV files must comply with CITE CTS.
# The specs are at
# http://cite-architecture.github.io/ctsurn_spec/specification.html.

def cts_csv_writeline(madhab, tafseer, line_no, opts = {nospecialchars: false})
  # @hash contents example:
  #
  # {"position_sura"=>1,
  #  "position_aaya"=>1,
  #  "position_madhab"=>1,
  #  "position_tafsir"=>1,
  #  "meta_title"=>"     ",
  #  "meta_author"=>"",
  #  "meta_year"=>"310",
  #  "aaya"=>"   ",
  #  "text"=>
  #    "<section><p>   {  }.
  return unless urn = encode_urn(line_no)
  outname = "%03d-%03d.csv" % [madhab, tafseer]
  optname = (opts.map {|k,v| k if v}).compact.join(',')
  outpath = File.join(@outpath, 'csv', ['cts+aaya', optname].reject {|i| i.empty?}.
  text = @hash['text']
  text = remove_specialchars(text) if opts[:nospecialchars]
  values = [urn, text, @hash['aaya']]
  outfile = File.join(outpath, outname)
  FileUtils.mkdir_p(outpath)
  write_header = !(File.file?(outfile))
  header = ['urn', 'text', 'aaya']
  CSV.open(outfile, 'ab') do |csv|
    if write_header
      csv << header
    else
      csv << values
    end
  end
end

# afterwards may need to do:
# delete from cts_units where (category_id || '-' || author_id) = '1-1' and coale
def cts_sqlite_writeline(madhab, tafseer, line_no)
```

```
  current_txt = Pathname("../../corpora/altafsir_com/processed/plain/complete/%03d-
  return unless File.exist? current_txt
  begin
    text = remove_specialchars(@hash['text'])
    if _urn = urn(line_no)
      CTSUnit.create(
        cts_urn:     _urn,
        text:        text,
        text_hash:   Digest::MD5.hexdigest(text),
        label:       @hash['aaya'],
        title:       @hash['meta_title'],
        author_name: @hash['meta_author'],
        author_era:  @hash['meta_year'],
        category_id: @hash['position_madhab'],
        author_id:   @hash['position_tafsir'],
        sura_id:     @hash['position_sura'].to_i,
        aaya_id:     @hash['position_aaya'].to_i
      )
    end
  rescue Exception => e
    puts "INSERT failed: #{e.inspect}"
    return
  end
end

def html5_addline
  @html += "<h1>#{@hash['meta_title']} #{@hash['meta_author']} (. #{@hash['meta_yea
  @html += "<h2 class='quran'>#{@hash['aaya']}</h2>\n#{@hash['text']}\n"
end

def html5_write_unless_exists(madhab, tafseer)
  outname = "%03d-%03d.html" % [madhab, tafseer]
  outpath = File.join(@outpath, 'html5')
  outfile = File.join(outpath, outname)
  unless File.file?(outfile)
    print 'html5 '
    FileUtils.mkdir_p(outpath)
    File.write(outfile, @header+@html+@footer)
  end
  outfile
end
```

```
def plain_text_write(html_file, madhab, tafseer, opts = {nohadith: false})
  outname = "%03d-%03d.txt" % [madhab, tafseer]
  lastdir = opts[:nohadith] ? 'nohadith' : 'complete'
  outpath = File.join(@outpath, 'plain', lastdir)
  plain_file = File.join(outpath, outname)
  unless File.exist?(plain_file)
    html = Nokogiri::HTML(File.read(html_file))
    if opts[:nohadith]
      print 'plain_nohadith '
      hadith_paragraphs = html.at_css('p.hadith')
      hadith_paragraphs.remove if hadith_paragraphs
    else
      print 'plain '
    end
    html = html.at_css('body').text.strip
    FileUtils.mkdir_p(outpath)
    File.open(plain_file, 'w') do |outfile|
      outfile.puts Sanitize.fragment(html, {
        whitespace_elements: {
          'h1':      { before: "\n",   after: "\n\n" },
          'h2':      { before: "\n",   after: "\n"   },
          'section': { before: "\n\n", after: "\n"   },
          'p':       { before: "\n",   after: "\n"   }
      }})
    end
  end
end

def other_formats_write(infile, madhab, tafseer, formats)
  outname = "%03d-%03d" % [madhab, tafseer]
  fs = formats; %w{csv plain html5}.each {|f| fs.delete(f)}
  formats.each do |format|
    print "#{format} "
    case format
      when 'markdown' then ext = 'md'
      when 'latex'    then ext = 'tex'
      else ext = format
    end
    outpath = File.join(@outpath, format)
    outfile = File.join(outpath, "#{outname}.#{ext}")
    FileUtils.mkdir_p(outpath)
```

```
    File.open(outfile, 'w') do |file|
      file.puts PandocRuby.html([infile]).convert({to: format}, wrap: 'none')
    end
  end
end

def set_format_flags(formats)
  @formats = {other: formats.any? {|x| (x != 'csv' && x != 'csv_nospecialchars' &&
  formats.each do |f|
    @formats[f.to_sym] = formats.include?(f)
  end
end

def walk_tree__by_book(formats)
  set_format_flags(formats)
  puts "Writing books:"
  (1..@number_of_madahib).each do |m|
    puts "  madhab #{m}"
    (1..@number_of_tafaseer_per_madhab[m-1]).each do |t|
      t0 = Time.now
      print "    tafseer #{t} "
      print 'csv ' if @formats[:csv] || @formats[:csv_nospecialchars]
      print 'sqlite ' if @formats[:sqlite]
      pattern = File.join(@inpath, 'sura_???', 'aaya_???', "madhab_#{"%02d" % m}",
      files = Dir.glob(pattern).sort
      @html = '' # Wipe last BOOK's data
      i = 0 # Line number should be available after loop
      files.each do |infile|
        i += 1 # Next file = next line in the CSV
        @hash = {} # Wipe last FILE's data
        @yaml = YAML.load(File.open(infile))
        flat_hash(@yaml).each do |k,v|
          col = k.join('_')
          @hash[col] = v
        end
        cts_csv_writeline(m, t, i) if @formats[:csv]
        cts_csv_writeline(m, t, i, nospecialchars: true) if @formats[:csv_nospecial
        cts_sqlite_writeline(m, t, i) if @formats[:sqlite]
        html5_addline if @formats[:plain] || @formats[:plain_nohadith] || @formats[
      end # sura, aaya
      html_file = html5_write_unless_exists(m, t)
```

```
        plain_text_write(html_file, m, t) if @formats[:plain]
        plain_text_write(html_file, m, t, nohadith: true) if @formats[:plain_nohadith
        other_formats_write(html_file, m, t, formats) if @formats[:other]
        puts "(%s files, %ss)" % [i, (Time.now-t0).round(1)]
      end # tafaseer
    end # madahib
  end
end

if (ARGV.include?('-h') || ARGV.empty?)
  puts "Usage: ./convert.rb [sqlite|csv|csv_nospecialchars|html5|plain|plain_nohadith
else
  AlTafsirYAMLFiles.convert_to(ARGV)
end
```

V Script 02_analysis/wordcount_by_author.rb

```
#!/usr/bin/env ruby

require 'pathname'

database = Pathname('../../corpora/altafsir_com/processed/corpus.sqlite3')
query = Pathname('./wordcount_whole_corpus.sql')
big_output_file = Pathname('./data_automated/wordcounts/wordcount_ratios_perAuthorPer
command = "/usr/bin/env sqlite3 -header -csv #{database} < #{query} > #{big_output_fi

puts "Running query..."
`#{command}`

# Example line:
# 02-15," ",002:110,6,2864427,0.00021

header = ''

File.readlines(big_output_file).each_with_index do |line,i|
  if i == 0
    header = line
    puts "header is #{header}"
  else
    author = line.split(',')[0]
  end
```

```
  if i > 0
    File.open(Pathname("./data_automated/wordcounts/wordcount_ratios_perPageForAuthor
      if file.size == 0
        puts "writing file for #{author}"
        file.write header
      end
      file.write line
    end
  end
end
```

VI Query 02_analysis/wordcount_whole_corpus.sql

```
--
-- Helper table needed for second query
-- Get rid of it first if it already existed
--
DROP TABLE IF EXISTS wordcounts_by_author;
CREATE TABLE wordcounts_by_author (id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
  author_name CHAR(255) UNIQUE NOT NULL, words INTEGER NOT NULL);

--
-- Determine wordcounts for everything the author wrote
-- Note this doesn't take into consideration duplicate pages!
--
INSERT INTO wordcounts_by_author (author_name, words)
SELECT
  author_name,
  SUM((CASE WHEN LENGTH(text) >= 1
        THEN
          (LENGTH(text) - LENGTH(REPLACE(text, ' ', '')) + 1)
        ELSE
          (LENGTH(text) - LENGTH(REPLACE(text, ' ', '')))
        END)) AS words
FROM cts_units
GROUP BY author_name
ORDER BY words ASC;

--
-- Another helper table to keep things flexible,
-- giant queries are hard as they are...
```

```
--
DROP TABLE IF EXISTS wordcounts_by_unit;
CREATE TABLE wordcounts_by_unit (id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
  text_hash CHAR(255), author_id CHAR(255), author_name CHAR(255), sura INTEGER, aaya

--
-- Now determine wordcounts per page
--
INSERT INTO wordcounts_by_unit (text_hash, author_id, author_name, sura, aaya, words,
SELECT
  text_hash,
  --
  -- CATEGORY AND AUTHOR:
  -- same format as we use everywhere else
  --
  (SUBSTR('00'||category_id,-2,2) ||'-'|| SUBSTR('00'||author_id,-2,2))
    AS author_id,
    U.author_name,
  --
  -- QUR'AN PASSAGE:
  --
  sura_id,
  aaya_id,
  --
  -- AMOUNT OF WORDS SPENT ON EACH AAYA:
  -- we have to use a condition here to get a good value
  -- http://stackoverflow.com/questions/3293790/query-to-count-words-sqlite-3
  --
  (CASE WHEN LENGTH(`text`) >= 1
        THEN
          (LENGTH(`text`) - LENGTH(REPLACE(`text`, ' ', '')) + 1)
        ELSE
          (LENGTH(`text`) - LENGTH(REPLACE(`text`, ' ', '')))
        END)
    AS words_spent,
  --
  -- CHARACTER COUNT FOR AUTHOR'S WHOLE BOOK:
  -- doesn't work without a subquery, correct *word* count would
  -- need condition inside of it again so leaving that be as it
  -- would really push the running time of the query
  --
  -- (W.words)
```

```
  --   AS author_wordcount,
  --
  -- PERCENTAGE OF AAYA WORDCOUNT WRT AUTHOR WORDCOUNT:
  -- getting the ratio is just a simple matter of
  -- dividing the smaller of the two numbers (words spent
  -- on the aaya) by the larger one (words spent on the
  -- whole of the Quran) - unfortunately we must repeat
  -- the words_spent calculation here
  --
  ROUND(100 * ((CASE WHEN LENGTH(`text`) >= 1
        THEN
          (LENGTH(`text`) - LENGTH(REPLACE(`text`, ' ', ''')) + 1)
        ELSE
          (LENGTH(`text`) - LENGTH(REPLACE(`text`, ' ', ''')))
        END) * 1.0 / W.words * 1.0), 5)
    AS percentage
FROM cts_units AS U
JOIN wordcounts_by_author AS W
  ON U.author_name=W.author_name
ORDER BY U.author_name ASC, percentage DESC, words_spent DESC, sura_id ASC, aaya_id A

--
-- Finally ready to get real distinct wordcounts
--
SELECT
  author_id, author_name, text_hash,
  MIN(words) AS words,
  sura,
  MIN(aaya) AS aaya
FROM
  wordcounts_by_unit
GROUP BY author_id, author_name, text_hash, sura
ORDER BY author_id ASC, words DESC, sura ASC, aaya ASC;
```