10 Hirsch indices from Google Scholar

What's going on with the Google Scholar indices (h5-index or `h5`, h5-median or `m5`) of the journals in the SciELO network for each subject area?

In [1]:
```
from collections import Counter
from itertools import accumulate, chain
from statistics import median
```

In [2]:
```
import matplotlib.pyplot as plt
import networkx as nx
import numpy as np
import pandas as pd
import seaborn as sns
```

In [3]:
```
pd.options.display.max_colwidth = 400
pd.options.display.max_rows = 150
sns.set() # Plot style
%matplotlib inline
```

In [4]:
```
# As of 2018-09-18, Seaborn 0.9.0 uses Scipy 1.1.0,
# and "scipy.stats" uses a deprecated indexing style.
# It's just to avoid an annoying warning message,
# and has nothing to do with the code of this notebook.
import warnings
warnings.filterwarnings("ignore", category=FutureWarning)
```

10.1 Loading the dataset

Let's load the dataset with all journals in the SciELO network.

In [5]:
```
journals = pd.read_csv("tabs_network/journals.csv")
```

10.1.1 Column normalization

These are the column names in the raw CSV file:

In [6]:
```
journals.columns
```

Out [6]:
```
Index(['extraction date', 'study unit', 'collection', 'ISSN SciELO', 'ISSN's',
       'title at SciELO', 'title thematic areas',
       'title is agricultural sciences', 'title is applied social sciences',
       'title is biological sciences', 'title is engineering',
       'title is exact and earth sciences', 'title is health sciences',
       'title is human sciences', 'title is linguistics, letters and arts',
       'title is multidisciplinary', 'title current status',
       'title + subtitle SciELO', 'short title SciELO', 'short title ISO',
       'title PubMed', 'publisher name', 'use license', 'alpha frequency',
       'numeric frequency (in months)', 'inclusion year at SciELO',
       'stopping year at SciELO', 'stopping reason',
       'date of the first document', 'volume of the first document',
       'issue of the first document', 'date of the last document',
       'volume of the last document', 'issue of the last document',
       'total of issues', 'issues at 2018', 'issues at 2017', 'issues at 2016',
       'issues at 2015', 'issues at 2014', 'issues at 2013',
       'total of regular issues', 'regular issues at 2018',
```

```
       'regular issues at 2017', 'regular issues at 2016',
       'regular issues at 2015', 'regular issues at 2014',
       'regular issues at 2013', 'total of documents', 'documents at 2018',
       'documents at 2017', 'documents at 2016', 'documents at 2015',
       'documents at 2014', 'documents at 2013', 'citable documents',
       'citable documents at 2018', 'citable documents at 2017',
       'citable documents at 2016', 'citable documents at 2015',
       'citable documents at 2014', 'citable documents at 2013',
       'portuguese documents at 2018 ', 'portuguese documents at 2017 ',
       'portuguese documents at 2016 ', 'portuguese documents at 2015 ',
       'portuguese documents at 2014 ', 'portuguese documents at 2013 ',
       'spanish documents at 2018 ', 'spanish documents at 2017 ',
       'spanish documents at 2016 ', 'spanish documents at 2015 ',
       'spanish documents at 2014 ', 'spanish documents at 2013 ',
       'english documents at 2018 ', 'english documents at 2017 ',
       'english documents at 2016 ', 'english documents at 2015 ',
       'english documents at 2014 ', 'english documents at 2013 ',
       'other language documents at 2018 ',
       'other language documents at 2017 ',
       'other language documents at 2016 ',
       'other language documents at 2015 ',
       'other language documents at 2014 ',
       'other language documents at 2013 ', 'google scholar h5 2018 ',
       'google scholar h5 2017 ', 'google scholar h5 2016 ',
       'google scholar h5 2015 ', 'google scholar h5 2014 ',
       'google scholar h5 2013 ', 'google scholar m5 2018 ',
       'google scholar m5 2017 ', 'google scholar m5 2016 ',
       'google scholar m5 2015 ', 'google scholar m5 2014 ',
       'google scholar m5 2013 '],
      dtype='object')
```

The column names aren't helping us with all the small details like the trailing whitespace in the fields whose name starts with `google scholar`. The easiest approach is to run this normalization function from the column names simplification notebook:

In [7]:
```python
def normalize_column_title(name):
    import re
    name_unbracketed = re.sub(r".*\((.*)\)", r"\1",
                              name.replace("(in months)", "in_months"))
    words = re.sub("[^a-z0-9+_ ]", "", name_unbracketed.lower()).split()
    ignored_words = ("at", "the", "of", "and", "google", "scholar", "+")
    replacements = {
        "document": "doc",
        "documents": "docs",
        "frequency": "freq",
        "language": "lang",
    }
    return "_".join(replacements.get(word, word)
                    for word in words if word not in ignored_words) \
              .replace("title_is", "is")
```

Applying it is straightforward, and the order of the columns is kept as is.

In [8]:
```python
journals.rename(columns=normalize_column_title, inplace=True)
journals.columns
```

Out [8]:
```
Index(['extraction_date', 'study_unit', 'collection', 'issn_scielo', 'issns',
       'title_scielo', 'title_thematic_areas', 'is_agricultural_sciences',
```

```
       'is_applied_social_sciences', 'is_biological_sciences',
       'is_engineering', 'is_exact_earth_sciences', 'is_health_sciences',
       'is_human_sciences', 'is_linguistics_letters_arts',
       'is_multidisciplinary', 'title_current_status', 'title_subtitle_scielo',
       'short_title_scielo', 'short_iso', 'title_pubmed', 'publisher_name',
       'use_license', 'alpha_freq', 'numeric_freq_in_months',
       'inclusion_year_scielo', 'stopping_year_scielo', 'stopping_reason',
       'date_first_doc', 'volume_first_doc', 'issue_first_doc',
       'date_last_doc', 'volume_last_doc', 'issue_last_doc', 'total_issues',
       'issues_2018', 'issues_2017', 'issues_2016', 'issues_2015',
       'issues_2014', 'issues_2013', 'total_regular_issues',
       'regular_issues_2018', 'regular_issues_2017', 'regular_issues_2016',
       'regular_issues_2015', 'regular_issues_2014', 'regular_issues_2013',
       'total_docs', 'docs_2018', 'docs_2017', 'docs_2016', 'docs_2015',
       'docs_2014', 'docs_2013', 'citable_docs', 'citable_docs_2018',
       'citable_docs_2017', 'citable_docs_2016', 'citable_docs_2015',
       'citable_docs_2014', 'citable_docs_2013', 'portuguese_docs_2018',
       'portuguese_docs_2017', 'portuguese_docs_2016', 'portuguese_docs_2015',
       'portuguese_docs_2014', 'portuguese_docs_2013', 'spanish_docs_2018',
       'spanish_docs_2017', 'spanish_docs_2016', 'spanish_docs_2015',
       'spanish_docs_2014', 'spanish_docs_2013', 'english_docs_2018',
       'english_docs_2017', 'english_docs_2016', 'english_docs_2015',
       'english_docs_2014', 'english_docs_2013', 'other_lang_docs_2018',
       'other_lang_docs_2017', 'other_lang_docs_2016', 'other_lang_docs_2015',
       'other_lang_docs_2014', 'other_lang_docs_2013', 'h5_2018', 'h5_2017',
       'h5_2016', 'h5_2015', 'h5_2014', 'h5_2013', 'm5_2018', 'm5_2017',
       'm5_2016', 'm5_2015', 'm5_2014', 'm5_2013'],
      dtype='object')
```

10.1.2 Thematic areas

Based on the thematic area normalization notebook, these are the columns for each thematic area after column name normalization:

In [9]:
```python
areas_map = {
    "Agricultural Sciences": "is_agricultural_sciences",
    "Applied Social Sciences": "is_applied_social_sciences",
    "Biological Sciences": "is_biological_sciences",
    "Engineering": "is_engineering",
    "Exact and Earth Sciences": "is_exact_earth_sciences",
    "Health Sciences": "is_health_sciences",
    "Human Sciences": "is_human_sciences",
    "Linguistics, Letters and Arts": "is_linguistics_letters_arts",
}
areas = list(areas_map.values())
areas
```

Out [9]:
```
['is_agricultural_sciences',
 'is_applied_social_sciences',
 'is_biological_sciences',
 'is_engineering',
 'is_exact_earth_sciences',
 'is_health_sciences',
 'is_human_sciences',
 'is_linguistics_letters_arts']
```

One missing is the `is_multidisciplinary`, which is 1 if the journal has at least 3 of the distinct thematic areas above, otherwise it's 0.

In [10]:
```
areaswm = areas + ["is_multidisciplinary"]
```

10.1.3 Normalization

We'll need the ISSN and the thematic areas, so these should be normalized. The code below follows the normalization notebooks for these fields.

In [11]:
```
# ISSN normalization
issn_scielo_fix = {"0001-6002": "0001-6012",
                   "0258-6444": "2215-3535",
                   "0325-8203": "1668-7027",
                   "0719-448x": "0719-448X",
                   "0797-9789": "1688-499X",
                   "0807-8967": "0870-8967",
                   "0858-6444": "0258-6444",
                   "1315-5216": "1316-5216",
                   "1667-8682": "1667-8982",
                   "1678-5177": "0103-6564",
                   "1683-0789": "1683-0768",
                   "1688-4094": "1688-4221",
                   "1852-4418": "1852-4184",
                   "1980-5438": "0103-5665",
                   "2175-3598": "0104-1282",
                   "2233-7666": "2223-7666",
                   "2237-101X": "1518-3319",
                   "24516600": "2451-6600",
                   "2993-6797": "2393-6797"}
journals["issn_scielo"].replace(issn_scielo_fix, inplace=True)
```

In [12]:
```
# Thematic area normalization
tta_map = journals.groupby("issn_scielo").apply(
    lambda df: df.assign(title_thematic_areas=df["title_thematic_areas"]
                                              .replace("Psicanalise",
                                                       "Human Sciences")
                                              .fillna("Human Sciences"),
                         order=df["collection"].isin(["sss", "psi"]) |
                               (df["title_thematic_areas"] == "Psicanalise") |
                               df["title_thematic_areas"].isna())
                 .sort_values("order")["title_thematic_areas"].iloc[0]
)
tta_text_n = journals["issn_scielo"].map(tta_map) \
                                    .rename("title_thematic_areas")
tta_list_n = tta_text_n.str.split(";")
tta_n = pd.DataFrame(tta_text_n).assign(**{
    area: tta_list_n.apply((lambda n: lambda entries: int(n in entries))(name))
    for name, area in areas_map.items()
}).assign(
    is_multidisciplinary=lambda df: (df[areas].sum(axis=1) >= 3).map(int)
)
journals = journals.assign(**tta_n)
```

10.1.4 Selecting the Google Scholar fields

The fields regarding to the h5/m5 indices from Google Scholar are:

In [13]:
```
h5_fields = sorted(k for k in journals.columns if k.startswith("h5"))
m5_fields = sorted(k for k in journals.columns if k.startswith("m5"))
h5_fields, m5_fields
```

Out [13]:
```
(['h5_2013', 'h5_2014', 'h5_2015', 'h5_2016', 'h5_2017', 'h5_2018'],
 ['m5_2013', 'm5_2014', 'm5_2015', 'm5_2016', 'm5_2017', 'm5_2018'])
```

Are the h5/m5 fields consistent?

In [14]:
```
h5m5_inconsistency = journals.groupby("issn_scielo")[h5_fields + m5_fields] \
    .apply(lambda df: df.apply(lambda col: set(col.dropna()))
                        .apply(len).max() > 1)
h5m5_inconsistency[h5m5_inconsistency]
```

Out [14]:
```
Series([], dtype: bool)
```

Yes, they are! The only requirement we have regarding these fields is to ignore any possible `NaN` values.

Does every column have information?

In [15]:
```
h5m5_is_empty = journals[h5_fields + m5_fields].count() == 0
h5m5_empty_list = list(h5m5_is_empty[h5m5_is_empty].index)
h5m5_empty_list
```

Out [15]:
```
['h5_2018', 'm5_2018']
```

No! These two empty columns shown above have no use for us.

In [16]:
```
h5_fields = [k for k in h5_fields if k not in h5m5_empty_list]
m5_fields = [k for k in m5_fields if k not in h5m5_empty_list]
gs_fields = list(chain(*zip(h5_fields, m5_fields)))
gs_fields
```

Out [16]:
```
['h5_2013',
 'm5_2013',
 'h5_2014',
 'm5_2014',
 'h5_2015',
 'm5_2015',
 'h5_2016',
 'm5_2016',
 'h5_2017',
 'm5_2017']
```

10.1.5 Data de-duplication (building the dataset)

A missing normalization step regards to this issue: there should be no more than a single entry for each journal. Since there are many journals that in more than one collection, we have data duplication. The normalization step already enforced that any of these rows have the same thematic area, besides a common `issn_scielo` that can be regarded as its *primary key*. As we've seen in the Google Scholar indices section, the indices are consistent, as long as we coalesce the `NaN` entries.

Now that we're aware of that, let's remove the duplicate entries.

The renamed columns we'll need are (these names are the ones after the renaming step):

In [17]:
```
columns = ["issn_scielo"] + gs_fields + areaswm
columns
```

Out [17]:
```
['issn_scielo',
 'h5_2013',
 'm5_2013',
 'h5_2014',
 'm5_2014',
 'h5_2015',
 'm5_2015',
 'h5_2016',
 'm5_2016',
 'h5_2017',
 'm5_2017',
 'is_agricultural_sciences',
 'is_applied_social_sciences',
 'is_biological_sciences',
 'is_engineering',
 'is_exact_earth_sciences',
 'is_health_sciences',
 'is_human_sciences',
 'is_linguistics_letters_arts',
 'is_multidisciplinary']
```

Since everything is already normalized, getting the dataset is straightforward:

In [18]:
```python
dataset = journals[columns].groupby("issn_scielo").agg("max")
dataset.head().T
```

Out [18]:

issn_scielo	0001-3714	0001-3765	0001-6012	0001-6365	0002-0591
h5_2013	NaN	19.0	NaN	NaN	3.0
m5_2013	NaN	28.0	NaN	NaN	5.0
h5_2014	NaN	19.0	NaN	NaN	3.0
m5_2014	NaN	31.0	NaN	NaN	5.0
h5_2015	NaN	19.0	NaN	NaN	4.0
m5_2015	NaN	24.0	NaN	NaN	4.0
h5_2016	NaN	18.0	6.0	NaN	5.0
m5_2016	NaN	25.0	7.0	NaN	5.0
h5_2017	NaN	16.0	7.0	NaN	5.0
m5_2017	NaN	19.0	11.0	NaN	6.0
is_agricultural_sciences	1.0	1.0	0.0	0.0	0.0
is_applied_social_sciences	0.0	0.0	0.0	0.0	0.0
is_biological_sciences	1.0	1.0	0.0	0.0	0.0
is_engineering	0.0	1.0	0.0	0.0	0.0
is_exact_earth_sciences	0.0	1.0	0.0	0.0	0.0
is_health_sciences	0.0	1.0	1.0	1.0	0.0
is_human_sciences	0.0	1.0	1.0	0.0	1.0
is_linguistics_letters_arts	0.0	0.0	0.0	0.0	0.0
is_multidisciplinary	0.0	1.0	0.0	0.0	0.0

There's a lot of `NaN`s! Let's see what we can do with that information.

10.2 Understanding the Hirsch index

The dataset is ready, but before we start exploring the data, it makes sense to know what's the meaning of the Google Scholar indices columns.

10.2.1 Definition

In the Google Scholar top publications web page[1] we can see the definition of the *h5-index* and *h5-median* (the `h5_*` and `m5_*` columns of the dataset, respectively). It defines:

> h5-index is the h-index for articles published in the last 5 complete years. It is the largest number h such that h articles published in 2013-2017 have at least h citations each.
>
> h5-median for a publication is the median number of citations for the articles that make up its h5-index.

The Hirsch index for a graph is:

$$h_{journal} = \max_{node_i \in journal} \left\{ \min \left[\text{in-degree}(node_i), \sum_{\substack{node_j \in journal \\ \text{in-degree}(node_j) \geq \text{in-degree}(node_i)}} 1 \right] \right\}$$

Where the Google Scholar's `h5-index` is just the Hirsch index for a journal in the directed graph of articles (nodes) connected by citations (edges) in the last completed 5 years, including the edges coming from other journals. The *in-degree* of an article is the number of citations it received in the network.

And, by the above definition, the Google Scholar's `h5-median` is:

$$\operatorname*{median}_{\substack{node_i \in journal \\ \text{in-degree}(node_i) \geq h_{journal}}} [\text{in-degree}(node_i)]$$

Which means that h5-median $\geq$ h5-index.

10.2.2 Simplified explanation of the Hirsch index definition

The mathematical definition of the index might *look* complicated, but thinking on it iteratively and avoiding the graph theory jargon (*degree* and *node*) might make its logic quite straightforward:

- The context is the citation network of all articles from 2013 to 2017 in Google Scholar.
- Let's select the nodes of a specific journal J;
- The Hirsch index couldn't be less than 0, so let's say it's at least 0;
- Is there at least 1 article from J receiving at least 1 citation in the network? If yes, then the Hirsch index is at least 1;
- Are there at least 2 articles from J receiving at least 2 citations in the network? If yes, then the Hirsch index is at least 2;
- ...
- Are there at least h articles from J receiving at least h citations in the network? If yes, then the Hirsch index is at least h;
- Are there at least $h+1$ articles from J receiving at least $h+1$ citations in the network? If not, then the Hirsch index is smaller than $h+1$, and since it's a whole number, it's the last ceiling we've found: h.

10.2.3 Implementation using a NetworkX directed graph

Given the citation graph as an instance of `nx.DiGraph` and a selected set of nodes (the journal's articles in the graph), the following functions calculate the `h5-index` and `h5-median`.

In [19]:
```
def h_index(graph, nodes=None):
    if nodes is None:
        nodes = graph.nodes
```

[1] https://scholar.google.com/citations?view_op=top_venues

```
    degree_counts = Counter(degree for node, degree in graph.in_degree
                                if node in nodes)
    degrees, counts = zip(*sorted(degree_counts.items(), reverse=True))
    cum_counts = accumulate(counts)
    return max(min(degree, cc) for degree, cc in zip(degrees, cum_counts))
```

In [20]:
```
def h_median(graph, nodes=None, h=None):
    if nodes is None:
        nodes = graph.nodes
    if h is None:
        h = h_index(graph, nodes)
    return median(degree for node, degree in graph.in_degree
                     if node in nodes and degree >= h)
```

By default, these functions use all nodes in the graph, and since the median calculation depends on the Hirsch index, it's also an optional input (by default, it'll calculate the index).

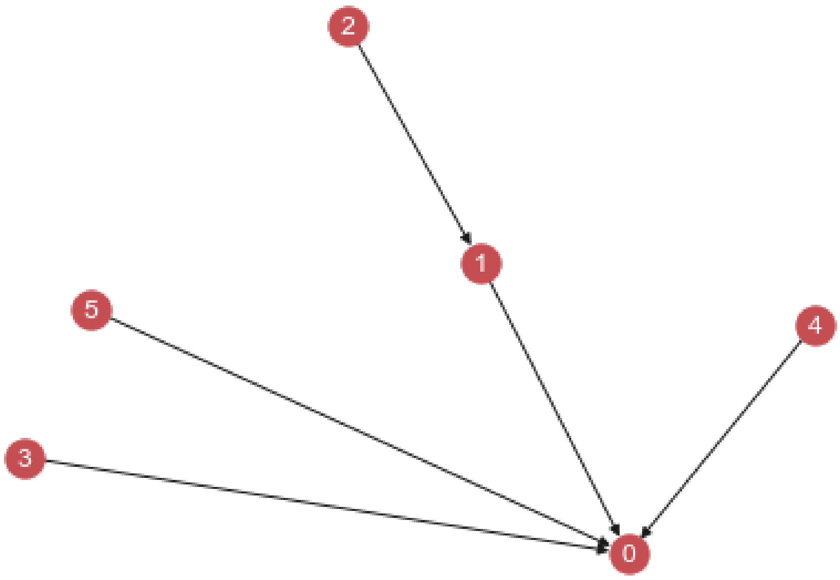
10.2.4 Example

This dataset doesn't have a full citation graph including citations from external journals, so we will just create arbitrary/random graphs as examples for the calculation of these indices.

Suppose we have a citation network of a single journal, like this:

In [21]:
```
arbitrary_citation_graph = nx.DiGraph([(1, 0), (2, 1), (3, 0), (4, 0), (5, 0)])
nx.draw(arbitrary_citation_graph,
        pos=nx.spring_layout(arbitrary_citation_graph,
                             seed=42), # "random_state" in older NetworkX
        font_color="w",
        with_labels=True)
print(f"h5-index: {h_index(arbitrary_citation_graph)}")
print(f"h5-median: {h_median(arbitrary_citation_graph)}")
```

```
h5-index: 1
h5-median: 2.5
```



The number of citations for each numbered node from 0 to 5 is:

In [22]:
```
[degree for node, degree in sorted(arbitrary_citation_graph.in_degree)]
```

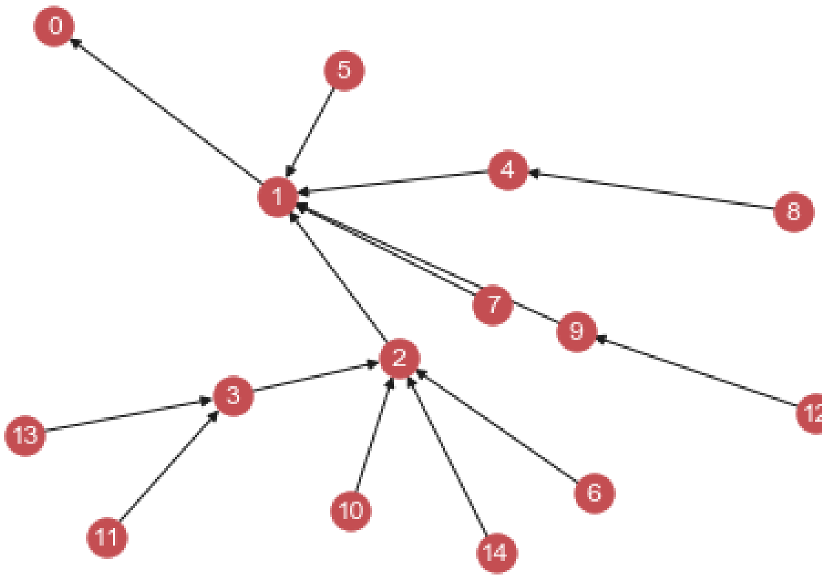
Out [22]:
```
[4, 1, 0, 0, 0, 0]
```

The Hirsch index couldn't be 2 as there's only a single article with more than a single citation.

Another example:

In [23]:
```
random_citation_graph = nx.gn_graph(15, seed=0)
nx.draw(random_citation_graph,
        pos=nx.spring_layout(random_citation_graph, seed=42),
        font_color="w",
        with_labels=True)
print(f"h5-index: {h_index(random_citation_graph)}")
print(f"h5-median: {h_median(random_citation_graph)}")
```

```
h5-index: 2
h5-median: 4
```



For each numbered node sorted in ascending order, the number of citations is:

In [24]:
```
[degree for node, degree in sorted(random_citation_graph.in_degree)]
```

Out [24]:
```
[1, 5, 4, 2, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]
```

The only entries greater or equal to 2 (the Hirsch index) are $[5,4,2]$, whose median is 4. If that entry with 2 citations had been received 3 citations instead, this graph would have an Hirsch index of 3.

Let the even and odd indexed nodes be from two distinct journals, then this analysis of each journal would be:

- $[1,4,1,0,0,0,0,0]$ citations for each article, h5-index $= 1$ and h5-median $= 1$;
- $[5,2,0,0,1,0,0]$ citations for each article, h5-index $= 2$ and h5-median $= 3.5$.

10.3 Exploratory data analysis

Though we have information about every thematic area, we don't have the Google Scholar indices for all the documents/articles:

In [25]:
```
print(dataset.shape)
dataset.count()
```

```
(1653, 19)
```

Out [25]:
```
h5_2013                265
m5_2013                265
h5_2014                266
m5_2014                266
h5_2015                803
m5_2015                803
h5_2016                772
```

```
m5_2016                        772
h5_2017                        923
m5_2017                        923
is_agricultural_sciences      1653
is_applied_social_sciences    1653
is_biological_sciences        1653
is_engineering                1653
is_exact_earth_sciences       1653
is_health_sciences            1653
is_human_sciences             1653
is_linguistics_letters_arts   1653
is_multidisciplinary          1653
dtype: int64
```

The `h5` and `m5` are pairs, having one means we have the other one.

We have that index for $\approx 55.8\%$ of the data in 2017.

In [26]:
```python
dataset["h5_2017"].count() / dataset.shape[0]
```

Out [26]:
```
0.55837870538415
```

We should emphasize the 2017 data, as it has way more information than the remaining of the data.

10.3.1 Data summary

Let's get all the most usual descriptive statistics for single areas in this dataset. The sum isn't 100% as several journals regards to more than a single area.

In [27]:
```python
ddata = pd.concat([
    dataset[dataset[area] == 1][gs_fields].describe().assign(area=area[3:])
    for area in areaswm
])
ddata
```

Out [27]:

The table is in the next page ...

	h5_2013	m5_2013	h5_2014	m5_2014	h5_2015	m5_2015	h5_2016	m5_2016	h5_2017	m5_2017	area
count	35.000000	35.000000	36.000000	36.000000	76.000000	76.000000	82.000000	82.000000	92.000000	92.000000	agricultural_sciences
mean	12.285714	15.742857	11.111111	14.138889	9.644737	12.328947	9.817073	12.975610	10.043478	13.304348	agricultural_sciences
std	5.159636	6.386304	4.833087	6.564636	4.813049	5.888720	4.343657	5.989653	4.185042	5.704710	agricultural_sciences
min	3.000000	4.000000	2.000000	2.000000	3.000000	4.000000	3.000000	3.000000	2.000000	2.000000	agricultural_sciences
25%	9.000000	12.000000	8.750000	11.000000	6.000000	8.000000	6.000000	8.000000	7.000000	9.000000	agricultural_sciences
50%	11.000000	14.000000	11.000000	13.000000	8.000000	11.000000	9.000000	11.500000	9.000000	13.000000	agricultural_sciences
75%	15.000000	19.500000	13.250000	16.000000	12.250000	15.000000	12.750000	16.750000	13.000000	17.000000	agricultural_sciences
max	25.000000	31.000000	23.000000	31.000000	21.000000	28.000000	21.000000	31.000000	20.000000	31.000000	agricultural_sciences
count	31.000000	31.000000	27.000000	27.000000	162. 000000	162. 000000	174. 000000	174. 000000	192. 000000	192. 000000	applied_social_sciences
mean	8.032258	12.129032	6.814815	10.407407	6.895062	10.055556	7.063218	10.235632	7.734375	11.052083	applied_social_sciences
std	3.610022	5.925886	4.123451	5.534919	3.169356	4.961028	3.528828	5.728611	3.774711	5.720121	applied_social_sciences
min	3.000000	5.000000	1.000000	3.000000	2.000000	3.000000	1.000000	1.000000	1.000000	3.000000	applied_social_sciences
25%	5.000000	8.000000	3.000000	5.500000	5.000000	7.000000	5.000000	6.000000	5.000000	7.000000	applied_social_sciences
50%	8.000000	11.000000	7.000000	9.000000	6.000000	9.000000	6.000000	9.000000	7.000000	10.000000	applied_social_sciences
75%	10.000000	14.500000	9.500000	14.000000	9.000000	12.000000	9.000000	13.000000	10.000000	14.000000	applied_social_sciences
max	19.000000	32.000000	17.000000	24.000000	20.000000	32.000000	20.000000	32.000000	22.000000	38.000000	applied_social_sciences
count	29.000000	29.000000	30.000000	30.000000	93.000000	93.000000	91.000000	91.000000	102. 000000	102. 000000	biological_sciences
mean	12.793103	17.137931	13.733333	18.133333	9.946237	12.924731	10.670330	14.384615	10.784314	14.333333	biological_sciences
std	6.281374	8.617773	5.771113	8.881338	5.061289	6.369487	5.450902	7.282810	5.285188	7.188182	biological_sciences
min	3.000000	4.000000	6.000000	8.000000	4.000000	5.000000	3.000000	4.000000	4.000000	5.000000	biological_sciences
25%	9.000000	12.000000	9.000000	11.250000	6.000000	8.000000	7.000000	9.000000	7.000000	10.000000	biological_sciences
50%	11.000000	17.000000	13.500000	16.500000	8.000000	11.000000	9.000000	13.000000	9.500000	12.000000	biological_sciences
75%	17.000000	22.000000	16.000000	21.750000	12.000000	16.000000	14.000000	18.500000	14.000000	18.000000	biological_sciences
max	32.000000	46.000000	33.000000	52.000000	25.000000	34.000000	29.000000	38.000000	29.000000	39.000000	biological_sciences
count	16.000000	16.000000	21.000000	21.000000	60.000000	60.000000	60.000000	60.000000	71.000000	71.000000	engineering
mean	9.312500	12.125000	8.666667	12.285714	7.583333	10.100000	8.016667	10.750000	8.366197	11.591549	engineering
std	4.757012	6.064926	4.963198	6.671903	4.334691	5.601755	4.831014	6.374273	4.645242	6.370866	engineering
min	2.000000	4.000000	2.000000	4.000000	2.000000	3.000000	2.000000	2.000000	3.000000	3.000000	engineering
25%	6.750000	8.750000	5.000000	7.000000	4.000000	6.000000	4.750000	7.000000	5.000000	7.000000	engineering
50%	8.500000	11.500000	7.000000	11.000000	7.000000	8.500000	7.000000	9.000000	7.000000	10.000000	engineering
75%	12.250000	15.250000	13.000000	15.000000	9.250000	13.000000	9.250000	13.000000	10.500000	14.000000	engineering
max	19.000000	28.000000	19.000000	31.000000	19.000000	24.000000	21.000000	31.000000	21.000000	31.000000	engineering

Continued on next page

	h5_2013	m5_2013	h5_2014	m5_2014	h5_2015	m5_2015	h5_2016	m5_2016	h5_2017	m5_2017	area
count	14.000000	14.000000	16.000000	16.000000	58.000000	58.000000	60.000000	60.000000	64.000000	64.000000	exact_earth_sciences
mean	10.857143	14.428571	10.875000	15.500000	8.724138	11.637931	8.866667	12.066667	9.406250	12.562500	exact_earth_sciences
std	5.815856	8.149698	5.806605	9.040649	4.644533	5.981546	4.670378	6.655867	4.655493	6.334273	exact_earth_sciences
min	2.000000	3.000000	3.000000	5.000000	3.000000	4.000000	1.000000	1.000000	2.000000	3.000000	exact_earth_sciences
25%	8.000000	10.250000	6.750000	10.000000	6.000000	8.000000	5.000000	7.000000	6.000000	8.000000	exact_earth_sciences
50%	9.500000	11.500000	10.500000	13.500000	7.000000	10.000000	7.000000	10.000000	9.000000	11.000000	exact_earth_sciences
75%	14.000000	19.000000	12.750000	16.750000	11.000000	13.750000	11.000000	15.000000	11.250000	15.250000	exact_earth_sciences
max	24.000000	31.000000	25.000000	38.000000	24.000000	31.000000	23.000000	31.000000	25.000000	31.000000	exact_earth_sciences
count	97.000000	97.000000	91.000000	91.000000	266.000000	266.000000	210.000000	210.000000	286.000000	286.000000	health_sciences
mean	14.886598	20.041237	14.274725	19.274725	11.127820	15.060150	12.147619	16.414286	11.741259	16.000000	health_sciences
std	7.908166	10.997081	9.056693	13.851647	7.747344	10.407733	8.262564	11.484326	7.460035	10.201479	health_sciences
min	4.000000	5.000000	2.000000	2.000000	1.000000	2.000000	2.000000	2.000000	1.000000	1.000000	health_sciences
25%	9.000000	12.000000	7.000000	9.000000	6.000000	8.000000	6.000000	8.000000	7.000000	9.000000	health_sciences
50%	14.000000	19.000000	12.000000	16.000000	9.000000	12.000000	10.000000	13.000000	9.000000	13.500000	health_sciences
75%	19.000000	26.000000	20.000000	25.500000	14.000000	19.000000	16.000000	21.000000	14.750000	19.000000	health_sciences
max	38.000000	56.000000	39.000000	76.000000	53.000000	72.000000	53.000000	81.000000	54.000000	75.000000	health_sciences
count	73.000000	73.000000	76.000000	76.000000	222.000000	222.000000	238.000000	238.000000	272.000000	272.000000	human_sciences
mean	9.328767	13.191781	8.434211	11.855263	7.675676	10.801802	7.991597	11.432773	8.180147	11.591912	human_sciences
std	4.564521	6.428361	5.342498	7.346117	4.385625	6.513223	4.493776	6.645459	4.593503	6.621474	human_sciences
min	2.000000	3.000000	1.000000	2.000000	1.000000	1.000000	1.000000	1.000000	2.000000	2.000000	human_sciences
25%	5.000000	7.000000	4.000000	6.000000	4.000000	6.000000	4.000000	6.000000	5.000000	7.000000	human_sciences
50%	10.000000	13.000000	7.500000	11.000000	7.000000	9.000000	7.000000	10.000000	7.000000	10.000000	human_sciences
75%	12.000000	18.000000	12.000000	17.250000	10.000000	14.000000	10.000000	16.000000	11.000000	15.000000	human_sciences
max	21.000000	28.000000	23.000000	31.000000	22.000000	35.000000	23.000000	33.000000	23.000000	35.000000	human_sciences
count	6.000000	6.000000	8.000000	8.000000	37.000000	37.000000	34.000000	34.000000	45.000000	45.000000	linguistics_letters_arts
mean	5.000000	8.666667	3.125000	4.750000	4.594595	6.324324	5.500000	8.029412	5.577778	7.977778	linguistics_letters_arts
std	2.449490	5.006662	1.356203	3.370036	2.140346	3.520212	2.402650	3.857188	2.996125	4.303745	linguistics_letters_arts
min	3.000000	4.000000	2.000000	2.000000	1.000000	1.000000	2.000000	3.000000	1.000000	2.000000	linguistics_letters_arts
25%	3.250000	6.000000	2.000000	2.000000	3.000000	3.000000	4.000000	5.000000	4.000000	5.000000	linguistics_letters_arts
50%	4.000000	6.000000	3.000000	3.000000	5.000000	6.000000	5.000000	7.500000	5.000000	7.000000	linguistics_letters_arts
75%	6.250000	12.000000	3.250000	7.500000	5.000000	8.000000	6.750000	10.000000	7.000000	9.000000	linguistics_letters_arts
max	9.000000	16.000000	6.000000	10.000000	9.000000	16.000000	11.000000	16.000000	16.000000	23.000000	linguistics_letters_arts
count	5.000000	5.000000	6.000000	6.000000	27.000000	27.000000	28.000000	28.000000	32.000000	32.000000	multidisciplinary

Continued on next page

	h5_2013	m5_2013	h5_2014	m5_2014	h5_2015	m5_2015	h5_2016	m5_2016	h5_2017	m5_2017	area
mean	12.600000	17.400000	11.833333	16.833333	8.518519	11.444444	9.500000	12.821429	9.687500	13.562500	multidisciplinary
std	4.335897	6.804410	4.622409	7.704977	4.660558	5.535434	5.245810	7.237012	5.012484	7.237615	multidisciplinary
min	9.000000	11.000000	8.000000	10.000000	4.000000	5.000000	3.000000	4.000000	4.000000	5.000000	multidisciplinary
25%	9.000000	12.000000	8.250000	12.500000	5.000000	7.500000	5.750000	7.750000	5.750000	8.500000	multidisciplinary
50%	11.000000	17.000000	10.000000	14.000000	7.000000	10.000000	8.000000	10.000000	8.000000	11.500000	multidisciplinary
75%	15.000000	19.000000	14.750000	18.500000	10.000000	13.500000	13.250000	18.000000	13.750000	18.250000	multidisciplinary
max	19.000000	28.000000	19.000000	31.000000	19.000000	24.000000	21.000000	31.000000	20.000000	31.000000	multidisciplinary

10.3.2 Heat map plotting

This function is a helper for the heat map plots of the subsections that follows.

In [28]:
```
def hmap(data, title):
    fig, (ax1, ax2) = plt.subplots(ncols=2, figsize=(10, 6), sharey=True)
    sns.heatmap(data[h5_fields], ax=ax1,
                cmap="rainbow", annot=True, fmt="g", cbar=False)
    ax1.set(title=f"h5-index ({title})")
    sns.heatmap(data[m5_fields], ax=ax2,
                cmap="rainbow", annot=True, fmt="g", cbar=False)
    ax2.set(title=f"h5-median ({title})")
    fig.tight_layout()
```

Each subsection that follows tries to plot the common variability measure of some measure just after the measure.

The main goal is to just make sense of the data.

Count

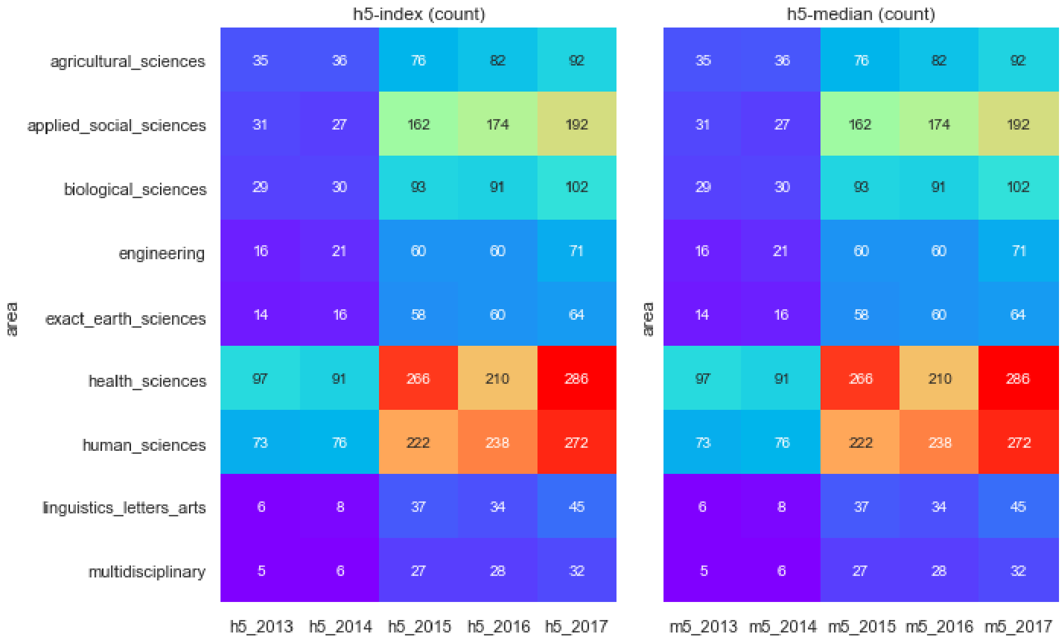
How many entries with the Google Scholar indices data are there for each thematic area?

In [29]:
```
dataset_counts = ddata.loc["count"].set_index("area")
hmap(dataset_counts, "count")
dataset_counts.astype(int)
```

Out [29]:

The table is in the next page ...

area	h5_2013	m5_2013	h5_2014	m5_2014	h5_2015	m5_2015	h5_2016	m5_2016	h5_2017	m5_2017
agricultural_sciences	35	35	36	36	76	76	82	82	92	92
applied_social_sciences	31	31	27	27	162	162	174	174	192	192
biological_sciences	29	29	30	30	93	93	91	91	102	102
engineering	16	16	21	21	60	60	60	60	71	71
exact_earth_sciences	14	14	16	16	58	58	60	60	64	64
health_sciences	97	97	91	91	266	266	210	210	286	286
human_sciences	73	73	76	76	222	222	238	238	272	272
linguistics_letters_arts	6	6	8	8	37	37	34	34	45	45
multidisciplinary	5	5	6	6	27	27	28	28	32	32

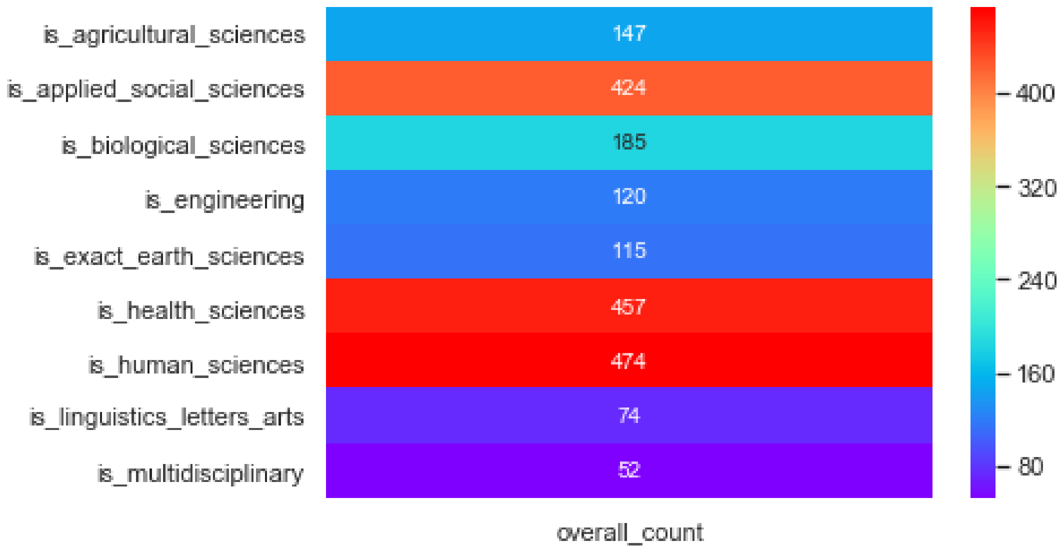


This can be compared with the overall counts:

In [30]:
```
overall_counts = pd.DataFrame(dataset[areaswm].sum().rename("overall_count"))
sns.heatmap(overall_counts, cmap="rainbow", annot=True, fmt="g")
overall_counts
```

Out [30]:

	overall_count
is_agricultural_sciences	147
is_applied_social_sciences	424
is_biological_sciences	185
is_engineering	120
is_exact_earth_sciences	115
is_health_sciences	457
is_human_sciences	474
is_linguistics_letters_arts	74
is_multidisciplinary	52



Mean

Central tendency

In [31]:
```
dataset_means = ddata.loc["mean"].set_index("area")
hmap(dataset_means, "mean")
dataset_means
```

Out [31]:

The table is in the next page ...

area	h5_2013	m5_2013	h5_2014	m5_2014	h5_2015	m5_2015	h5_2016	m5_2016	h5_2017	m5_2017
agricultural_sciences	12.285714	15.742857	11.111111	14.138889	9.644737	12.328947	9.817073	12.975610	10.043478	13.304348
applied_social_sciences	8.032258	12.129032	6.814815	10.407407	6.895062	10.055556	7.063218	10.235632	7.734375	11.052083
biological_sciences	12.793103	17.137931	13.733333	18.133333	9.946237	12.924731	10.670330	14.384615	10.784314	14.333333
engineering	9.312500	12.125000	8.666667	12.285714	7.583333	10.100000	8.016667	10.750000	8.366197	11.591549
exact_earth_sciences	10.857143	14.428571	10.875000	15.500000	8.724138	11.637931	8.866667	12.066667	9.406250	12.562500
health_sciences	14.886598	20.041237	14.274725	19.274725	11.127820	15.060150	12.147619	16.414286	11.741259	16.000000
human_sciences	9.328767	13.191781	8.434211	11.855263	7.675676	10.801802	7.991597	11.432773	8.180147	11.591912
linguistics_letters_arts	5.000000	8.666667	3.125000	4.750000	4.594595	6.324324	5.500000	8.029412	5.577778	7.977778
multidisciplinary	12.600000	17.400000	11.833333	16.833333	8.518519	11.444444	9.500000	12.821429	9.687500	13.562500

Standard deviation

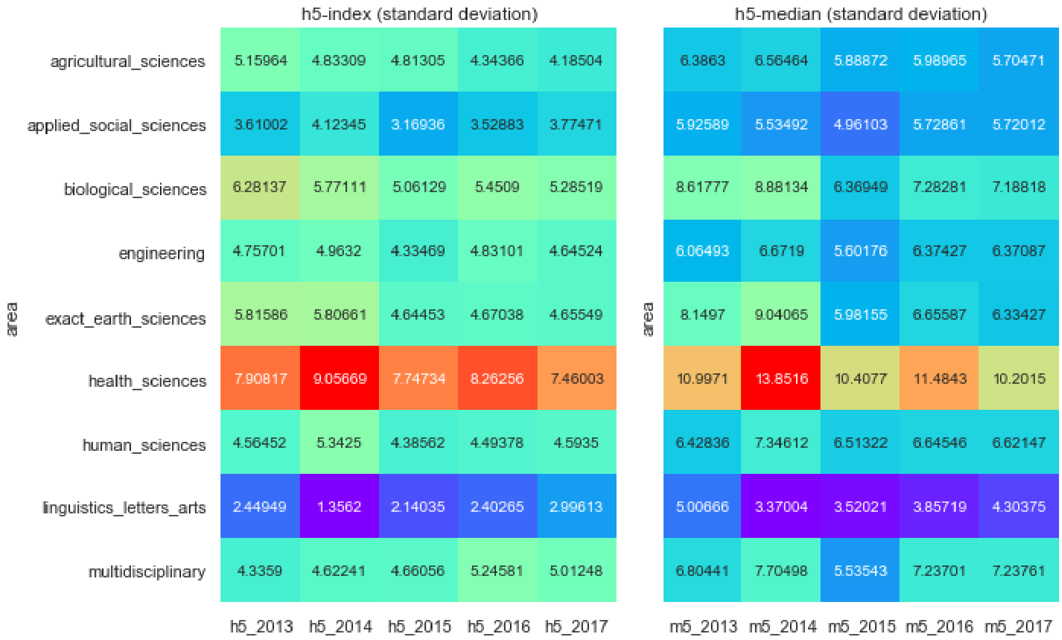
Variability

In [32]:
```
dataset_stds = ddata.loc["std"].set_index("area")
hmap(dataset_stds, "standard deviation")
dataset_stds
```

Out [32]:

The table is in the next page ...

area	h5_2013	m5_2013	h5_2014	m5_2014	h5_2015	m5_2015	h5_2016	m5_2016	h5_2017	m5_2017
agricultural_sciences	5.159636	6.386304	4.833087	6.564636	4.813049	5.888720	4.343657	5.989653	4.185042	5.704710
applied_social_sciences	3.610022	5.925886	4.123451	5.534919	3.169356	4.961028	3.528828	5.728611	3.774711	5.720121
biological_sciences	6.281374	8.617773	5.771113	8.881338	5.061289	6.369487	5.450902	7.282810	5.285188	7.188182
engineering	4.757012	6.064926	4.963198	6.671903	4.334691	5.601755	4.831014	6.374273	4.645242	6.370866
exact_earth_sciences	5.815856	8.149698	5.806605	9.040649	4.644533	5.981546	4.670378	6.655867	4.655493	6.334273
health_sciences	7.908166	10.997081	9.056693	13.851647	7.747344	10.407733	8.262564	11.484326	7.460035	10.201479
human_sciences	4.564521	6.428361	5.342498	7.346117	4.385625	6.513223	4.493776	6.645459	4.593503	6.621474
linguistics_letters_arts	2.449490	5.006662	1.356203	3.370036	2.140346	3.520212	2.402650	3.857188	2.996125	4.303745
multidisciplinary	4.335897	6.804410	4.622409	7.704977	4.660558	5.535434	5.245810	7.237012	5.012484	7.237615



Median

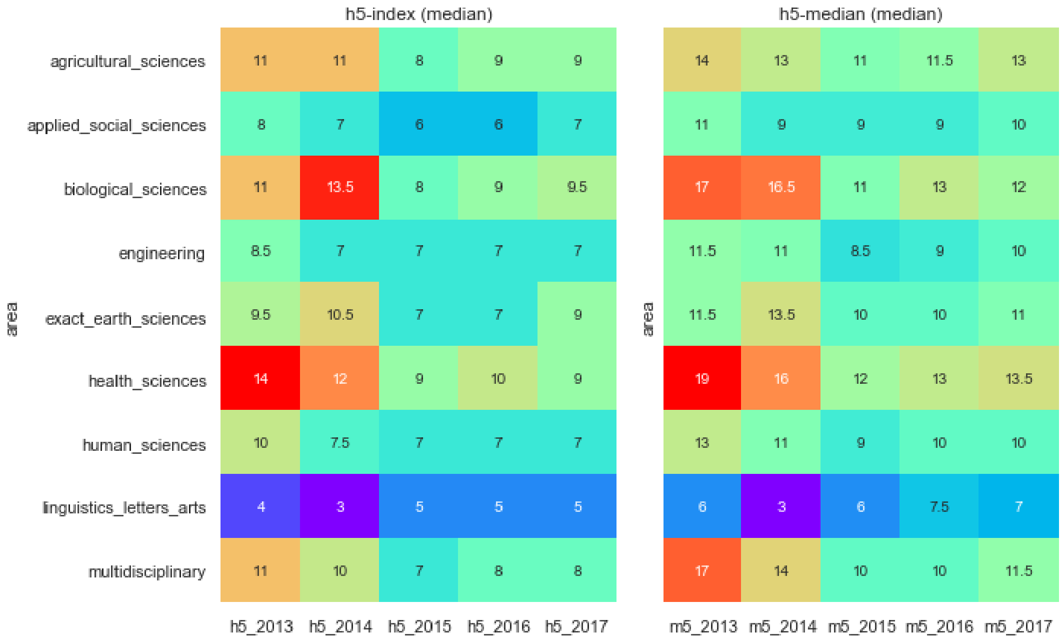
Central tendency (robust)

In [33]:
```
dataset_medians = ddata.loc["50%"].set_index("area")
hmap(dataset_medians, "median")
dataset_medians
```

Out [33]:

The table is in the next page ...

area	h5_2013	m5_2013	h5_2014	m5_2014	h5_2015	m5_2015	h5_2016	m5_2016	h5_2017	m5_2017
agricultural_sciences	11.0	14.0	11.0	13.0	8.0	11.0	9.0	11.5	9.0	13.0
applied_social_sciences	8.0	11.0	7.0	9.0	6.0	9.0	6.0	9.0	7.0	10.0
biological_sciences	11.0	17.0	13.5	16.5	8.0	11.0	9.0	13.0	9.5	12.0
engineering	8.5	11.5	7.0	11.0	7.0	8.5	7.0	9.0	7.0	10.0
exact_earth_sciences	9.5	11.5	10.5	13.5	7.0	10.0	7.0	10.0	9.0	11.0
health_sciences	14.0	19.0	12.0	16.0	9.0	12.0	10.0	13.0	9.0	13.5
human_sciences	10.0	13.0	7.5	11.0	7.0	9.0	7.0	10.0	7.0	10.0
linguistics_letters_arts	4.0	6.0	3.0	3.0	5.0	6.0	5.0	7.5	5.0	7.0
multidisciplinary	11.0	17.0	10.0	14.0	7.0	10.0	8.0	10.0	8.0	11.5



IQR (Inter-Quartile Range)

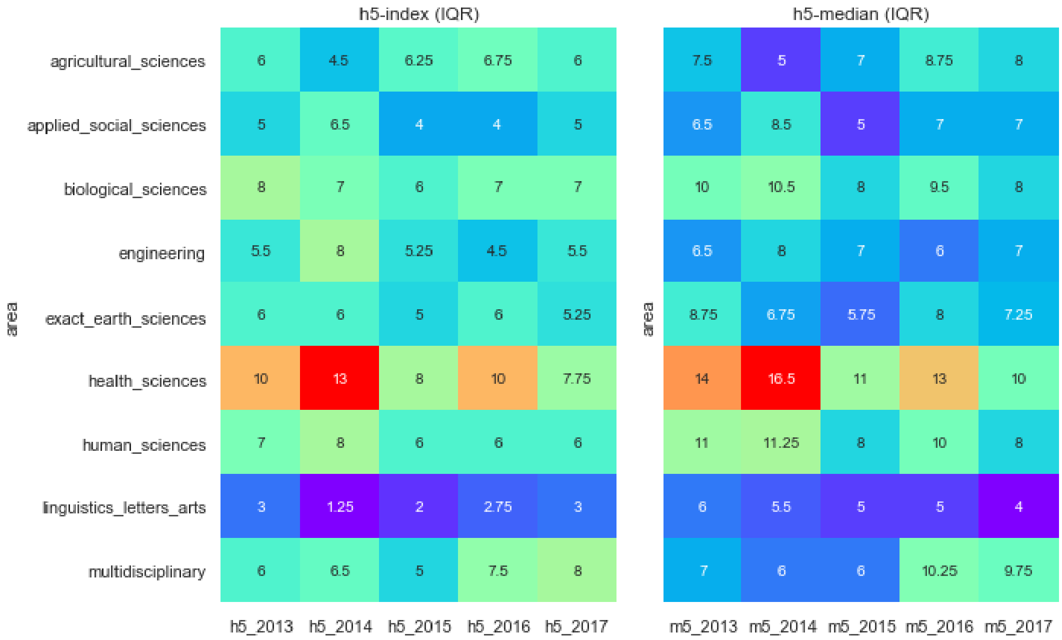
Variability (robust)

In [34]:
```
dataset_iqrs = ddata.loc["75%"].set_index("area") - \
               ddata.loc["25%"].set_index("area")
hmap(dataset_iqrs, "IQR")
dataset_iqrs
```

Out [34]:

The table is in the next page ...

area	h5_2013	m5_2013	h5_2014	m5_2014	h5_2015	m5_2015	h5_2016	m5_2016	h5_2017	m5_2017
agricultural_sciences	6.0	7.50	4.50	5.00	6.25	7.00	6.75	8.75	6.00	8.00
applied_social_sciences	5.0	6.50	6.50	8.50	4.00	5.00	4.00	7.00	5.00	7.00
biological_sciences	8.0	10.00	7.00	10.50	6.00	8.00	7.00	9.50	7.00	8.00
engineering	5.5	6.50	8.00	8.00	5.25	7.00	4.50	6.00	5.50	7.00
exact_earth_sciences	6.0	8.75	6.00	6.75	5.00	5.75	6.00	8.00	5.25	7.25
health_sciences	10.0	14.00	13.00	16.50	8.00	11.00	10.00	13.00	7.75	10.00
human_sciences	7.0	11.00	8.00	11.25	6.00	8.00	6.00	10.00	6.00	8.00
linguistics_letters_arts	3.0	6.00	1.25	5.50	2.00	5.00	2.75	5.00	3.00	4.00
multidisciplinary	6.0	7.00	6.50	6.00	5.00	6.00	7.50	10.25	8.00	9.75



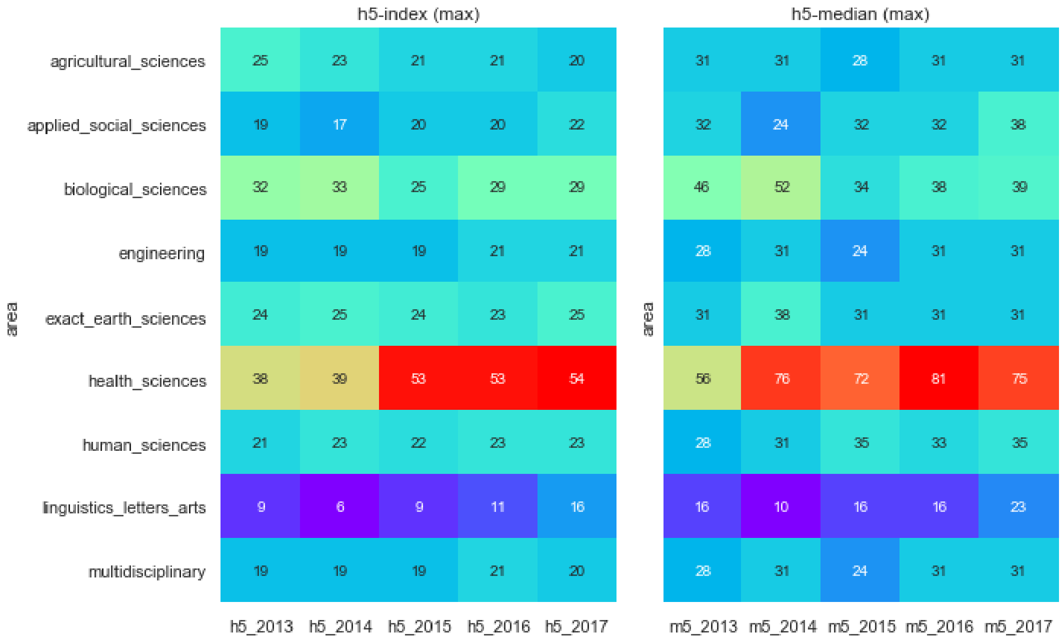
Maximum

In [35]:
```
dataset_maxs = ddata.loc["max"].set_index("area")
hmap(dataset_maxs, "max")
dataset_maxs.astype(int)
```

Out [35]:

The table is in the next page ...

area	h5_2013	m5_2013	h5_2014	m5_2014	h5_2015	m5_2015	h5_2016	m5_2016	h5_2017	m5_2017
agricultural_sciences	25	31	23	31	21	28	21	31	20	31
applied_social_sciences	19	32	17	24	20	32	20	32	22	38
biological_sciences	32	46	33	52	25	34	29	38	29	39
engineering	19	28	19	31	19	24	21	31	21	31
exact_earth_sciences	24	31	25	38	24	31	23	31	25	31
health_sciences	38	56	39	76	53	72	53	81	54	75
human_sciences	21	28	23	31	22	35	23	33	23	35
linguistics_letters_arts	9	16	6	10	9	16	11	16	16	23
multidisciplinary	19	28	19	31	19	24	21	31	20	31



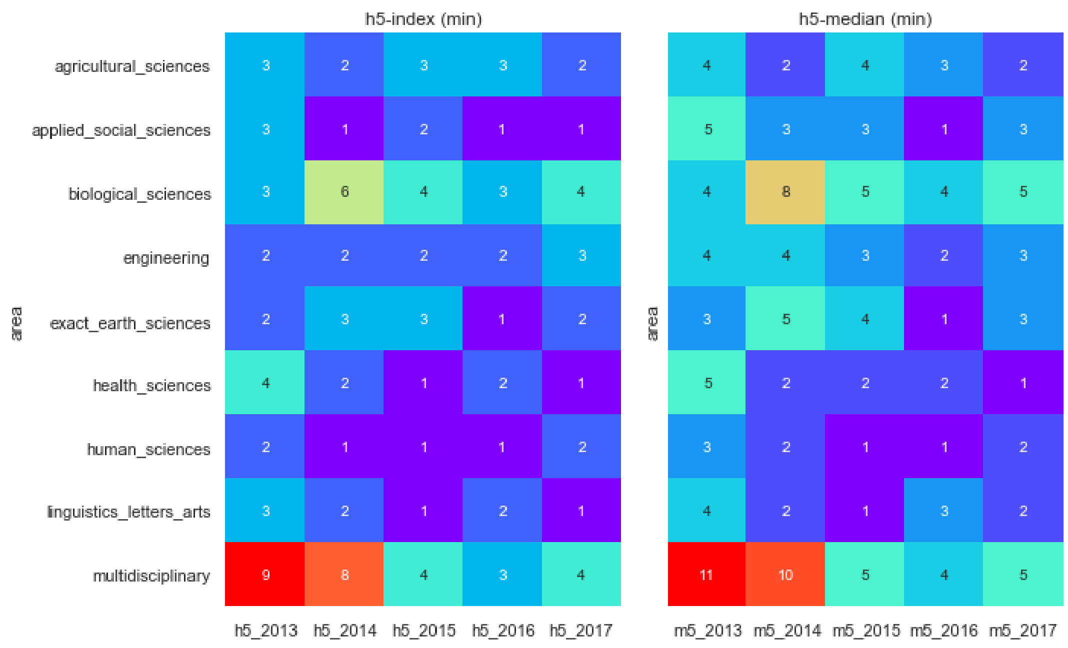
Minimum

In [36]:
```
dataset_mins = ddata.loc["min"].set_index("area")
hmap(dataset_mins, "min")
dataset_mins.astype(int)
```

Out [36]:

The table is in the next page ...

area	h5_2013	m5_2013	h5_2014	m5_2014	h5_2015	m5_2015	h5_2016	m5_2016	h5_2017	m5_2017
agricultural_sciences	3	4	2	2	3	4	3	3	2	2
applied_social_sciences	3	5	1	3	2	3	1	1	1	3
biological_sciences	3	4	6	8	4	5	3	4	4	5
engineering	2	4	2	4	2	3	2	2	3	3
exact_earth_sciences	2	3	3	5	3	4	1	1	2	3
health_sciences	4	5	2	2	1	2	2	2	1	1
human_sciences	2	3	1	2	1	1	1	1	2	2
linguistics_letters_arts	3	4	2	2	1	1	2	3	1	2
multidisciplinary	9	11	8	10	4	5	3	4	4	5



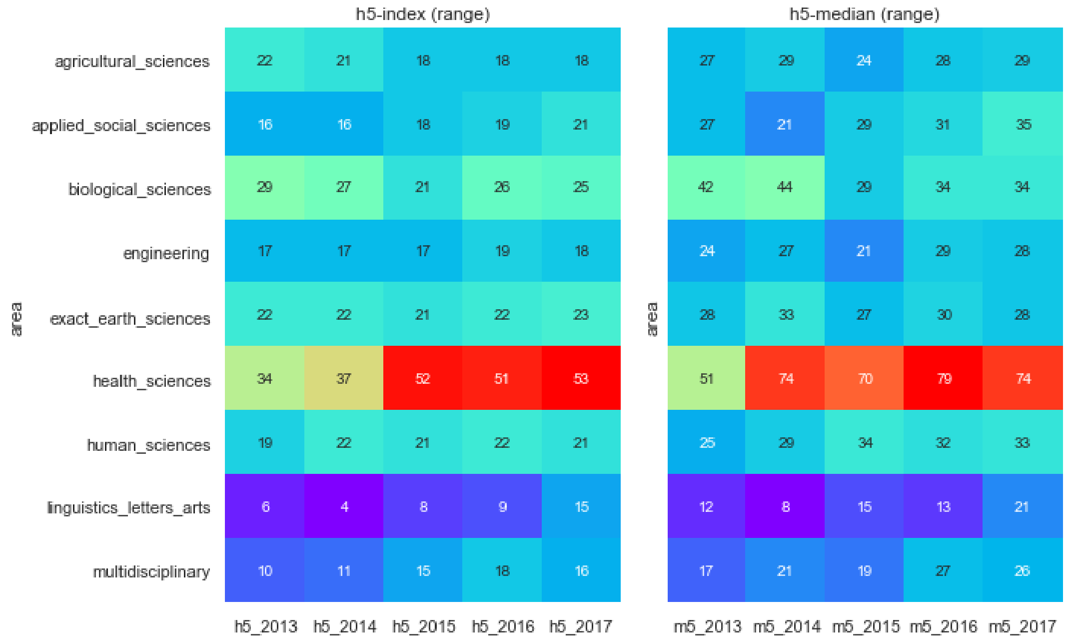
Range

In [37]:
```
dataset_ranges = ddata.loc["max"].set_index("area") - \
                 ddata.loc["min"].set_index("area")
hmap(dataset_ranges, "range")
dataset_ranges.astype(int)
```

Out [37]:

The table is in the next page ...

area	h5_2013	m5_2013	h5_2014	m5_2014	h5_2015	m5_2015	h5_2016	m5_2016	h5_2017	m5_2017
agricultural_sciences	22	27	21	29	18	24	18	28	18	29
applied_social_sciences	16	27	16	21	18	29	19	31	21	35
biological_sciences	29	42	27	44	21	29	26	34	25	34
engineering	17	24	17	27	17	21	19	29	18	28
exact_earth_sciences	22	28	22	33	21	27	22	30	23	28
health_sciences	34	51	37	74	52	70	51	79	53	74
human_sciences	19	25	22	29	21	34	22	32	21	33
linguistics_letters_arts	6	12	4	8	8	15	9	13	15	21
multidisciplinary	10	17	11	21	15	19	18	27	16	26



10.3.3 Summary

It's hard to know how great a number of citations of an article is without knowing its area. The above heat maps gives us some reference on how should we regard the number of citations in some thematic area.

It's clear that the whole dataset have more health sciences and human sciences entries, but just a few multidisciplinary entries. There are a lot of applied social sciences entries too, but most of them don't have the index.

Linguistics, letters and arts seem to get less citations, that might be a characteristic of this thematic area. Maybe the time interval (past 5 years) is too short, or perhaps the whole network of articles in this thematic is smaller, the latter could be justified with the quantity of articles in this dataset, but that's not an information we have.

The variability of indices in health sciences is huge if compared with other areas. Multidisciplinary entries have a small variation.

Biological sciences have a high maximum and minimum, besides the highest *h5-index* for the robust central tendency measurement (median), but this area doesn't have the highest *h5-median*, nor the highest mean of *h5-index*. This suggests it has a more stable *h5-index* than the ones with higher means of *h5-index*.

10.4 Full distributions in 2017

The above *heat maps* might be difficult to understand, mainly the ones regarding dispersion/variability. They have too much information scattered in distinct plots, which are mostly useful to understand the evolution of these indexes from Google Scholar, yet the older columns (2013 to 2016) isn't that much representative of the whole. For now, let's stick with the 2017 data, and plot let's plot the distributions of these indices for both the entire SciELO network and the distinct thematic areas.

The full data with a single-area column repeating an ISSN row for each area it's assigned to is:

In [38]:
```
fdata = pd.concat([
    dataset[dataset[area] == 1][gs_fields].assign(area=area[3:])
```

```
    for area in areaswm
])
fdata.iloc[11::500]
```

Out [38]:

The table is in the next page ...

issn_scielo	h5_2013	m5_2013	h5_2014	m5_2014	h5_2015	m5_2015	h5_2016	m5_2016	h5_2017	m5_2017	area
0100-2945	13.0	17.0	8.0	13.0	14.0	19.0	10.0	13.0	9.0	12.0	agricultural_sciences
2145-9444	NaN	NaN	NaN	NaN	NaN	NaN	4.0	5.0	6.0	7.0	applied_social_sciences
0034-7167	26.0	36.0	30.0	36.0	28.0	33.0	27.0	33.0	29.0	37.0	health_sciences
0104-4036	3.0	5.0	3.0	4.0	9.0	12.0	13.0	17.0	13.0	17.0	human_sciences
0717-3458	NaN	NaN	NaN	NaN	17.0	20.0	18.0	22.0	18.0	23.0	multidisciplinary

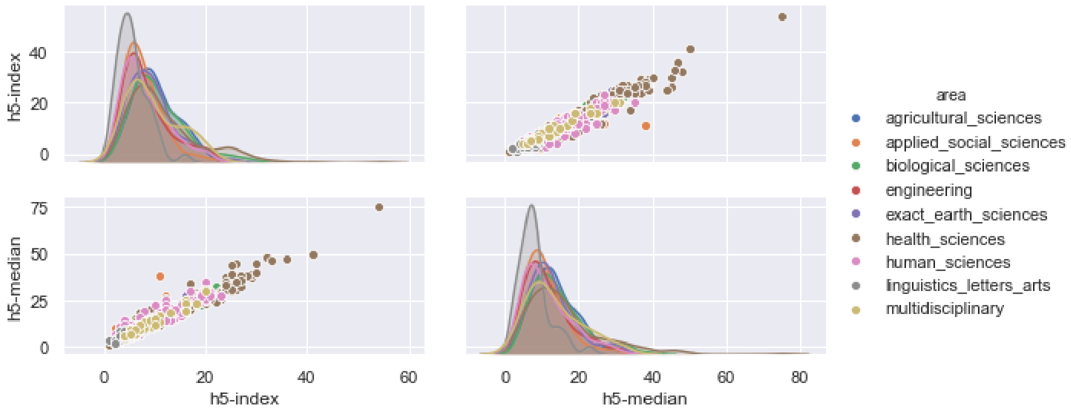
From which we can get just the 2017 data:

In [39]:
```
fdata2017 = fdata[["h5_2017", "m5_2017", "area"]].dropna().rename(columns={
    "h5_2017": "h5-index",
    "m5_2017": "h5-median",
})
fdata2017.head()
```

Out [39]:

	h5-index	h5-median	area
issn_scielo			
0001-3765	16.0	19.0	agricultural_sciences
0006-8705	12.0	15.0	agricultural_sciences
0030-2465	15.0	18.0	agricultural_sciences
0034-737X	13.0	16.0	agricultural_sciences
0038-2353	20.0	31.0	agricultural_sciences

In [40]:
```
sns.pairplot(fdata2017, hue="area", height=2, aspect=2);
```



Most KDEs (kernel density estimates) seem alike, yet there's a single huge index.

Seaborn's FacetGrid can plot all the distribution histograms and KDEs. Forcing the bin step size equal to 1, it'll be a bar plot counting the frequency of each index value. However, it requires a stacked representation of the same data:

In [41]:
```
fdata2017_stack = (fdata2017
    .set_index("area", append=True)
    .stack()
    .reset_index()
    .rename(columns={"level_2": "type", 0: "value"})
    .set_index("issn_scielo")
)
fdata2017_stack.head()
```

Out [41]:

	area	type	value
issn_scielo			
0001-3765	agricultural_sciences	h5-index	16.0
0001-3765	agricultural_sciences	h5-median	19.0
0006-8705	agricultural_sciences	h5-index	12.0
0006-8705	agricultural_sciences	h5-median	15.0
0030-2465	agricultural_sciences	h5-index	15.0

That we can plot with a `sns.FacetGrid` (be careful when interpreting, the titles are **above**, not below!):

In [42]:
```
sns.FacetGrid(fdata2017_stack, row="area", col="type", aspect=2.7, height=2) \
   .map(sns.distplot, "value",
        bins=np.arange(fdata2017_stack["value"].max()),
        kde=False);
```


area = agricultural_sciences | type = h5-index
area = agricultural_sciences | type = h5-median
area = applied_social_sciences | type = h5-index
area = applied_social_sciences | type = h5-median
area = biological_sciences | type = h5-index
area = biological_sciences | type = h5-median
area = engineering | type = h5-index
area = engineering | type = h5-median
area = exact_earth_sciences | type = h5-index
area = exact_earth_sciences | type = h5-median
area = health_sciences | type = h5-index
area = health_sciences | type = h5-median
area = human_sciences | type = h5-index
area = human_sciences | type = h5-median
area = linguistics_letters_arts | type = h5-index
area = linguistics_letters_arts | type = h5-median
area = multidisciplinary | type = h5-index
area = multidisciplinary | type = h5-median
0
10
20
30
0 10 20 30 40 50 60 70
value

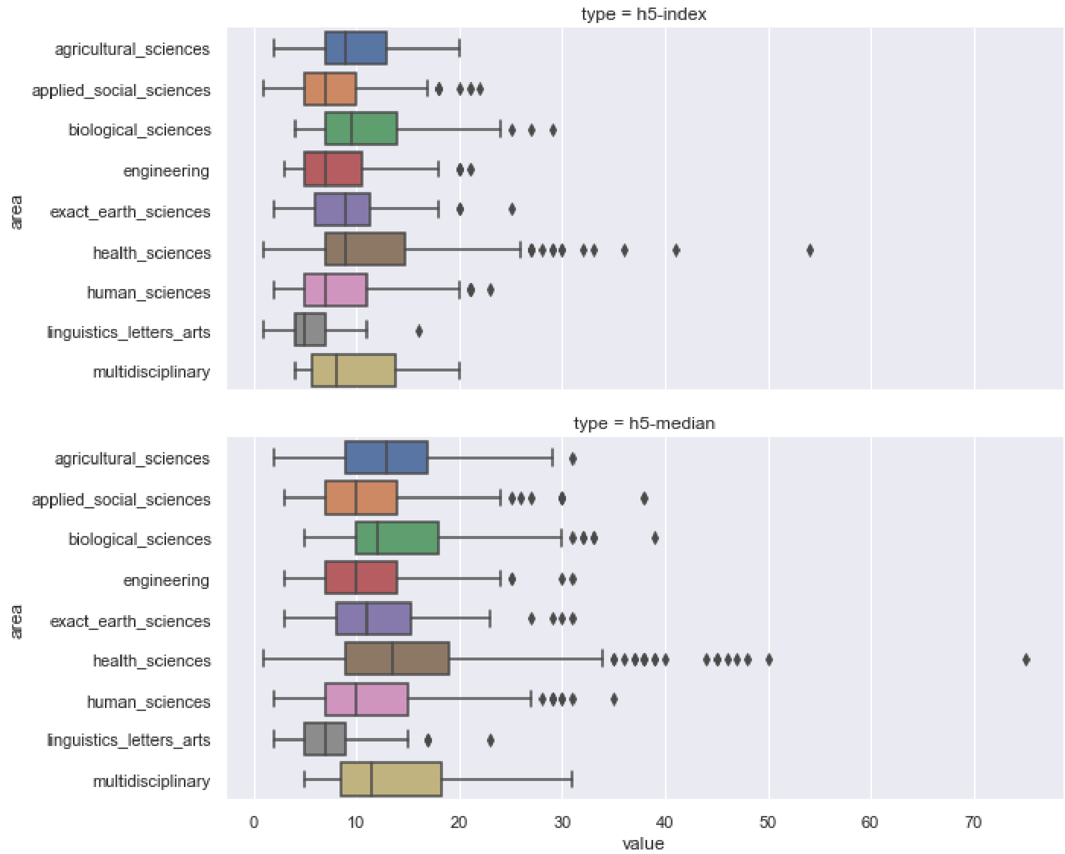
The same, normalized as a probability function and with its KDE:

In [43]:
```
sns.FacetGrid(fdata2017_stack, row="area", col="type", aspect=2.7, height=2) \
    .map(sns.distplot, "value", bins=np.arange(fdata2017_stack["value"].max()));
```


area = agricultural_sciences | type = h5-index
area = agricultural_sciences | type = h5-median
area = applied_social_sciences | type = h5-index
area = applied_social_sciences | type = h5-median
area = biological_sciences | type = h5-index
area = biological_sciences | type = h5-median
area = engineering | type = h5-index
area = engineering | type = h5-median
area = exact_earth_sciences | type = h5-index
area = exact_earth_sciences | type = h5-median
area = health_sciences | type = h5-index
area = health_sciences | type = h5-median
area = human_sciences | type = h5-index
area = human_sciences | type = h5-median
area = linguistics_letters_arts | type = h5-index
area = linguistics_letters_arts | type = h5-median
area = multidisciplinary | type = h5-index
area = multidisciplinary | type = h5-median
0.00
0.05
0.10
0.15
0
20
40
60
80
value

There are a lot of information there, but they're hard to compare. Perhaps a simple boxplot of it all would be simpler.

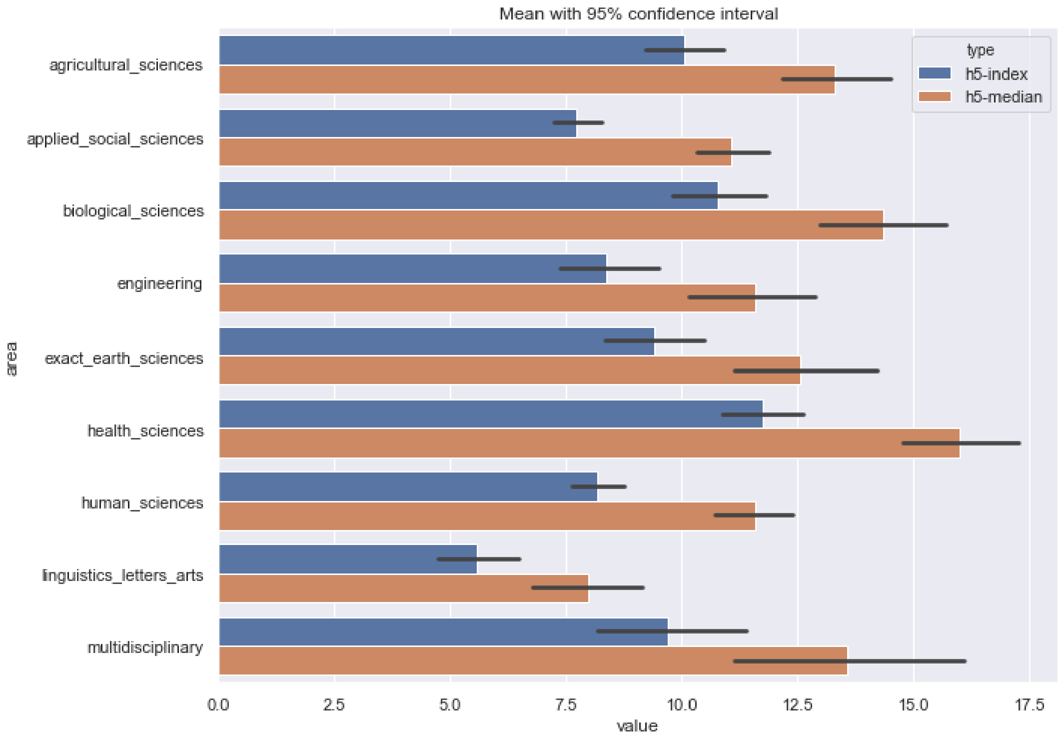
In [44]:
```
sns.catplot(kind="box", aspect=2.5, height=4,
            data=fdata2017_stack, row="type", y="area", x="value");
```



Now a several information is together in a single plot: quartiles, median, IQR, minimum, maximum, outlier thresholds and the outliers. This one is probably the most informative plot so far in this notebook.

We can see the standard statistics in a barplot:

In [45]:
```
sns.barplot(data=fdata2017_stack, y="area", x="value", hue="type",
            ax=plt.subplots(figsize=(10, 8))[1]) \
    .set(title="Mean with 95% confidence interval");
```



Let's perform:

`fdata2017.groupby("area").describe()`

in a stacked table style:

In [46]:
```
fdata2017_descr = (fdata2017
    .groupby("area")
    .describe()
    .stack(0)
    .rename_axis(["area", "type"])
    .reset_index()
)
fdata2017_descr
```

Out [46]:

	area	type	25%	50%	75%	count	max	mean	min	std
0	agricultural_sciences	h5-index	7.00	9.0	13.00	92.0	20.0	10.043478	2.0	4.185042
1	agricultural_sciences	h5-median	9.00	13.0	17.00	92.0	31.0	13.304348	2.0	5.704710
2	applied_social_sciences	h5-index	5.00	7.0	10.00	192.0	22.0	7.734375	1.0	3.774711
3	applied_social_sciences	h5-median	7.00	10.0	14.00	192.0	38.0	11.052083	3.0	5.720121
4	biological_sciences	h5-index	7.00	9.5	14.00	102.0	29.0	10.784314	4.0	5.285188
5	biological_sciences	h5-median	10.00	12.0	18.00	102.0	39.0	14.333333	5.0	7.188182
6	engineering	h5-index	5.00	7.0	10.50	71.0	21.0	8.366197	3.0	4.645242
7	engineering	h5-median	7.00	10.0	14.00	71.0	31.0	11.591549	3.0	6.370866

Continued on next page

	area	type	25%	50%	75%	count	max	mean	min	std
8	exact_earth_sciences	h5-index	6.00	9.0	11.25	64.0	25.0	9.406250	2.0	4.655493
9	exact_earth_sciences	h5-median	8.00	11.0	15.25	64.0	31.0	12.562500	3.0	6.334273
10	health_sciences	h5-index	7.00	9.0	14.75	286.0	54.0	11.741259	1.0	7.460035
11	health_sciences	h5-median	9.00	13.5	19.00	286.0	75.0	16.000000	1.0	10.201479
12	human_sciences	h5-index	5.00	7.0	11.00	272.0	23.0	8.180147	2.0	4.593503
13	human_sciences	h5-median	7.00	10.0	15.00	272.0	35.0	11.591912	2.0	6.621474
14	linguistics_letters_arts	h5-index	4.00	5.0	7.00	45.0	16.0	5.577778	1.0	2.996125
15	linguistics_letters_arts	h5-median	5.00	7.0	9.00	45.0	23.0	7.977778	2.0	4.303745
16	multidisciplinary	h5-index	5.75	8.0	13.75	32.0	20.0	9.687500	4.0	5.012484
17	multidisciplinary	h5-median	8.50	11.5	18.25	32.0	31.0	13.562500	5.0	7.237615

The information that the above plots misses is the total/overall count (already seen in a heat map, but easier to visualize in a bar plot).

In [47]:
```
sns.barplot(data=fdata2017_descr, y="area", x="count",
            ax=plt.subplots(figsize=(10, 8))[1]) \
   .set(title="Count");
```