TEDxCasey

May 16, 2019

Nine Hall

https://ninethehacker.neocities.org

Google developed a web app called ngram viewer, an app that allows the rapid creation of histograms tracking the useage of phrases over time. This allows us to get something approximating an objective measure of innovation, with some caveats:

- the google dataset becomes unreliable past 2000, so we lose the last 20 years
- the dataset is printed media so terms popularised on the internet arene't plotted
- terms that acquired their meaning from a popular term or were ideologically invisible cannot be plotted.

The first thing I did was create a dataset composed of lists of keywords/terms. This is a manual process done entirely by hand. I was hoping to revise this with supplemental data from youtube's subtitle api but unfortunately the stream was not uploaded to YouTube.

[3]:
```
danyal_diallo = ["where are you from", "stereotype", "uniquely creative", "own␣
 ↪journey", "my identity", "my best life", "team work makes the dream work",␣
 ↪"visualise what you want", "crazy and unique", "embrace your identity",␣
 ↪"embrace your individuality", "limitless potential"]
evita_march = ["cyberabuse", "just ignore it", "online", "it was everywhere",␣
 ↪"psychological profiling", "cognitive empathy", "affective empathy", "block␣
 ↪the haters", "radical empathy", "compassion and understanding", "love beats␣
 ↪hate"]
dheeren_velu = ["dawning of a new age", "age of augmented intelligence",␣
 ↪"augmented intelligence", "artificial intelligence", "machine learning",␣
 ↪"induction", "innovator", "revolution", "skin in the game"]
lana_johnson = ["wellbeing", "lifestyle diseases", "contagion", "emotional␣
 ↪contagion", "spiritual, belonging", "learn and grow", "make it stick"]
mark_carter = ["value quadrant", "emotional value", "social responsibility",␣
 ↪"wish were true", "human development", "relationship value", "biggest dirty␣
 ↪little secret", "candour", "tangible value", "service value", "kindness",␣
 ↪"love"]
bill_holmes = ["diverse", "interesting", "problem solving", "mistakes",␣
 ↪"strategies", "how we think", "do better"]
vicki_macdermid = ["connection", "power", "horsepower", "own power", "belief",␣
 ↪"intention", "personal power"]
```

```
henry_wu = ["space", "satellite", "science", "STEM", "space tourism", "asteroid␣
 ↪mining", "astronomy", "fresh perspective", "exciting times"]
philipe_guichard = ["innovation", "industrial design", "empowering people",␣
 ↪"empower everyone", "better world", "challenges"]
all_speakers = danyal_diallo + evita_march + dheeren_velu + lana_johnson +␣
 ↪mark_carter + bill_holmes + vicki_macdermid + henry_wu + philipe_guichard
```

1 Extract and transform

The next step is to extract the data from google's app. There is an open API wrapper at https://github.com/econpy/google-ngrams the difficult part is coming up with an ETL system that's relatively clean and gives us quality dataframes that suit our graphing. This is a work in progress: pandas not something you can learn in a month.

[24]:
```
# DON'T HAMMER GOOGLE'S UNDOCUMENTED API
# PROBABLY DON'T NEED TO RUN THIS BLOCK UNLESS THE DATASET CHANGES
import sys
sys.path.insert(0, './google-ngrams-master/')
import getngrams
def get_ngram_csv(list):
    for query in list:
        getngrams.runQuery(query)
```

[7]:
```
import pandas as pd
import matplotlib.pyplot as plt

def build_dataframe(list):
    dataframe = pd.read_csv(''.join(list[0].replace(',', '_').
 ↪split())+'-eng_2012-1800-2000-3-caseSensitive.csv',
                            index_col=0,
                            parse_dates=True)
    for query in list[1:]:
        new_dataframe = pd.read_csv(''.join(query.replace(',', '_').
 ↪split())+'-eng_2012-1800-2000-3-caseSensitive.csv',
                        index_col=0,
                        parse_dates=True)
        dataframe = dataframe.merge(right=new_dataframe, on='year')
    return dataframe

def plot_absolute_graph(title, dataframe, show_legend = True):
    if not show_legend:
        dataframe.plot(title=title+': Absolute values', figsize=(12,6),␣
 ↪legend=None).get_yaxis().set_visible(False)
    else:
        dataframe.plot(title=title+': Absolute values', figsize=(12,6)).
 ↪get_yaxis().set_visible(False)
```

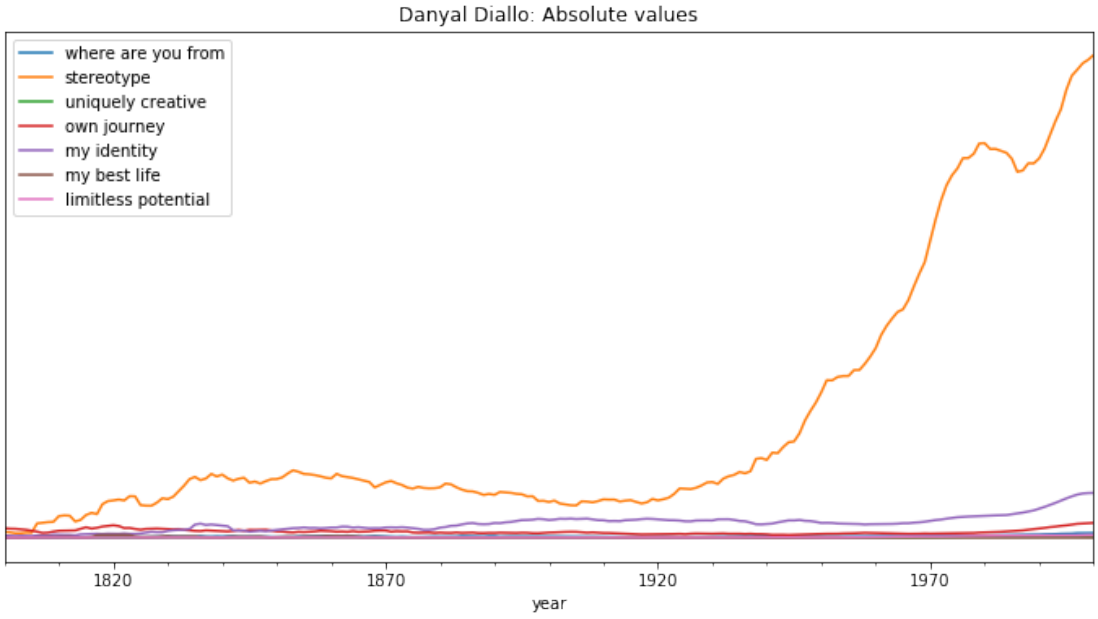
```
def plot_normalised_graph(title, dataframe, show_legend = True):
    temp = dataframe
    for col in dataframe.columns:
        temp[col] = [i/max(temp[col]) for i in temp[col]]
    if not show_legend:
        dataframe.plot(title=title+': Normalised', figsize=(12,6), legend=None).
↪get_yaxis().set_visible(False)
    else:
        dataframe.plot(title=title+': Normalised', figsize=(12,6)).get_yaxis().
↪set_visible(False)
```

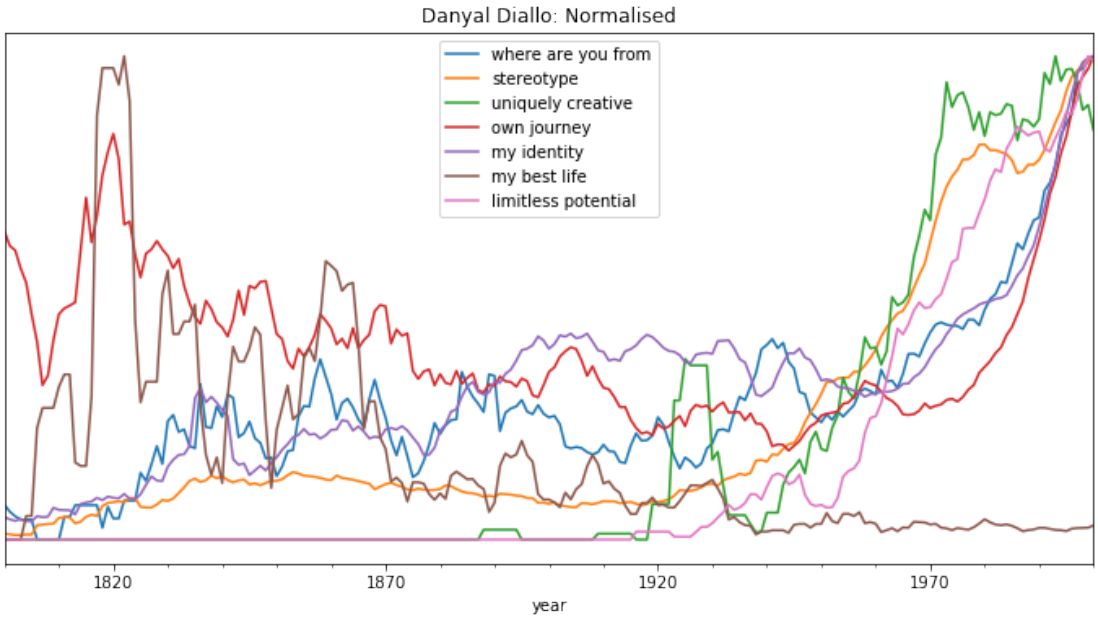
2 Speakers

2.1 Danyal Diallo: Breaking the stereotypes

I love this graph: it's not just the first talk, it's the talk tha sets a clear trendline and one we're going to see persist throughout our dataset.

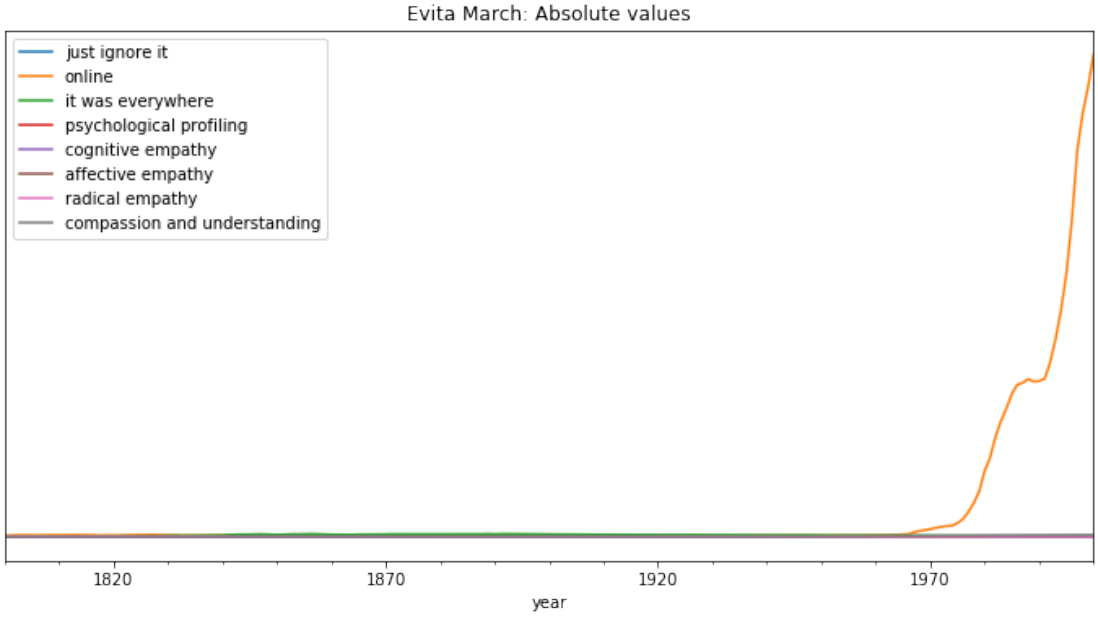
[102]:
```
name="Danyal Diallo"
danyal_diallo_df = build_dataframe(danyal_diallo)
plot_absolute_graph(name, danyal_diallo_df)
plot_normalised_graph(name, danyal_diallo_df)
```

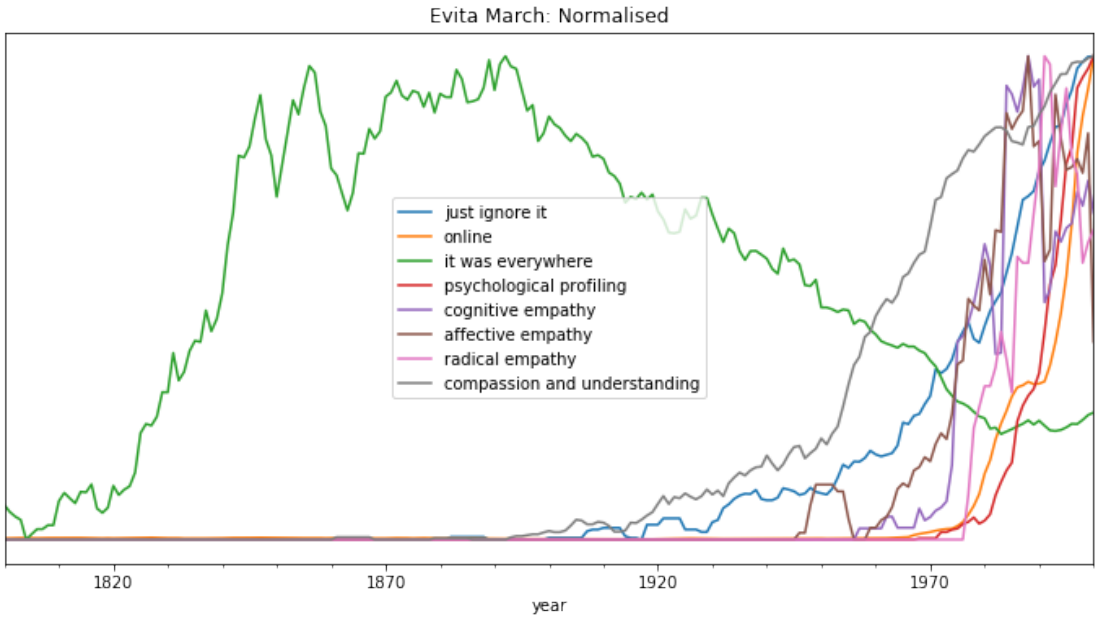




2.2 Evita March

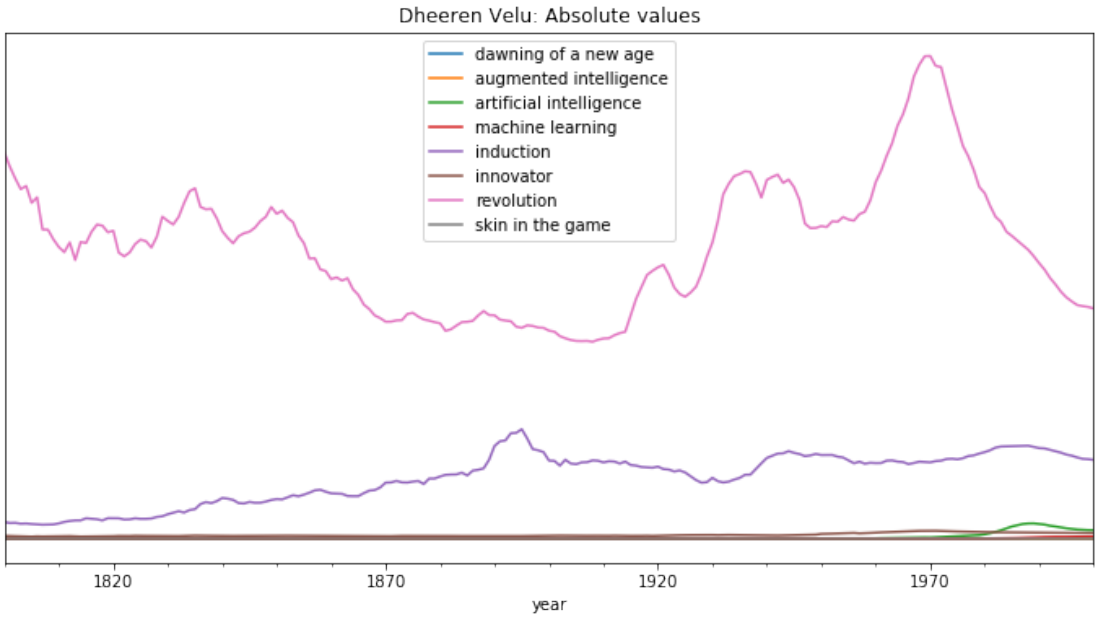
[104]:
```
name="Evita March"
evita_march_df = build_dataframe(evita_march)
plot_absolute_graph(name, evita_march_df)
plot_normalised_graph(name, evita_march_df)
```

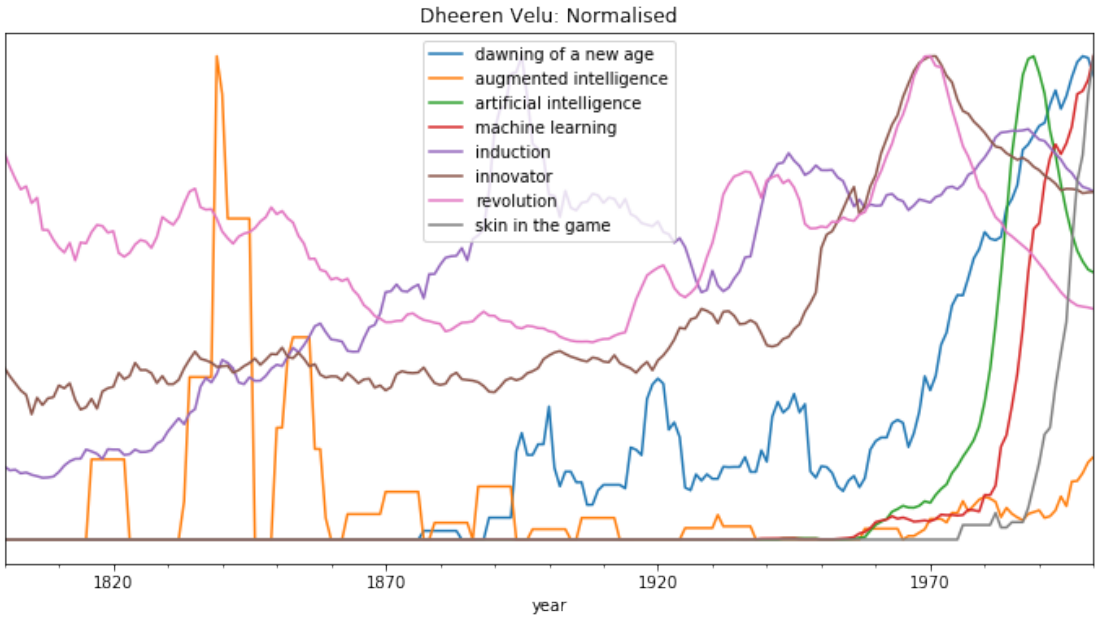




2.3 Dheeren Velu

[106]:
```
name="Dheeren Velu"
dheeren_velu_df = build_dataframe(dheeren_velu)
plot_absolute_graph(name, dheeren_velu_df)
plot_normalised_graph(name, dheeren_velu_df)
```

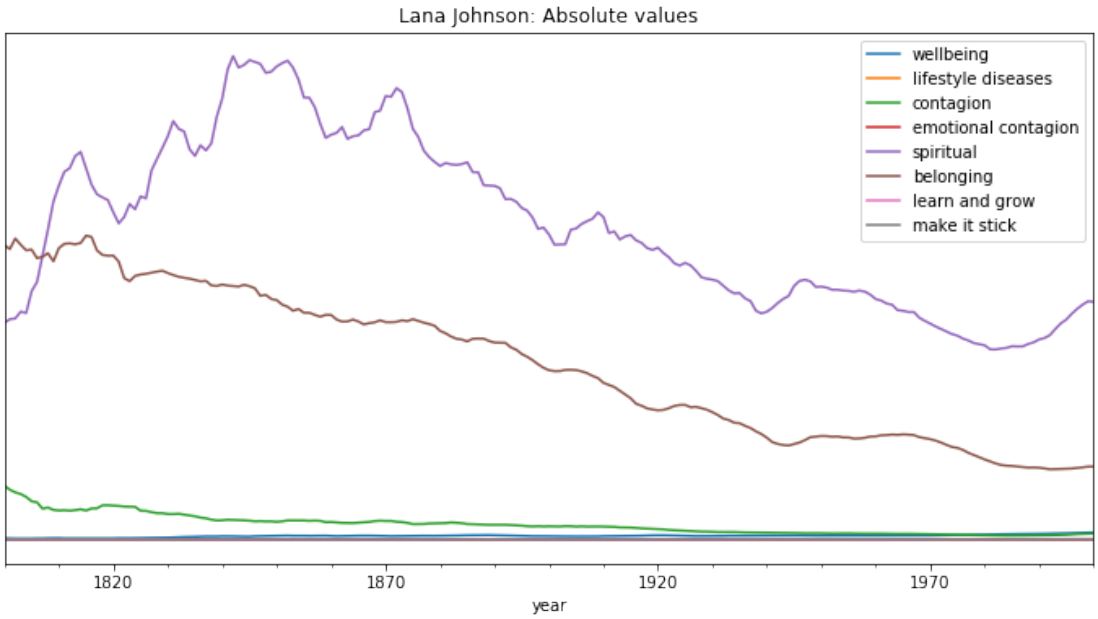


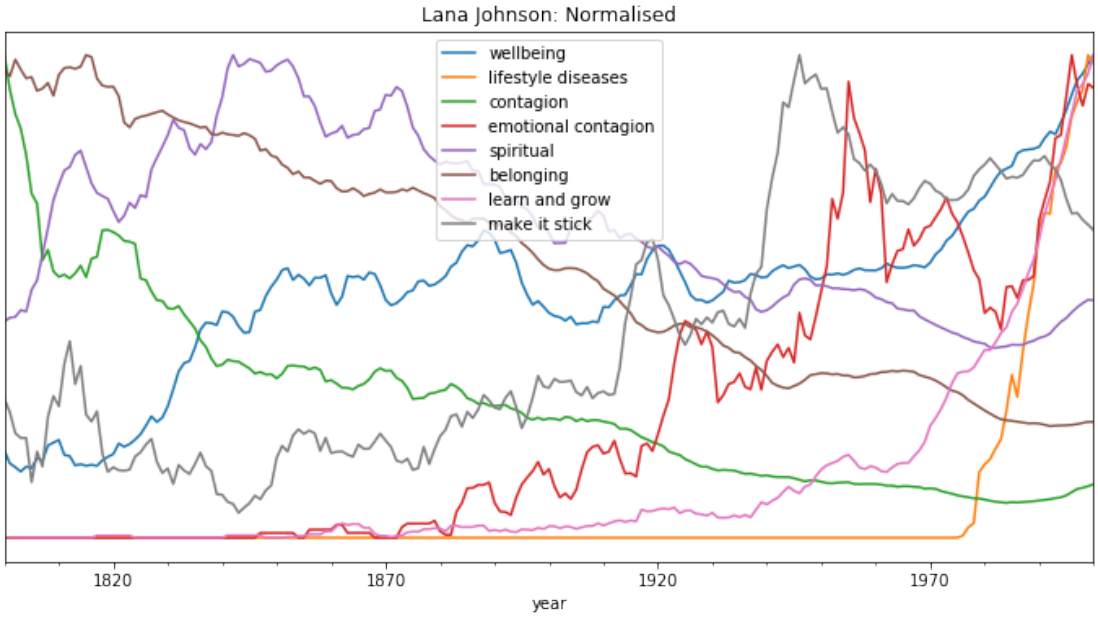


2.4 Lana Johnson: Emotional contagion

This graph ended up going kind of sideways. I guess emotional state is something well defined in the corpus long before 1800. The core terms "lifestyle disease" is relatively new, emerging in the 80s, possibly as a euphamism for living in the 80s.

[110]:
```
name="Lana Johnson"
lana_johnson_df = build_dataframe(lana_johnson)
plot_absolute_graph(name, lana_johnson_df)
plot_normalised_graph(name, lana_johnson_df)
```

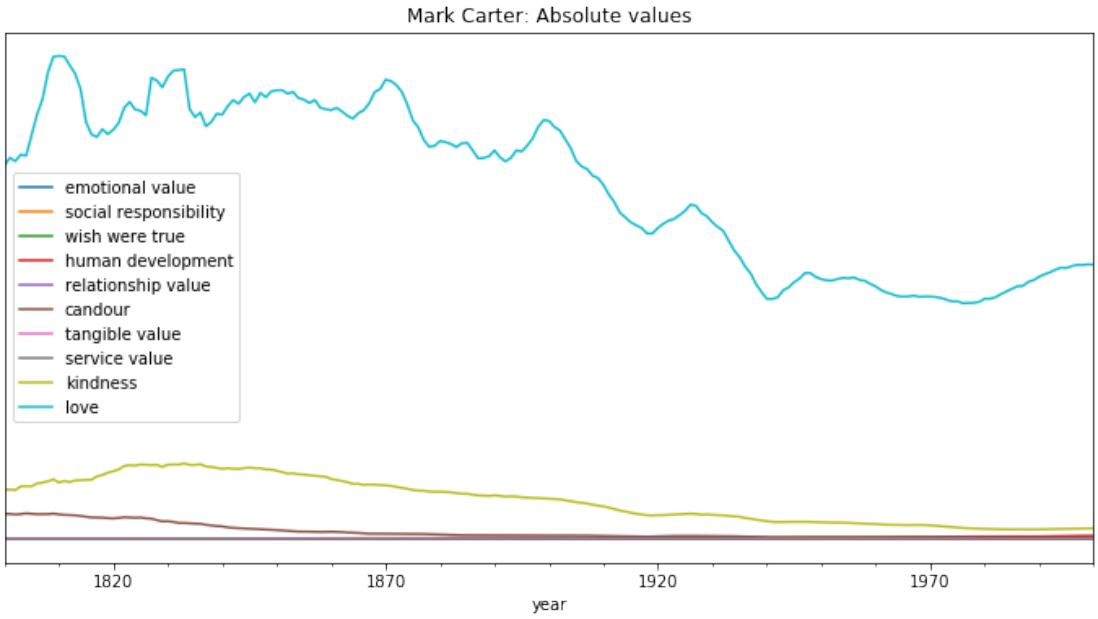


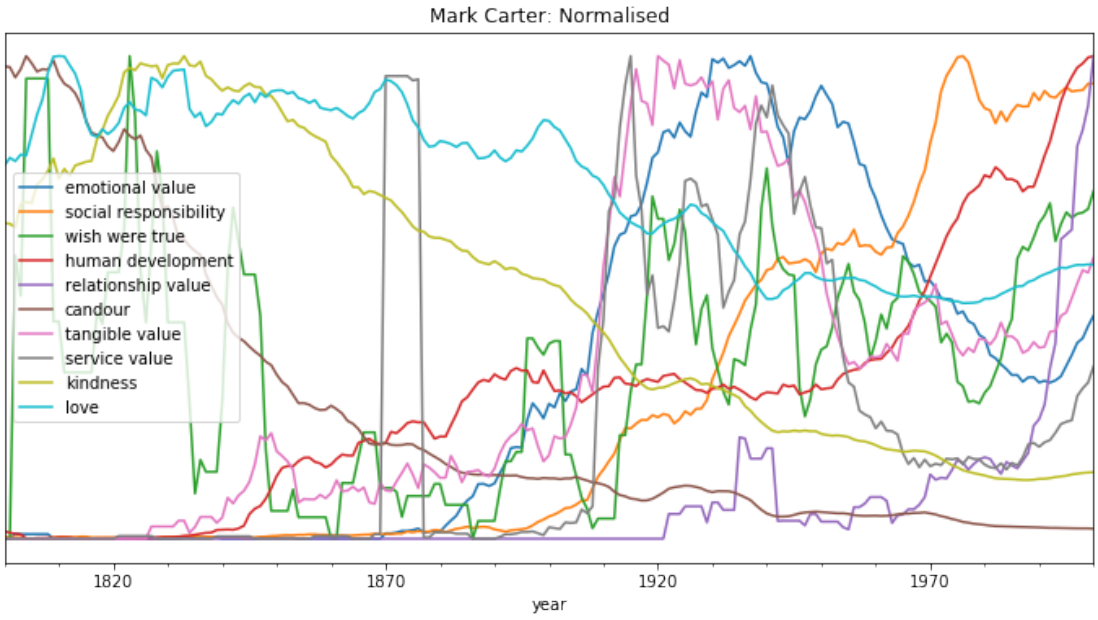


2.5 Mark Carter

[113]:
```
name="Mark Carter"
mark_carter_df = build_dataframe (mark_carter)
plot_absolute_graph(name, mark_carter_df)
```

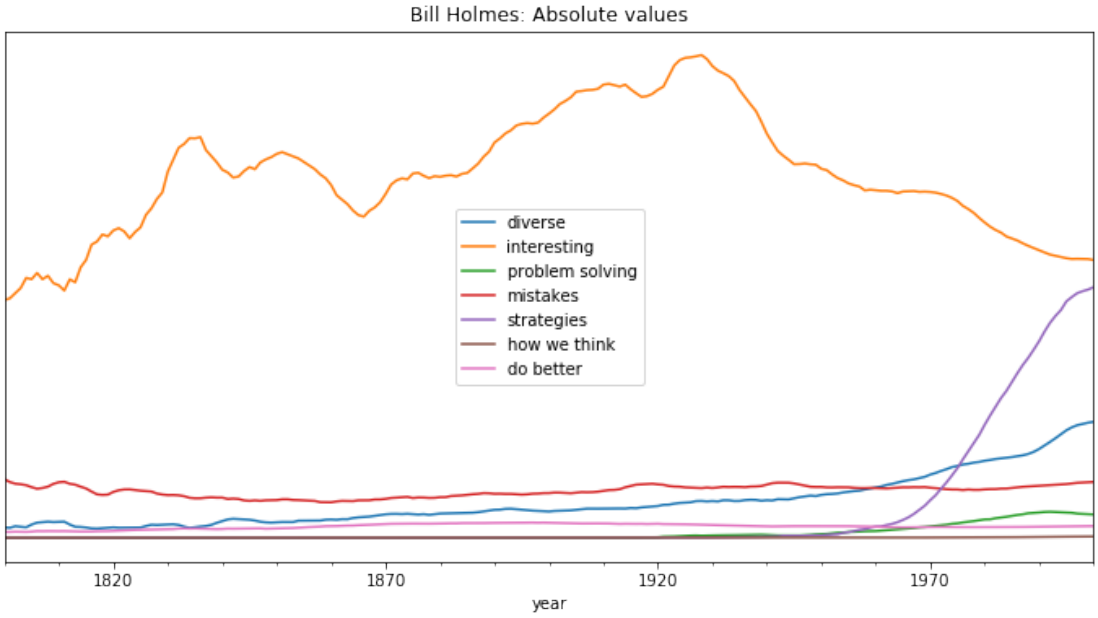
```
plot_normalised_graph(name, mark_carter_df)
```

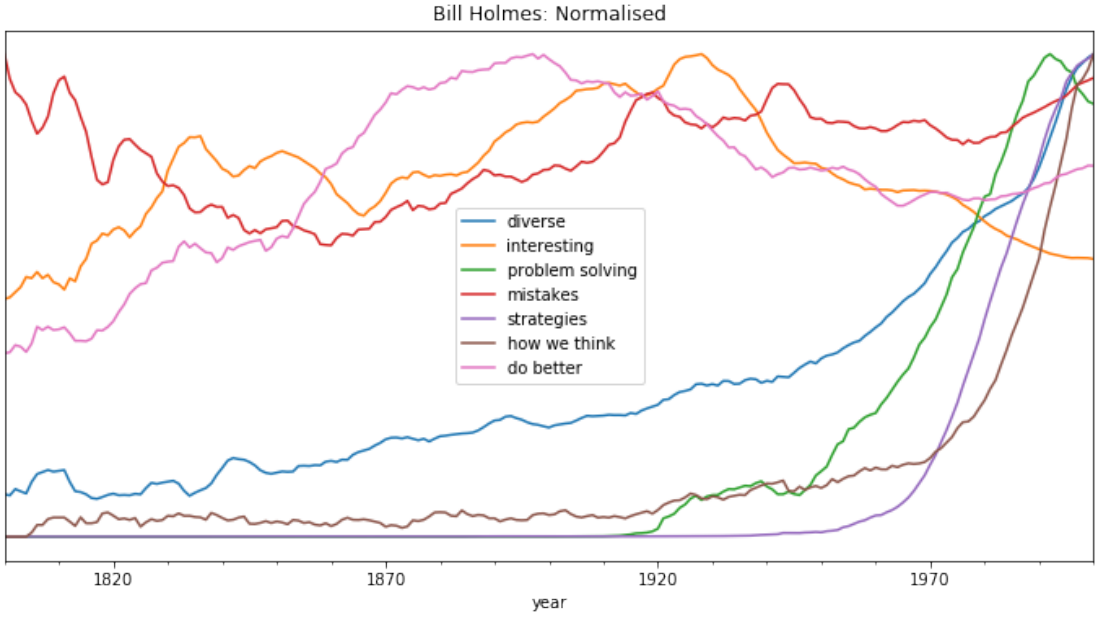




2.6 Bill Holmes: The power of why

[107]:
```
name="Bill Holmes"
bill_holmes_df = build_dataframe(bill_holmes)
plot_absolute_graph(name, bill_holmes_df)
plot_normalised_graph(name, bill_holmes_df)
```

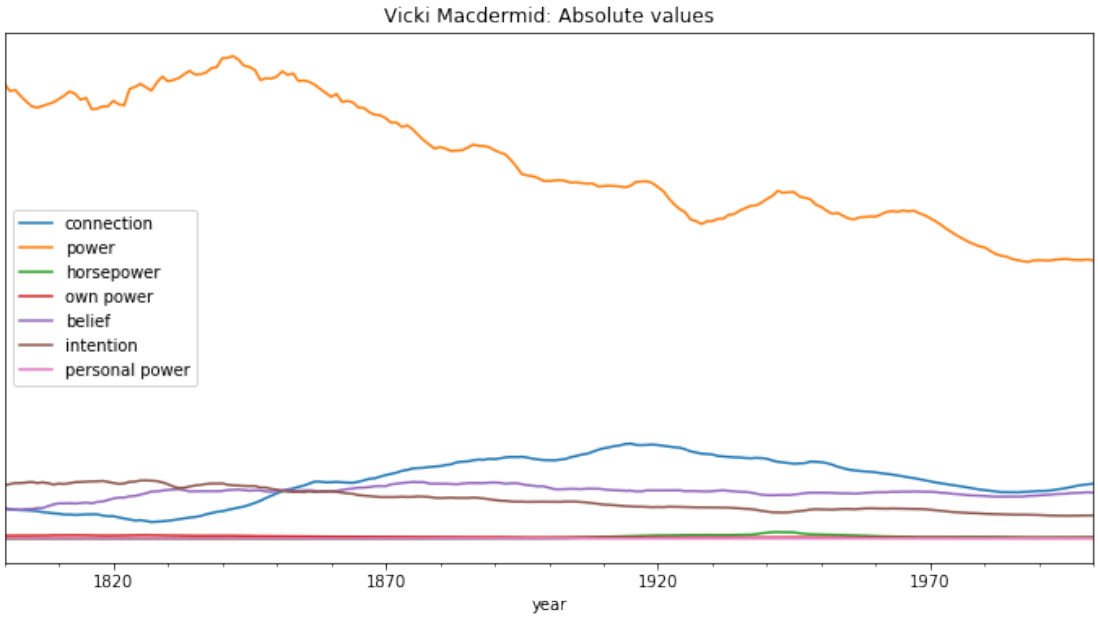


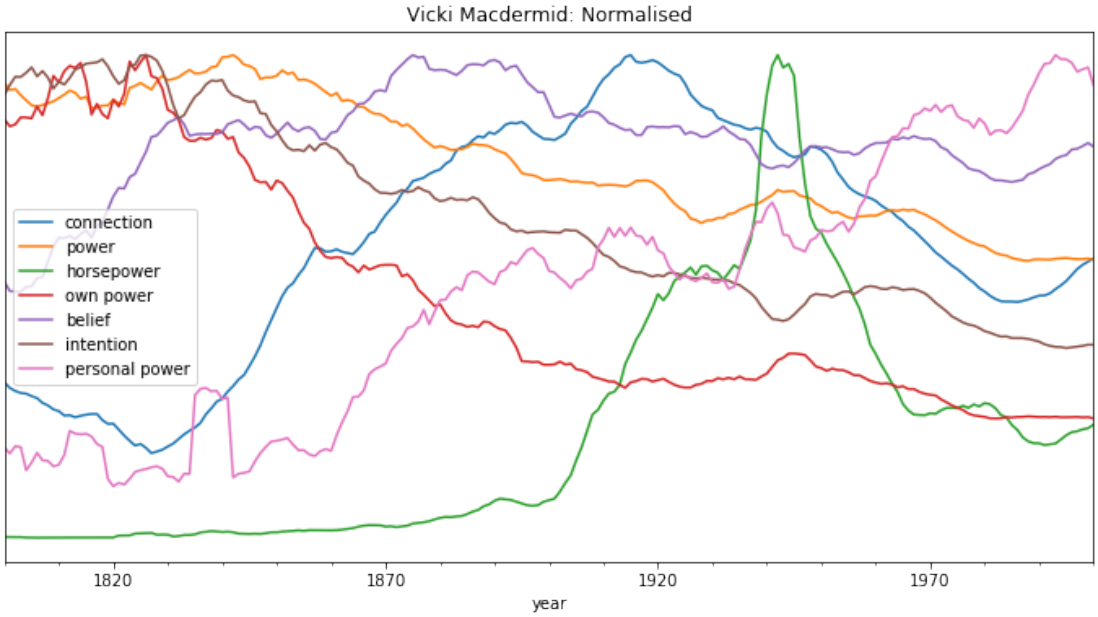


2.7 Vicki Macdermid: The power of connection

Vicki was ironically the least boomer.

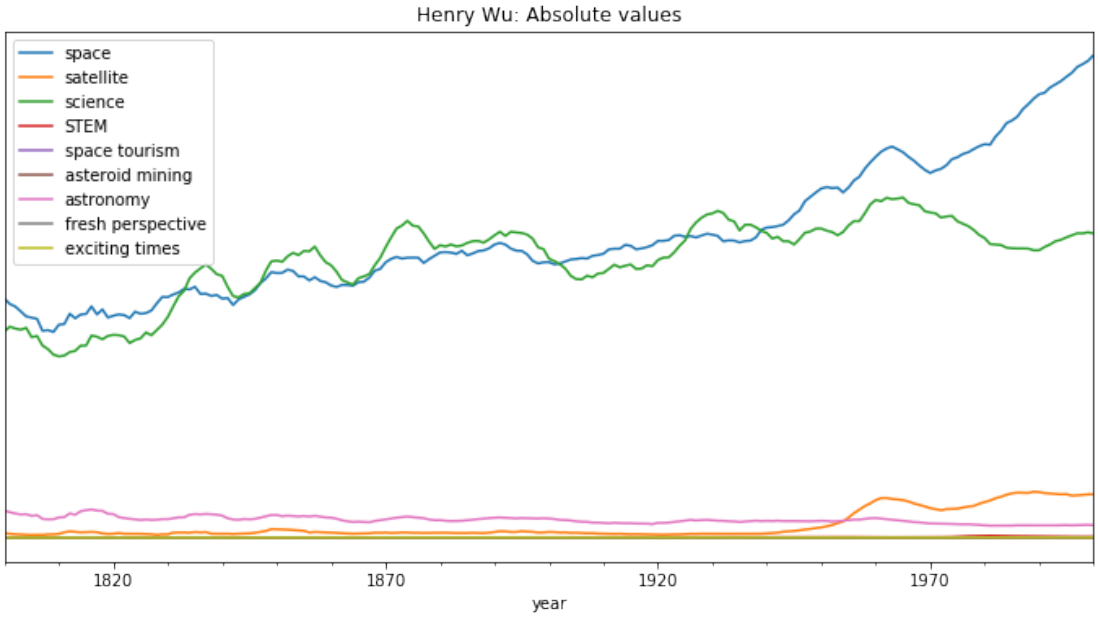
[98]:
```
name="Vicki Macdermid"
vicki_macdermid_df = build_dataframe(vicki_macdermid)
plot_absolute_graph(name, vicki_macdermid_df)
plot_normalised_graph(name, vicki_macdermid_df)
```

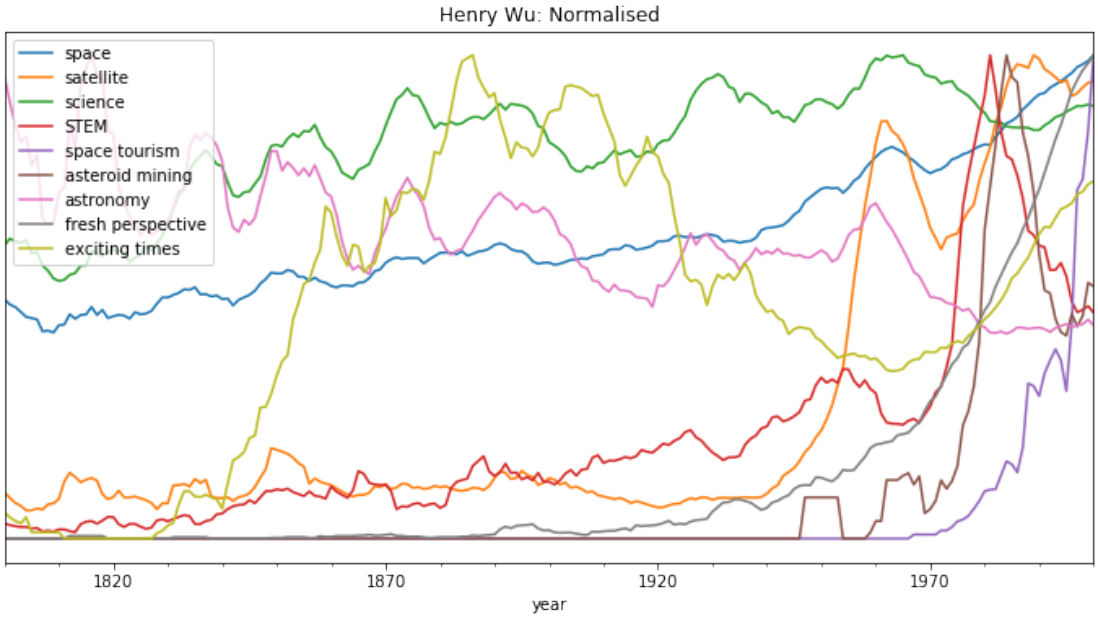




2.8 Henry Wu

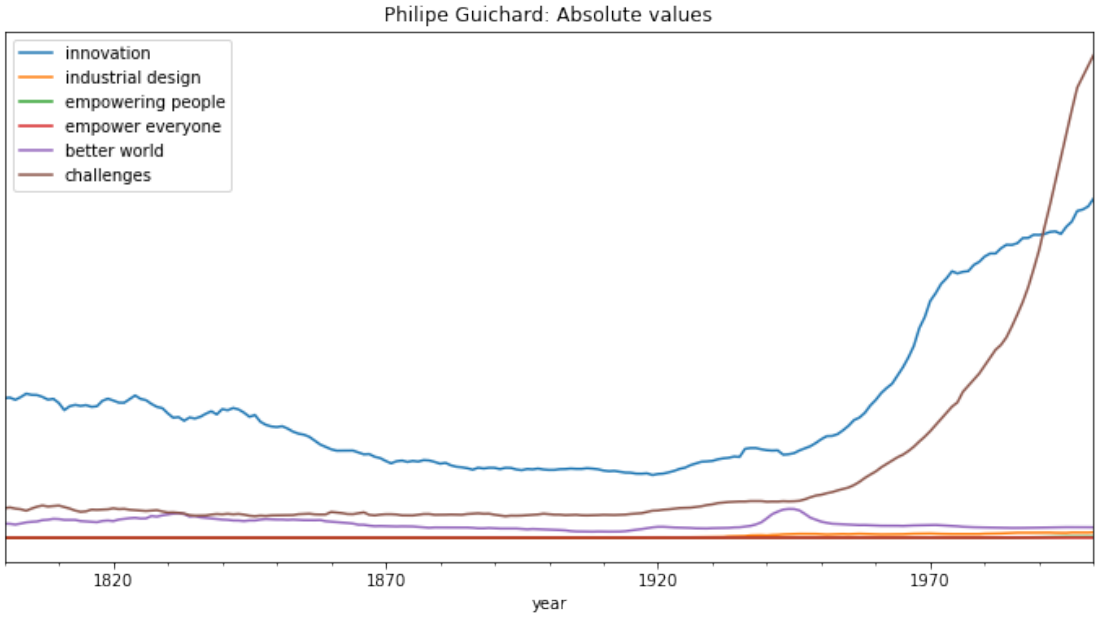
[96]:
```
name="Henry Wu"
henry_wu_df = build_dataframe(henry_wu)
plot_absolute_graph(name, henry_wu_df)
plot_normalised_graph(name, henry_wu_df)
```

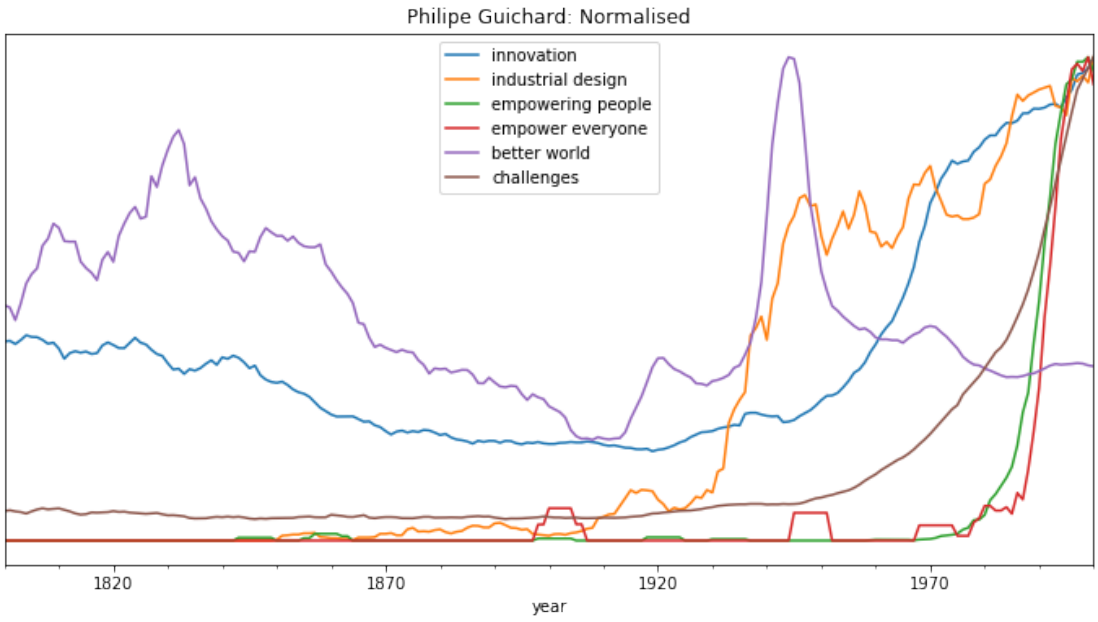




2.9 Philipe Guichard: Re-designing our world

[94]:
```
name="Philipe Guichard"
philipe_guichard_df = build_dataframe(philipe_guichard)
plot_absolute_graph(name, philipe_guichard_df)
plot_normalised_graph(name, philipe_guichard_df)
```

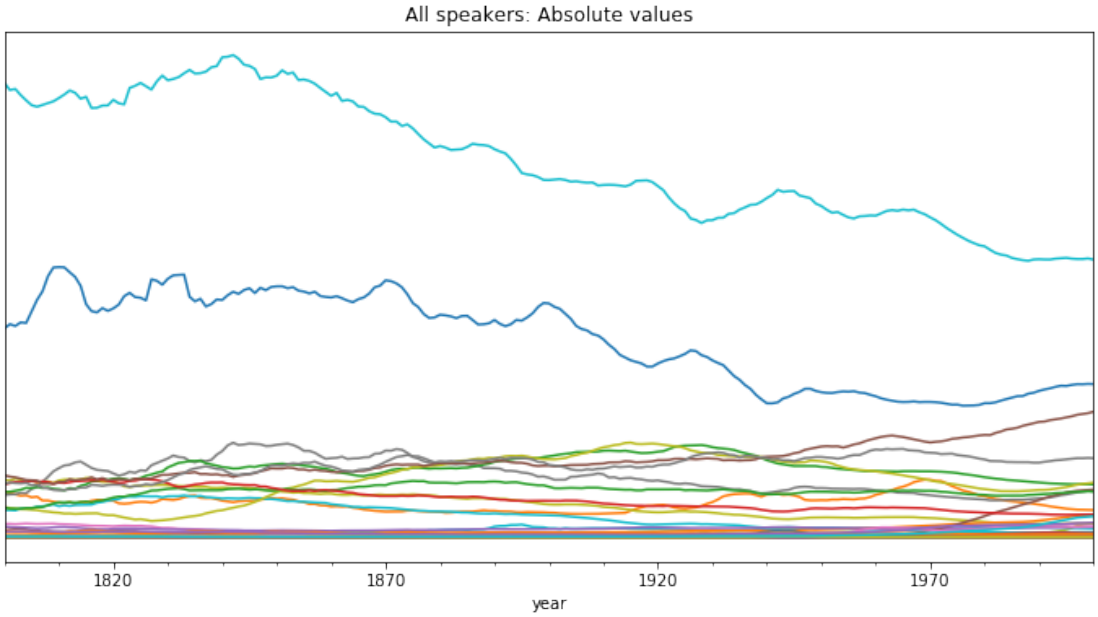


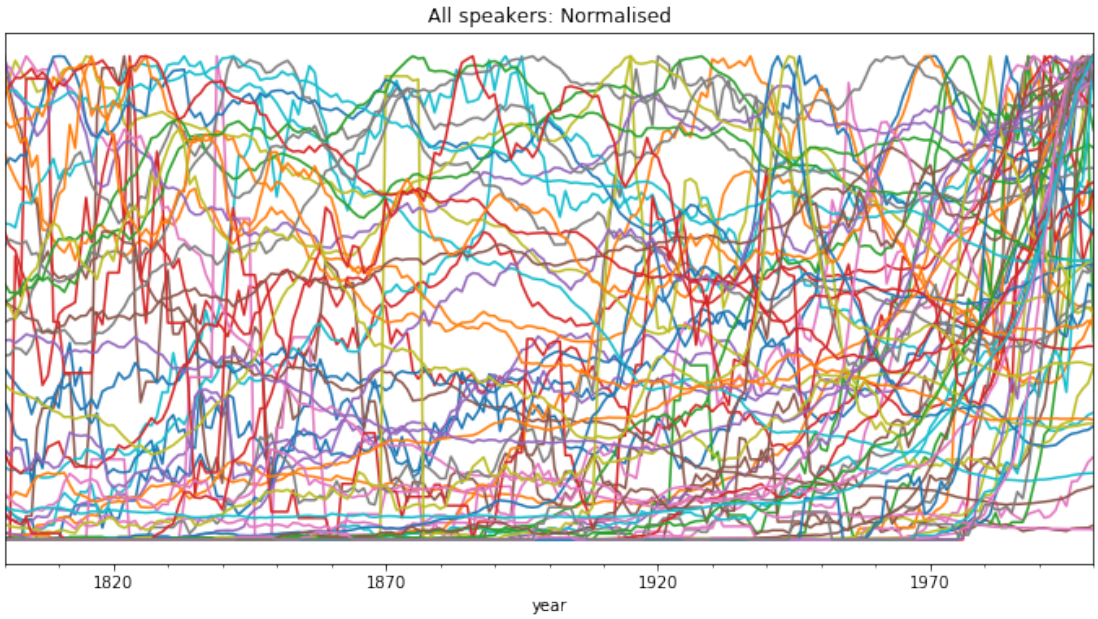


3 All speakers

The legend was removed for brevity.

[8]:
```
name="All speakers"
all_speakers_df = build_dataframe (all_speakers)
plot_absolute_graph(name, all_speakers_df, False)
plot_normalised_graph(name, all_speakers_df, False)
```





3.1 Top 10 by year

terms popular at different years in the dataset, but this is generalisable, I chose the start, end, and 1975 for this report.

[57]:
```
transpose = all_speakers_df.transpose()

year = pd.to_datetime('1800')
transpose[[year]].sort_values(year, ascending=False).head(10)
```

[57]:
```
year         1800-01-01
mistakes       1.000000
contagion      1.000000
candour        0.978684
belonging      0.968685
astronomy      0.957074
power          0.942220
intention      0.917471
own power      0.866510
revolution     0.799774
love           0.772841
```

[59]:
```
year = pd.to_datetime('2000')
transpose[[year]].sort_values(year, ascending=False).head(10)
```

[59]:
```
year                  2000-01-01
where are you from           1.0
machine learning             1.0
industrial design            1.0
```

```
innovation                1.0
fresh perspective         1.0
space tourism             1.0
space                     1.0
how we think              1.0
strategies                1.0
diverse                   1.0
```

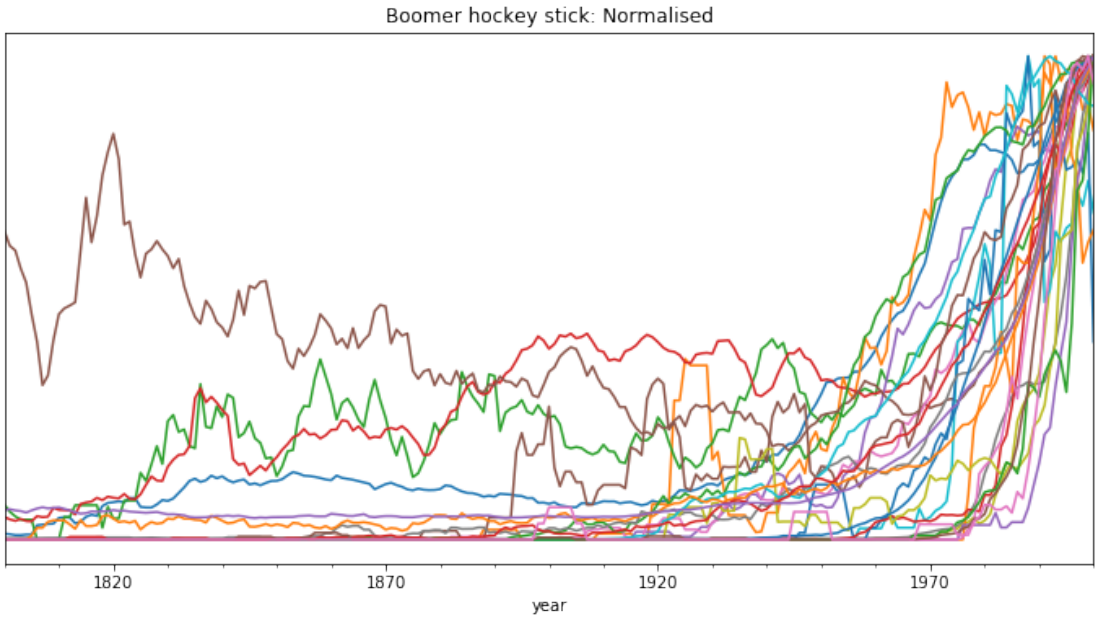
[60]:
```
year = pd.to_datetime('1975')
transpose[[year]].sort_values(year, ascending=False).head(10)
```

[60]:
```
year                    1975-01-01
social responsibility     0.997136
science                   0.930018
uniquely creative         0.919128
innovator                 0.909906
personal power            0.873250
revolution                0.852936
mistakes                  0.821783
space                     0.788144
belief                    0.784425
innovation                0.779259
```

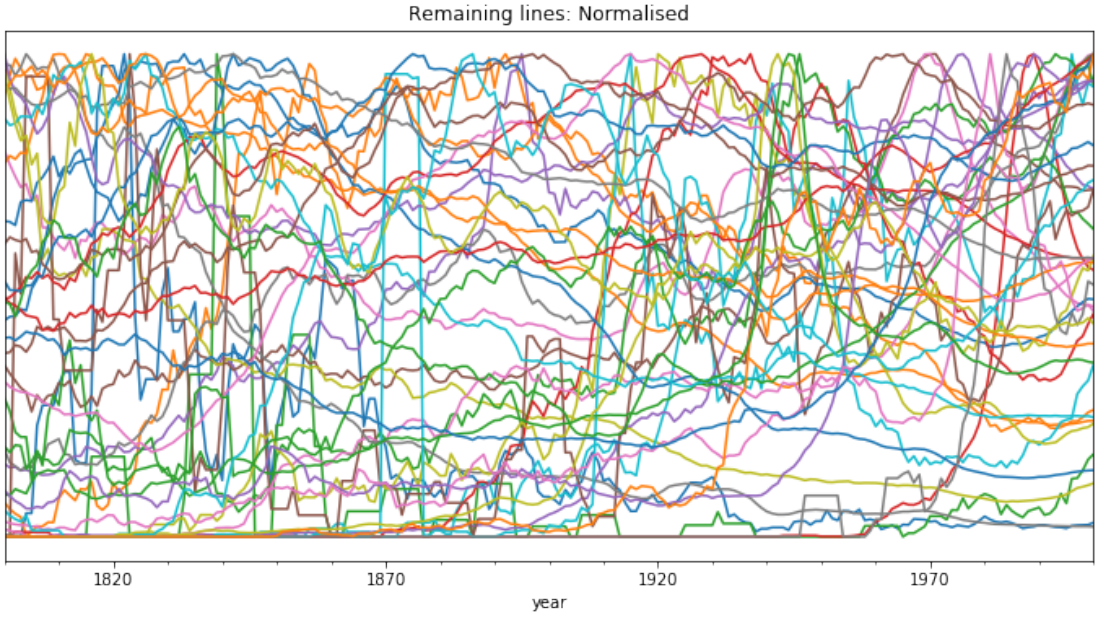
3.2 The boomer hockey stick

I chose to isolate terms that showed this trend just to show how prevalent it is. It'd be interesting to use machine learning and random words from the dictionary to find terms that have this shape.

[63]:
```
boomer_hockey = ["stereotype", "uniquely creative", "where are you from", "my␣
 ↪identity", "limitless potential", "own journey", "just ignore it", "online",␣
 ↪"psychological profiling", "cognitive empathy", "affective empathy",␣
 ↪"radical empathy", "compassion and understanding", "machine learning", "skin␣
 ↪in the game", "dawning of a new age", "lifestyle diseases", "learn and␣
 ↪grow", "relationship value", "problem solving", "strategies", "how we␣
 ↪think", "space tourism", "fresh perspective", "challenges", "empowering␣
 ↪people", "empower everyone"]
boomer_hockey_df = build_dataframe(boomer_hockey)
plot_normalised_graph("Boomer hockey stick", boomer_hockey_df, False)
```



[22]:
```
not_boomer = [term for term in all_speakers if term not in boomer_hockey]
not_boomer_df = build_dataframe(not_boomer)
plot_normalised_graph("Remaining lines", not_boomer_df, False)
```



4 Conclusions

- Even for relatively recent ideas we se a preponderance of terms and ideas popularised around the 70s
- This is true even for the younger speakers
- nothing new is really new
- This could indicate either a bias within the audience or a bias in the selection comittee
- On the strength of this one could assume that Ted is a brand for 40 year olds
- Without further research this cannot be extrapolated to wider society, but it's an interesting experiment

5 Future enhancements

there are a number of ways that I'd like to improve on this research:

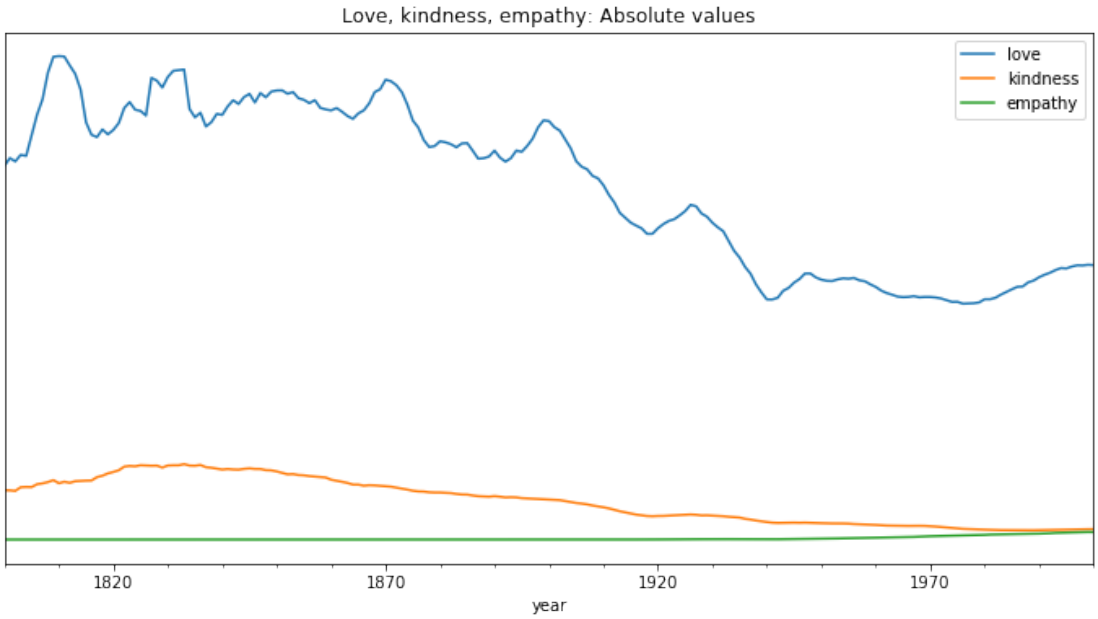
- generalise this modelling for different events, maybe iterate over youtube channels?
- use the youtube API to download the autogenerated TTS for their subtitle system
- develop an algorithm to mine interesting keywords (frequency analysis? an API?)
- rewrite the logic in JS that can be injected back into google's app
- machine learning AI to do linefitting and cluster analysis

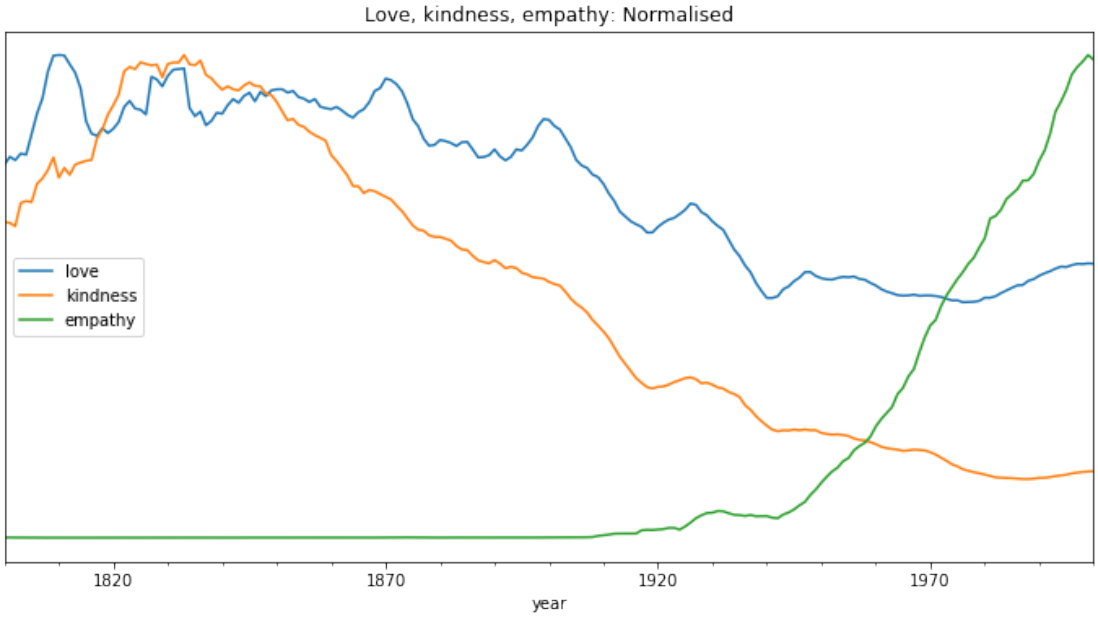
6 Appendix

6.1 Love, kindness, empathy

included by request

[65]:
```
name="Love, kindness, empathy"
lke = ["love", "kindness", "empathy"]
#get_ngram_csv(lke)
lke_df = build_dataframe(lke)
plot_absolute_graph(name, lke_df)
plot_normalised_graph(name, lke_df)
```


Love, kindness, empathy: Absolute values
love
kindness
empathy
1820
1870
1920
1970
year


Love, kindness, empathy: Normalised
love
kindness
empathy
1820
1870
1920
1970
year