Plotting alignment data

In [1]:
```
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
import json
import csv
import math
import pandas as pd
import utils.db_utils as db
import utils.plot_utils as plot
import utils.file_utils as file
import config

############################################
# get configuration
cfg = config.getConfig() # configure values in config.js
############################################

targetLang = cfg['targetLang']
bibleType = cfg['targetBibleType']
tWordsTypeList = cfg['tWordsTypeList']
dbPath = cfg['dbPath']
trainingDataPath = cfg['trainingDataPath']
testamentStr = cfg['testamentStr']
baseDataPath = cfg['baseDataPath']
```

In [2]:
```
# get alignments for tWords

minAlignments = 20
remove = ['ὁ', 'τὸ', 'τὰ', 'αὐτός', 'λέγω', 'ὡς', 'μέν', 'εἷς']

alignmentsForWord, filteredAlignmentsForWord = db.fetchAlignmentDataForAllTWordsCached(trainingDataPath, bib
print(f"Original Language Alignments: {len(filteredAlignmentsForWord)}")
```

```
Using cached Alignments
Unfiltered Alignments: 4368
filtered alignments by original list count is 243
Size of filtered alignments by original ./data/en/ult/TrainingData/kt_en_ult_NT_alignments_by_orig_20.json i
s 7.628 MB
Size of filtered alignments by original ./data/en/ult/TrainingData/kt_en_ult_NT_alignments_by_orig_20.csv is
2.076 MB
Filtered Alignments: 243
Using cached Alignments
Unfiltered Alignments: 538
filtered alignments by original list count is 33
Size of filtered alignments by original ./data/en/ult/TrainingData/names_en_ult_NT_alignments_by_orig_20.jso
n is 1.298 MB
Size of filtered alignments by original ./data/en/ult/TrainingData/names_en_ult_NT_alignments_by_orig_20.csv
is 0.363 MB
Filtered Alignments: 33
Using cached Alignments
Unfiltered Alignments: 7380
filtered alignments by original list count is 250
Size of filtered alignments by original ./data/en/ult/TrainingData/other_en_ult_NT_alignments_by_orig_20.jso
n is 6.275 MB
Size of filtered alignments by original ./data/en/ult/TrainingData/other_en_ult_NT_alignments_by_orig_20.csv
is 1.670 MB
Filtered Alignments: 250
Original Language Alignments: 429
```

Analysis of alignments for tWords in the en_ult:

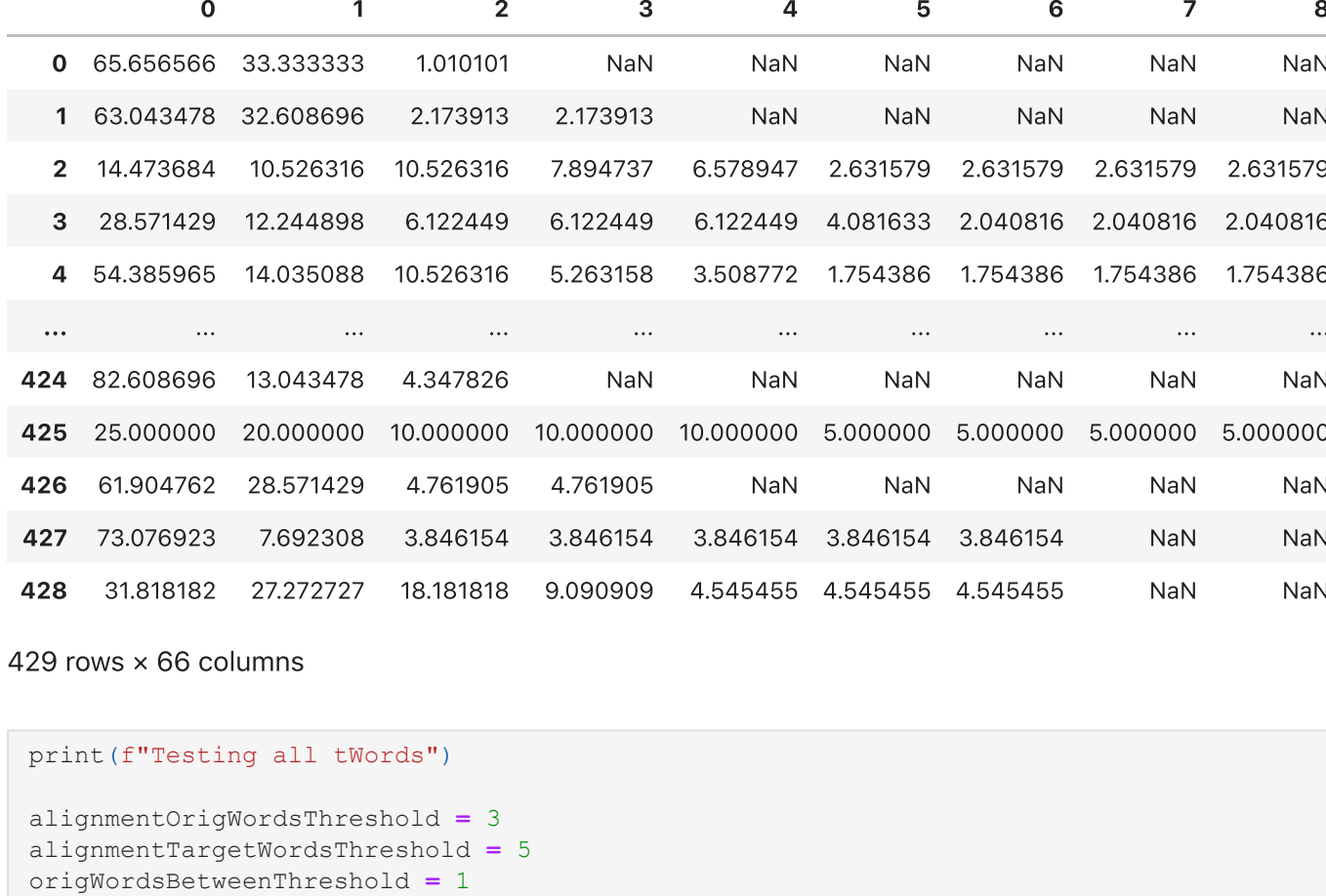
Frequency of alignments:

****Note that each line on the graphs below represents an alignment for a specific word. For example we have separate lines for 'Θεός', 'Θεὸς', or 'Θεοῦ' even though they have the same lemma. It made sense to group the alignments this way since aligners are likely to choose different target language words based on morphology of the word.*

In [3]:
```
frequenciesOfAlignments, stats = db.getFrequenciesOfFieldInAlignments(filteredAlignmentsForWord, 'alignmentT

print(f"Plotting of {len(filteredAlignmentsForWord)} tWord Alignments")
title = f"Plot of Variability of Specific Alignments in tWords"
ylabel = "Percent of Specific Alignments"
xlimit = [0, 10]
ylimit = [0, 75]
outputTable = plot.plotFrequencies(frequenciesOfAlignments, title, ylabel, showXValues=False, xlimit=xlimit,
csvPath = "plotData_freqOfTWords.csv"
db.saveListToCSV(csvPath, outputTable)
```

```
Plotting of 429 tWord Alignments
```

Plot of Variability of Specific Alignments in tWords



Out[3]:

	0	1	2	3	4	5	6	7	8	9	...	56	57
0	65.656566	33.333333	1.010101	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN
1	63.043478	32.608696	2.173913	2.173913	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN
2	14.473684	10.526316	10.526316	7.894737	6.578947	2.631579	2.631579	2.631579	2.631579	2.631579	...	NaN	NaN
3	28.571429	12.244898	6.122449	6.122449	6.122449	4.081633	2.040816	2.040816	2.040816	2.040816	...	NaN	NaN
4	54.385965	14.035088	10.526316	5.263158	3.508772	1.754386	1.754386	1.754386	1.754386	1.754386	...	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...
424	82.608696	13.043478	4.347826	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN
425	25.000000	20.000000	10.000000	10.000000	10.000000	5.000000	5.000000	5.000000	5.000000	5.000000	...	NaN	NaN
426	61.904762	28.571429	4.761905	4.761905	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN
427	73.076923	7.692308	3.846154	3.846154	3.846154	3.846154	3.846154	NaN	NaN	NaN	...	NaN	NaN
428	31.818182	27.272727	18.181818	9.090909	4.545455	4.545455	4.545455	NaN	NaN	NaN	...	NaN	NaN

429 rows × 66 columns

In [4]:
```
print(f"Testing all tWords")

alignmentOrigWordsThreshold = 3
alignmentTargetWordsThreshold = 5
origWordsBetweenThreshold = 1
targetWordsBetweenThreshold = 1
alignmentFrequencyMinThreshold = 5

type_ = 'all_twords'

warningPath = f'{baseDataPath}/{type_}_{bibleType}_{testamentStr}_warnings.json'
warningData = db.generateWarnings(warningPath, type_, bibleType, filteredAlignmentsForWord, alignmentOrigWor
                                  alignmentTargetWordsThreshold, origWordsBetweenThreshold,
                                  targetWordsBetweenThreshold, alignmentFrequencyMinThreshold,
                                  tag=f'{minAlignments}')

print(f"Found {len(warningData)} alignments to check - min threshold {minAlignments}")

frequencyWarnings = warningData[warningData['frequencyWarning'].str.len() > 0]
print (f"\nFound {len(frequencyWarnings)} frequencyWarnings")
frequencyWarningsByOrigWords = frequencyWarnings['originalWord'].value_counts()
print (f"FrequencyWarnings by original word:")
frequencyWarningsByOrigWords
```

```
Testing all tWords
Found 1436 alignments to check - min threshold 20

Found 1201 frequencyWarnings
FrequencyWarnings by original word:
```

Out[4]:
```
ἐγένετο      65
Θεοῦ         50
Ἰησοῦς       37
λόγον        37
Χριστοῦ      33
             ..
Πατὴρ         1
Ἅγιον         1
νόμον         1
μαθηταῖς      1
εὐαγγελίου    1
Name: originalWord, Length: 168, dtype: int64
```

In [5]:
```
frequencyWarningsByLemma = frequencyWarnings['lemma'].value_counts()
print (f"FrequencyWarnings by lemma:")
frequencyWarningsByLemma
```

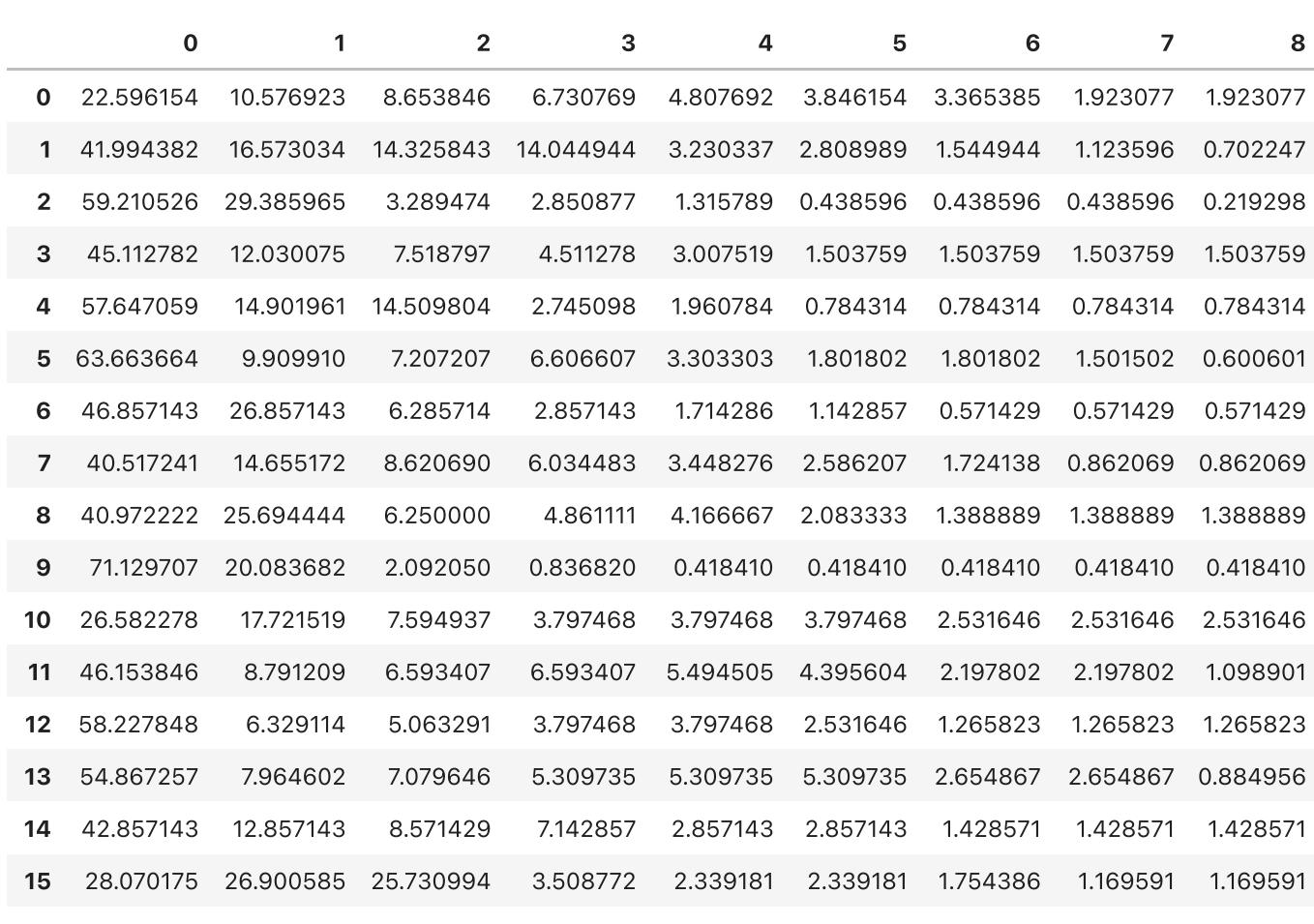
```
FrequencyWarnings by lemma:
```

Out[5]:
```
θεός        104
Ἰησοῦς       77
γίνομαι      65
λόγος        53
ὁράω         50
           ...
θρόνος        1
Σίμων         1
ἔρημος        1
δαιμόνιον     1
βασιλεία      1
Name: lemma, Length: 97, dtype: int64
```

In [6]:
```
minNumberOfWarnings = 10
origWordsWithWarnings = []
for key in frequencyWarningsByOrigWords.keys():
    count = frequencyWarningsByOrigWords[key]
    if count > minNumberOfWarnings:
        origWordsWithWarnings.append(key)

warningsAlignments = {}
for word in origWordsWithWarnings:
    warningsAlignments[word] = filteredAlignmentsForWord[word]
print(f"Found {len(origWordsWithWarnings)} original words with frequency warnings")

frequenciesOfAlignments, stats = db.getFrequenciesOfFieldInAlignments(warningsAlignments, 'alignmentText')

print(f"Plotting of {len(warningsAlignments)} tWord Alignments")
title = f"Plot of Variability of Alignments with Warnings"
ylabel = "Percent of Specific Alignments"
xlimit = [0, 10]
outputTable = plot.plotFrequencies(frequenciesOfAlignments, title, ylabel, showXValues=False, xlimit=xlimit)
csvPath = f"plotData_freqOfTWords_minWarnings_{minNumberOfWarnings}.csv"
db.saveListToCSV(csvPath, outputTable)
```

```
Found 29 original words with frequency warnings
Plotting of 29 tWord Alignments
```

Plot of Variability of Alignments with Warnings



Out[6]:

	0	1	2	3	4	5	6	7	8	9	...	56	
0	22.596154	10.576923	8.653846	6.730769	4.807692	3.846154	3.365385	1.923077	1.923077	1.442308	...	0.480769	0.
1	41.994382	16.573034	14.325843	14.044944	3.230337	2.808989	1.544944	1.123596	0.702247	0.561798	...	NaN	
2	59.210526	29.385965	3.289474	2.850877	1.315789	0.438596	0.438596	0.438596	0.219298	0.219298	...	NaN	
3	45.112782	12.030075	7.518797	4.511278	3.007519	1.503759	1.503759	1.503759	1.503759	1.503759	...	NaN	
4	57.647059	14.901961	14.509804	2.745098	1.960784	0.784314	0.784314	0.784314	0.784314	0.784314	...	NaN	
5	63.663664	9.909910	7.207207	6.606607	3.303303	1.801802	1.801802	1.501502	0.600601	0.600601	...	NaN	
6	46.857143	26.857143	6.285714	2.857143	1.714286	1.142857	0.571429	0.571429	0.571429	0.571429	...	NaN	
7	40.517241	14.655172	8.620690	6.034483	3.448276	2.586207	1.724138	0.862069	0.862069	0.862069	...	NaN	
8	40.972222	25.694444	6.250000	4.861111	4.166667	2.083333	1.388889	1.388889	1.388889	0.694444	...	NaN	
9	71.129707	20.083682	2.092050	0.836820	0.418410	0.418410	0.418410	0.418410	0.418410	0.418410	...	NaN	
10	26.582278	17.721519	7.594937	3.797468	3.797468	3.797468	2.531646	2.531646	2.531646	2.531646	...	NaN	
11	46.153846	8.791209	6.593407	6.593407	5.494505	4.395604	2.197802	2.197802	1.098901	1.098901	...	NaN	
12	58.227848	6.329114	5.063291	3.797468	3.797468	2.531646	1.265823	1.265823	1.265823	1.265823	...	NaN	
13	54.867257	7.964602	7.079646	5.309735	5.309735	5.309735	2.654867	2.654867	0.884956	0.884956	...	NaN	
14	42.857143	12.857143	8.571429	7.142857	2.857143	2.857143	1.428571	1.428571	1.428571	1.428571	...	NaN	
15	28.070175	26.900585	25.730994	3.508772	2.339181	2.339181	1.754386	1.169591	1.169591	0.584795	...	NaN	
16	27.710843	13.253012	13.253012	9.638554	8.433735	4.819277	2.409639	2.409639	1.204819	1.204819	...	NaN	
17	61.111111	6.944444	6.944444	5.555556	2.777778	1.388889	1.388889	1.388889	1.388889	1.388889	...	NaN	
18	55.555556	7.936508	4.761905	3.174603	3.174603	3.174603	1.587302	1.587302	1.587302	1.587302	...	NaN	
19	66.206897	6.896552	6.896552	6.206897	4.827586	2.758621	2.068966	1.379310	0.689655	0.689655	...	NaN	
20	72.727273	10.389610	2.597403	2.597403	2.597403	1.298701	1.298701	1.298701	1.298701	1.298701	...	NaN	
21	60.606061	16.666667	4.545455	3.030303	1.515152	1.515152	1.515152	1.515152	1.515152	1.515152	...	NaN	
22	33.168317	16.831683	15.841584	12.376238	7.425743	4.950495	3.465347	0.990099	0.990099	0.495050	...	NaN	
23	28.712871	20.132013	16.171617	11.221122	10.231023	7.920792	1.650165	0.990099	0.660066	0.660066	...	NaN	
24	73.626374	4.395604	4.395604	4.395604	3.296703	3.296703	1.098901	1.098901	1.098901	1.098901	...	NaN	
25	49.056604	7.547170	5.660377	5.660377	3.773585	3.773585	3.773585	1.886792	1.886792	1.886792	...	NaN	
26	53.658537	14.634146	4.878049	2.439024	2.439024	2.439024	2.439024	2.439024	2.439024	2.439024	...	NaN	
27	61.006289	25.786164	6.289308	1.257862	0.628931	0.628931	0.628931	0.628931	0.628931	0.628931	...	NaN	
28	47.435897	12.820513	7.692308	6.410256	3.846154	2.564103	2.564103	2.564103	1.282051	1.282051	...	NaN	

29 rows × 66 columns

Analysis:

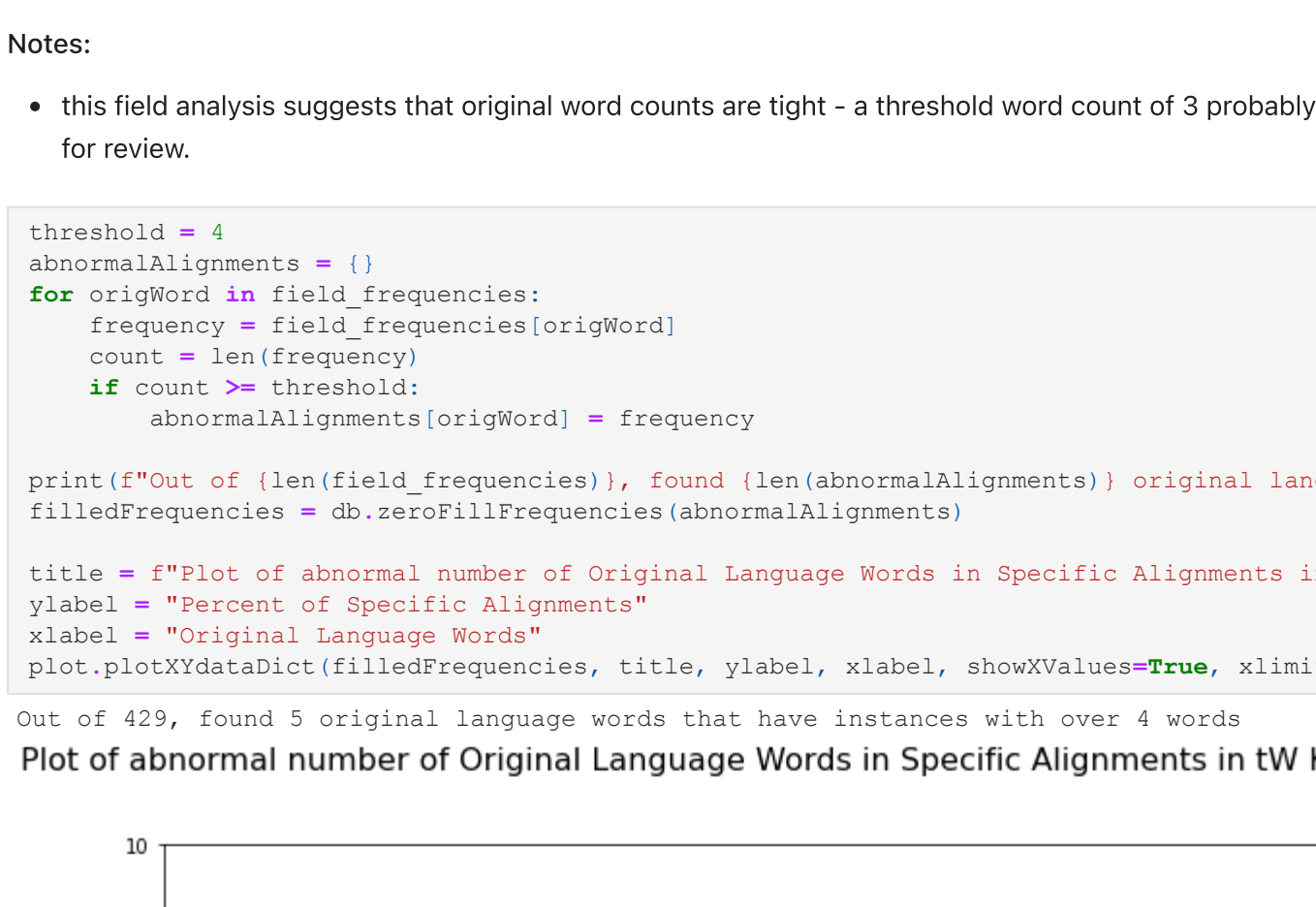
Analysis of numerical metrics:

Analysis of original language word count:

In [7]:
```
type_ = 'all'
field = 'origWordsCount'
field_frequencies, stats = db.getFrequenciesOfFieldInAlignments(filteredAlignmentsForWord, field, sortIndex
filledFrequencies = db.zeroFillFrequencies(field_frequencies)

print(f"Found {len(field_frequencies)} original language words for tW type {type_}")
title = f"Plot of number of Original Language Words in Specific Alignments in tW type {type_}"
ylabel = "Percent of Specific Alignments"
xlabel = "Original Language Words"
plot.plotXYdataDict(filledFrequencies, title, ylabel, xlabel, showXValues=True, xlimit=[1, 8])
```

```
Found 429 original language words for tW type all
```

Plot of number of Original Language Words in Specific Alignments in tW type all



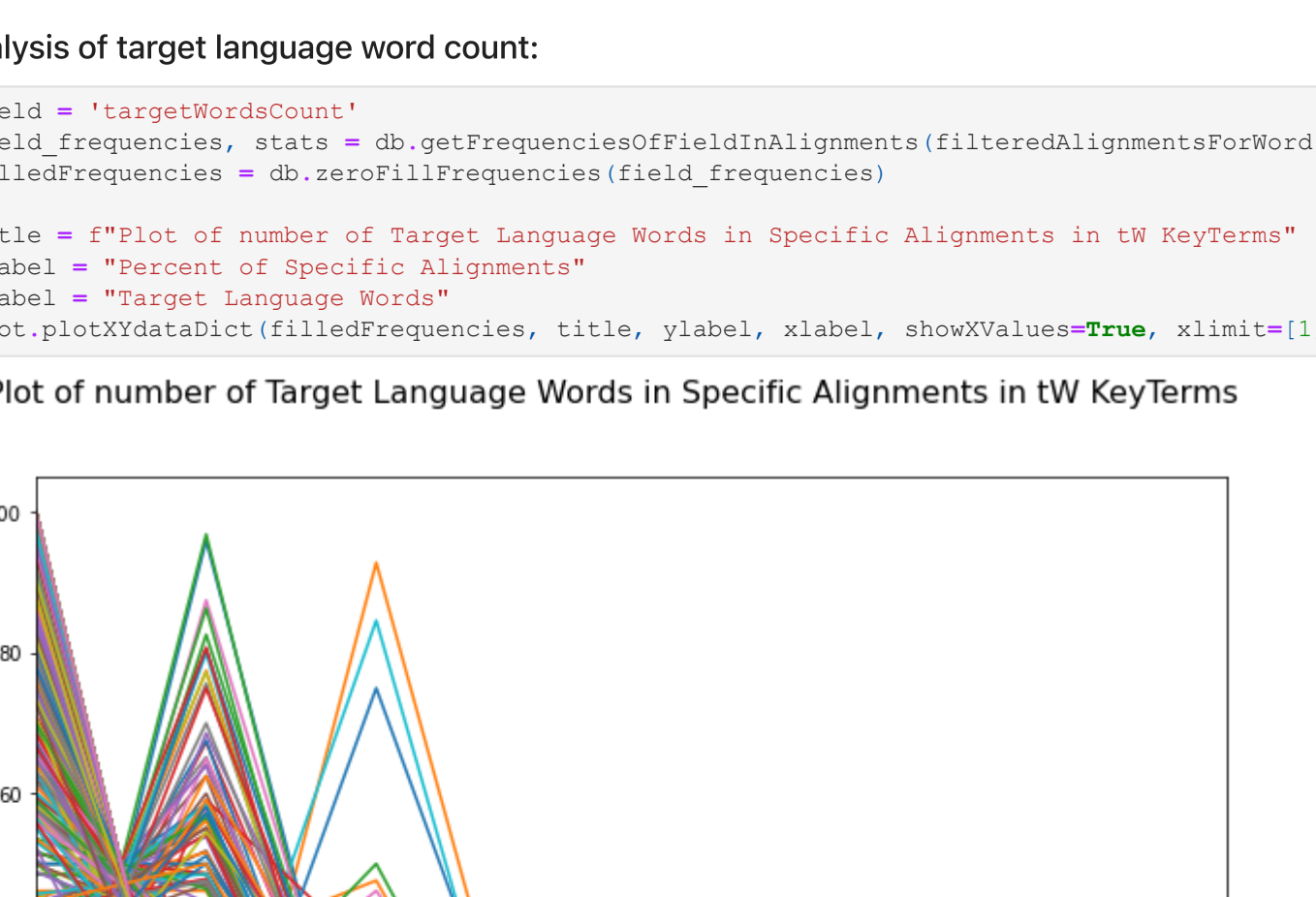
Notes:

- this field analysis suggests that original word counts are tight - a threshold word count of 3 probably good for Greek to flag for review.

In [8]:
```
threshold = 4
abnormalAlignments = {}
for origWord in field_frequencies:
    frequency = field_frequencies[origWord]
    count = len(frequency)
    if count >= threshold:
        abnormalAlignments[origWord] = frequency

print(f"Out of {len(field_frequencies)}, found {len(abnormalAlignments)} original language words that have i
filledFrequencies = db.zeroFillFrequencies(abnormalAlignments)

title = f"Plot of abnormal number of Original Language Words in Specific Alignments in tW KeyTerms"
ylabel = "Percent of Specific Alignments"
xlabel = "Original Language Words"
plot.plotXYdataDict(filledFrequencies, title, ylabel, xlabel, showXValues=True, xlimit=[threshold, 10], ylim
```

```
Out of 429, found 5 original language words that have instances with over 4 words
```

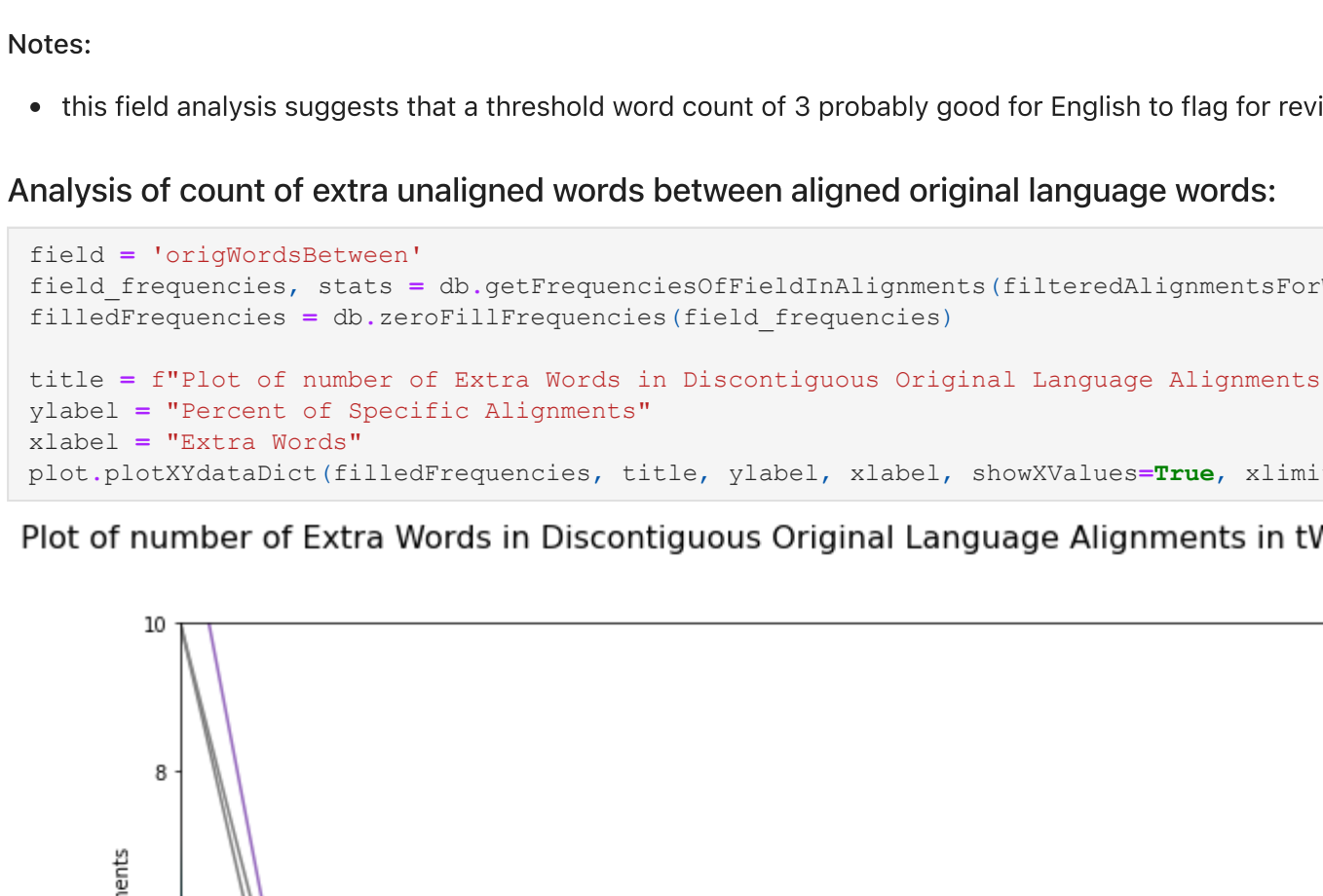
Plot of abnormal number of Original Language Words in Specific Alignments in tW KeyTerms



Analysis of target language word count:

In [9]:
```
field = 'targetWordsCount'
field_frequencies, stats = db.getFrequenciesOfFieldInAlignments(filteredAlignmentsForWord, field, sortIndex
filledFrequencies = db.zeroFillFrequencies(field_frequencies)

title = f"Plot of number of Target Language Words in Specific Alignments in tW KeyTerms"
ylabel = "Percent of Specific Alignments"
xlabel = "Target Language Words"
plot.plotXYdataDict(filledFrequencies, title, ylabel, xlabel, showXValues=True, xlimit=[1, 8])
```

Plot of number of Target Language Words in Specific Alignments in tW KeyTerms



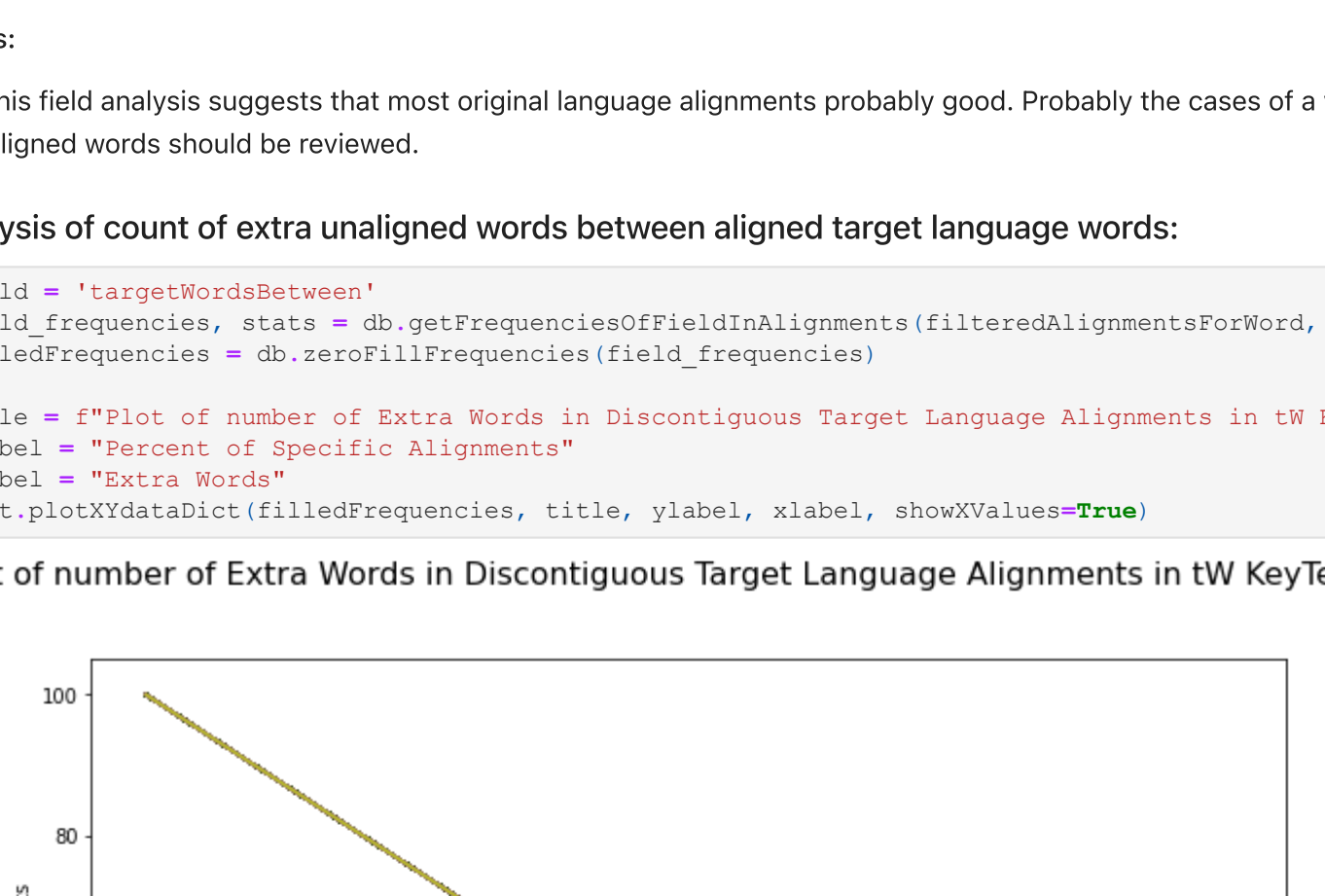
Notes:

- this field analysis suggests that a threshold word count of 3 probably good for English to flag for review.

Analysis of count of extra unaligned words between aligned original language words:

In [10]:
```
field = 'origWordsBetween'
field_frequencies, stats = db.getFrequenciesOfFieldInAlignments(filteredAlignmentsForWord, field, sortIndex
filledFrequencies = db.zeroFillFrequencies(field_frequencies)

title = f"Plot of number of Extra Words in Discontiguous Original Language Alignments in tW KeyTerms"
ylabel = "Percent of Specific Alignments"
xlabel = "Extra Words"
plot.plotXYdataDict(filledFrequencies, title, ylabel, xlabel, showXValues=True, xlimit=[1, 8], ylimit=[0,10]
```

Plot of number of Extra Words in Discontiguous Original Language Alignments in tW KeyTerms



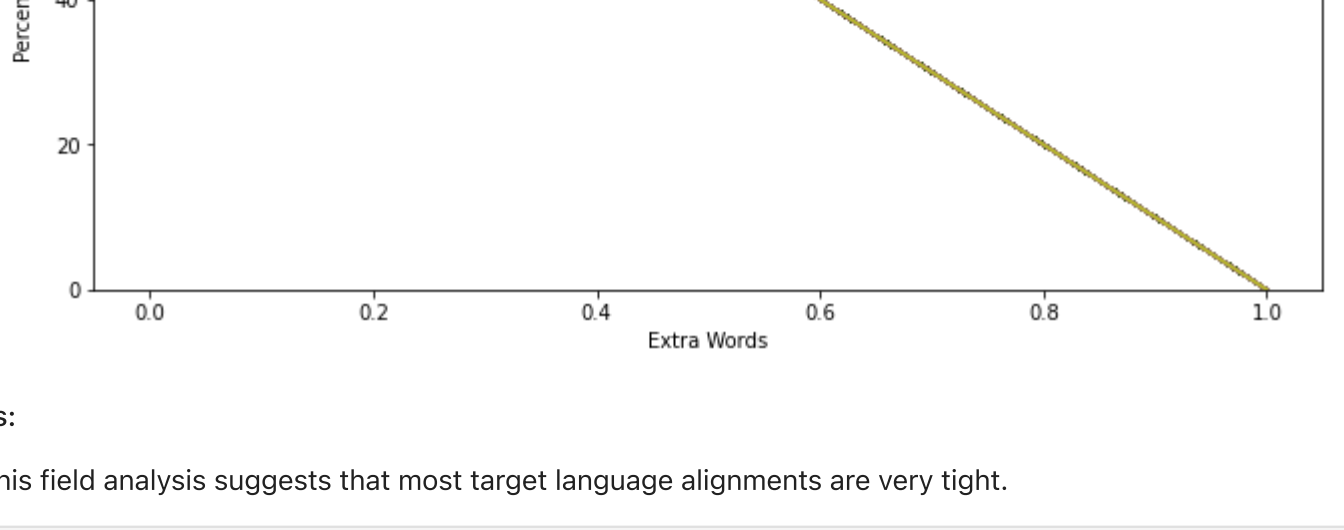
Notes:

- this field analysis suggests that most original language alignments probably good. Probably the cases of a word between aligned words should be reviewed.

Analysis of count of extra unaligned words between aligned target language words:

In [11]:
```
field = 'targetWordsBetween'
field_frequencies, stats = db.getFrequenciesOfFieldInAlignments(filteredAlignmentsForWord, field, sortIndex
filledFrequencies = db.zeroFillFrequencies(field_frequencies)

title = f"Plot of number of Extra Words in Discontiguous Target Language Alignments in tW KeyTerms"
ylabel = "Percent of Specific Alignments"
xlabel = "Extra Words"
plot.plotXYdataDict(filledFrequencies, title, ylabel, xlabel, showXValues=True)
```

Plot of number of Extra Words in Discontiguous Target Language Alignments in tW KeyTerms



Notes:

- this field analysis suggests that most target language alignments are very tight.

In [11]: