In [126…

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [4]:

```
df
```

Out[4]:

	id	survey_id	x	y	sound	concept	concept_other	date_
0	29	13	36.372925	195.413374	Where	other	where	
1	30	13	76.832695	176.607903	Want	WANT	None	
2	31	13	110.753512	196.639818	Love you	LOVE-YOU	None	
3	32	13	39.642401	154.531915	Bye	BYE	None	
4	33	13	114.022989	155.758359	Hi	HI	None	
...	...	...	...	...	...	...	...	
2575	2643	424	3556.363636	702.752885	Spielen	Play	None	00:0
2576	2644	424	545.454545	1592.506731	Garten	Garden	None	00:0
2577	2645	424	1479.272727	1649.206731	Später	Lat	None	00:0
2578	2646	424	2592.000000	1623.037500	Nein	No	None	00:0
2579	2647	424	3512.727273	1614.314423	Ja	yes	None	00:0

2580 rows × 8 columns

In [5]:

```
### Submissions with at least 5 buttons
acceptable_submission = df['survey_id'].value_counts() >= 5
usable_id = []
for i in range(len(acceptable_submission.to_list())):
    if acceptable_submission.to_list()[i] == True:
        usable_id.append(acceptable_submission.index[i])

model_df = df[df['survey_id'].isin(usable_id)]
model_df
```

Out[5]:

	id	survey_id	x	y	sound	concept	concept_other	date_
0	29	13	36.372925	195.413374	Where	other	where	
1	30	13	76.832695	176.607903	Want	WANT	None	

	id	survey_id	x	y	sound	concept	concept_other	date_
2	31	13	110.753512	196.639818	Love you	LOVE-YOU	None	
3	32	13	39.642401	154.531915	Bye	BYE	None	
4	33	13	114.022989	155.758359	Hi	HI	None	
...	...	...	...	...	...	...	...	
2575	2643	424	3556.363636	702.752885	Spielen	Play	None	00:0
2576	2644	424	545.454545	1592.506731	Garten	Garden	None	00:0
2577	2645	424	1479.272727	1649.206731	Später	Lat	None	00:0
2578	2646	424	2592.000000	1623.037500	Nein	No	None	00:0
2579	2647	424	3512.727273	1614.314423	Ja	yes	None	00:0

2166 rows × 8 columns

In [6]:

```
# Change sound to lowercase
model_df['sound'] = model_df['sound'].str.lower()
model_df.head()
```

```
<ipython-input-6-1ba44d8f91d3>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  model_df['sound'] = model_df['sound'].str.lower()
```

Out[6]:

	id	survey_id	x	y	sound	concept	concept_other	date_introduced
0	29	13	36.372925	195.413374	where	other	where	Na
1	30	13	76.832695	176.607903	want	WANT	None	Na
2	31	13	110.753512	196.639818	love you	LOVE-YOU	None	Na
3	32	13	39.642401	154.531915	bye	BYE	None	Na
4	33	13	114.022989	155.758359	hi	HI	None	Na

In [ ]:

Using survey_id 47

In [7]:
```
survey_47 = model_df.loc[model_df['survey_id'] == 47]
survey_47.head()
```

Out[7]:

	id	survey_id	x	y	sound	concept	concept_other	date_i
233	242	47	95.810277	102.554921	frustrated	Frustrated	None	2 00:00
234	243	47	91.541502	120.112039	concerned	Concerned	None	2 00:00
235	244	47	67.826087	220.235062	clean up	Put toys in toy box	None	2 00:00
236	245	47	73.517787	195.085677	high five	Someone will come high five Pixel	None	2 00:00
237	246	47	95.810277	186.069859	water	Pixel needs more water	None	2 00:00

In [8]:
```
### Use language model with linear cost function

def linear_cost(x):
    x = (x.max(axis=1) - x + 1)
    sum_of_rows = x.sum(axis=1)
    normalized_array = x / sum_of_rows[:, np.newaxis]
    return normalized_array

buttons = [Button(row["sound"], row["concept"], row["x"], row["y"]) for _,
b_linear = ButtonsBiGramLanguageModel(buttons,  linear_cost)
```

In [9]:
```
### The probability of each button
b_linear.P
```

Out[9]:
```
array([[0.0617264 , 0.05831973, 0.04459892, ..., 0.00611598, 0.01760712,
        0.01533418],
       [0.0576758 , 0.06274796, 0.04898392, ..., 0.00606594, 0.01584806,
        0.01379288],
       [0.06292103, 0.07244662, 0.13739   , ..., 0.0004412 , 0.0004412 ,
        0.0004412 ],
       ...,
       [0.00286621, 0.00348961, 0.00725349, ..., 0.05309681, 0.02196259,
        0.03725814],
       [0.01558257, 0.01495458, 0.01164312, ..., 0.02302348, 0.03504962,
        0.03350779],
       [0.00795528, 0.00739477, 0.00507834, ..., 0.03191309, 0.03035281,
        0.04889614]])
```

In [10]:
```
### Distance between a button from the rest of the buttons
b_linear.G
```

Out[10]:
```
array([[  0.        ,  18.06861426, 120.96168966, ..., 248.73876929,
        159.31908218, 217.53079206],
       [ 18.06861426,   0.        , 102.89334582, ..., 249.03366131,
        167.04260699, 224.29693022],
       [120.96168966, 102.89334582,   0.        , ..., 273.85438417,
        233.50259627, 282.00870615],
       ...,
       [248.73876929, 249.03366131, 273.85438417, ...,   0.        ,
        120.81376467,  90.14283145],
       [159.31908218, 167.04260699, 233.50259627, ..., 120.81376467,
          0.        ,  58.82188561],
       [217.53079206, 224.29693022, 282.00870615, ...,  90.14283145,
         58.82188561,   0.        ]])
```

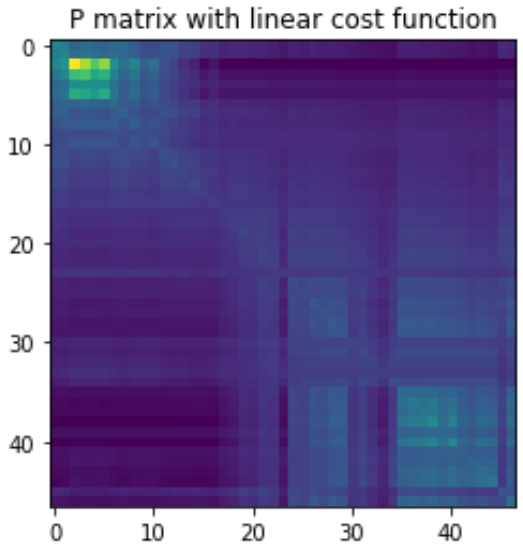
In [11]:
```
def quadratic_cost(x):
    x = ((x.max(axis=1) - x + 1)**2)
    sum_of_rows = x.sum(axis=1)
    normalized_array = x / sum_of_rows[:, np.newaxis]
    return normalized_array

buttons = [Button(row["sound"], row["concept"], row["x"], row["y"]) for _,
b_quadratic = ButtonsBiGramLanguageModel(buttons,  quadratic_cost)
```

In [12]:
```
def cubic_cost(x):
    x = ((x.max(axis=1) - x + 1)**3)
    sum_of_rows = x.sum(axis=1)
    normalized_array = x / sum_of_rows[:, np.newaxis]
    return normalized_array

buttons = [Button(row["sound"], row["concept"], row["x"], row["y"]) for _,
b_cubic = ButtonsBiGramLanguageModel(buttons,  cubic_cost)
```

Visualizations of P matrix with different cost functions

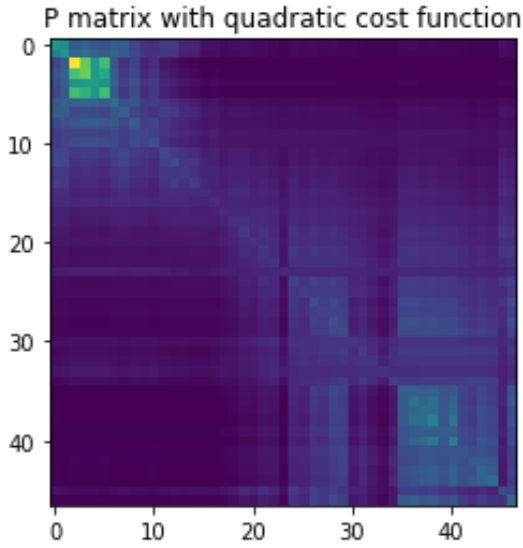
In [13]:
```
plt.imshow(b_linear.P)
plt.title("P matrix with linear cost function")
```

Out[13]:
```
Text(0.5, 1.0, 'P matrix with linear cost function')
```



In [14]:
```
plt.imshow(b_quadratic.P)
plt.title("P matrix with quadratic cost function")
```

Out[14]: Text(0.5, 1.0, 'P matrix with quadratic cost function')



In [15]:
```
plt.imshow(b_cubic.P)
plt.title("P matrix with cubic cost function")
```

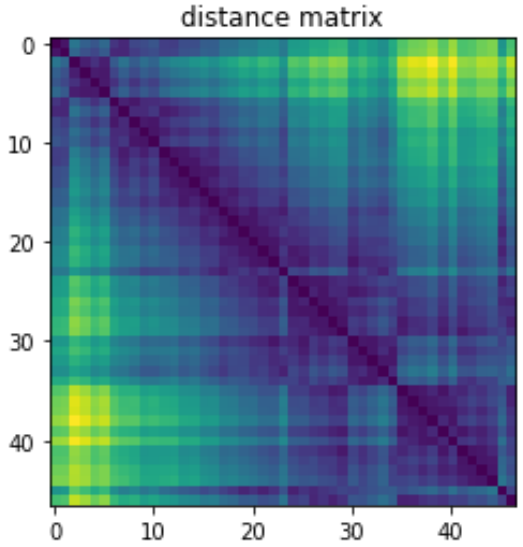
Out[15]: Text(0.5, 1.0, 'P matrix with cubic cost function')

The visualizations of P matrix with different cost functions are similar. However, there is some difference around (30, 30).

Visualization of distance matrix

In [16]:
```
plt.imshow(b_linear.G)
plt.title("distance matrix")
```

Out[16]:
```
Text(0.5, 1.0, 'distance matrix')
```



The distance matrix is the same across all three cost functions

Probability of sequence of buttons

In [17]:
```
def score(lang_model, buttons):
    old_button = buttons[0]
    p = 1
    for new_button in buttons[1:]:
        p = p * lang_model.P[old_button, new_button]
```

```
        old_button = new_button
    return p
```

In [18]:
```
### Probability of getting button 0, 2, and 4 with linear cost function

score(b_linear, [0, 2, 4])
```

Out[18]: 0.004514704452474175

Sampling a sequence of button presses

In [19]:
```
def sample(lang_model, length = None):
    ### if length of sequence isn't specify, randomly generate it
    if length == None:
        length = np.random.randint(1,4)
    len_of_model = len(lang_model.P)

    ### Use uniform distribution to generate an button
    old_button = list(np.random.randint(len_of_model, size=1))[0]
    probability = 1/len_of_model
    sequence = [old_button]
    for button in range(length - 1):

        ### Generate a new button using the probability of current button
        new_button = np.random.choice(len_of_model, 1, list(lang_model.P[c

        ### Update probability
        probability *= lang_model.score([old_button, new_button])

        ### Add new button to sequence
        sequence.append(new_button)

        ### Update button
        old_button = new_button

    #Get buttons with their index
    sequence = [lang_model.buttons[button_index] for button_index in seque

    return (sequence, probability)
```

In [131…
```
### Sampling a sequence of buttons with linear cost function

### Sampling without specifying the length of sequence
sample_one = sample(b_linear)

### Sampling with length of sequence
sample_two = sample(b_linear, 4)
```

In [128…
```
### Function that converts a tuple returned from sample to a tuple with se

### get_sound takes in a tuple and returns a tuple
```

```
def get_sound(tuple):
    sequence = [tuple[0][button].sound for button in np.arange(len(tuple[0
    return (sequence, tuple[1])
```

In [132…
```
print(get_sound(sample_one))

get_sound(sample_two)
```

```
(['ball'], 0.02127659574468085)
```

Out[132… `(['play', 'bird', 'outside', 'puzzle'], 2.727204282313739e-07)`

In [133…
```
### Sampling a sequence of buttons with quadratic cost function

### Sampling without specifying the length of sequence
sample_three = sample(b_quadratic)

### Sampling with length of sequence
sample_four = sample(b_quadratic, 4)
```

In [134…
```
print(get_sound(sample_three))

get_sound(sample_four)
```

```
(['drive'], 0.02127659574468085)
```

Out[134… `(['puzzle', 'now', 'happy', 'want'], 3.942969366558967e-09)`

In [135…
```
### Sampling a sequence of buttons with cubic cost function

### Sampling without specifying the length of sequence
sample_five = sample(b_cubic)

### Sampling with length of sequence
sample_six = sample(b_cubic, 4)
```

In [136…
```
print(get_sound(sample_five))

get_sound(sample_six)
```

```
(['tug', 'upstairs'], 0.0009750831144792087)
```

Out[136… `(['bird', 'high five', 'happy', 'potty'], 8.65896565118861e-08)`

Generate 10 2-button sequences containing the word "outside"

In [27]:
```
### Function that generates 10 sequences of buttons containing the word of
### button_num = int, 2 or 3 for button sequences
### cost = cost_function (b_linear, b_quadratic, b_cubic)
### word = word of your choice
```

```
def generate_ten_sequences(button_num, cost, word):
    ten_sequences = []
    total_sequences = []
    count = 0
    total = 0
    while count != 10:
        sampling = sample(cost, button_num)
        sounds = get_sound(sampling)[0]
        total_sequences.append(sounds)
        total += 1
        if word in sounds:
            ten_sequences.append([sounds, get_sound(sampling)[1]])
            count += 1

    ten_sequences = sorted(ten_sequences, key=lambda l:l[1], reverse=True)

    print('Total = ' + str(total) + ' sequences')
    return ten_sequences
```

In [28]:
```
generate_ten_sequences(2, b_linear, 'outside')
```

```
Total = 388 sequences
```

Out[28]:
```
[[['outside', 'now'], 0.000818771612648854],
 [['chew toy', 'outside'], 0.00080654376402274 73],
 [['???', 'outside'], 0.0006012081815875582],
 [['what', 'outside'], 0.0005955502712928398],
 [['cuddles', 'outside'], 0.0004744215671395084],
 [['outside', 'brush teeth'], 7.307758404941588e-05],
 [['outside', 'no'], 7.184028537403585e-05],
 [['outside', 'love you'], 7.089845135786947e-05],
 [['outside', 'love you'], 7.089845135786947e-05],
 [['outside', 'concerned'], 7.050112860212909e-05]]
```

Generate 10 3-button sequences containing the word "outside"

In [29]:
```
generate_ten_sequences(3, b_linear, 'outside')
```

```
Total = 348 sequences
```

Out[29]:
```
[[['outside', 'outside', 'porch '], 6.087394468800636e-05],
 [['with', 'outside', 'later'], 1.9855534497014383e-05],
 [['outside', 'put down your phone please', 'cuddles'], 1.001057037849097e
-05],
 [['no', 'stranger', 'outside'], 9.029174388657431e-06],
 [['walk', 'outside', 'cuddles'], 7.434409328574174e-06],
 [['sound', 'with', 'outside'], 6.77058083277108e-06],
 [['now', 'settle', 'outside'], 5.427520289286089e-06],
 [['puzzle', 'mom', 'outside'], 3.819547965918366e-06],
 [['walk', 'happy', 'outside'], 1.871044676837753e-06],
 [['bird', 'outside', 'water'], 1.4806462803761165e-06]]
```

In [ ]:
```
```

Using survey_id 45

Words in survey_id 45 are not in English

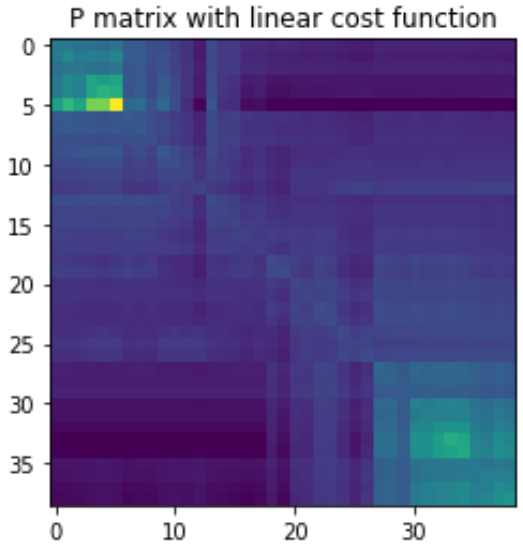
In [30]:
```
survey_45 = model_df.loc[model_df['survey_id'] == 45]
```

In [31]:
```
survey_45_buttons = [Button(row["sound"], row["concept"], row["x"], row["y

model_two_b_linear = ButtonsBiGramLanguageModel(survey_45_buttons,  linear
model_two_b_quadratic = ButtonsBiGramLanguageModel(survey_45_buttons,  qua
model_two_b_cubic = ButtonsBiGramLanguageModel(survey_45_buttons,  cubic_c
```

Visualizations of P matrix with different cost functions

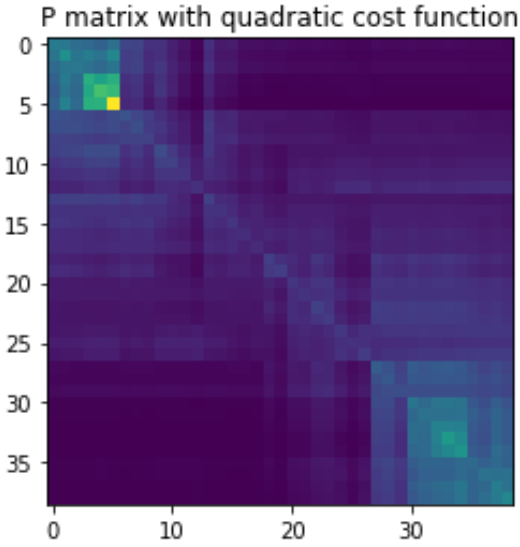
In [32]:
```
plt.imshow(model_two_b_linear.P)
plt.title("P matrix with linear cost function")
```

Out[32]:
```
Text(0.5, 1.0, 'P matrix with linear cost function')
```



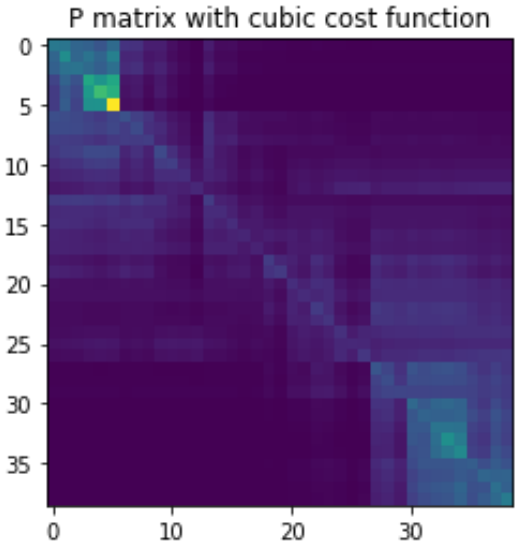
In [33]:
```
plt.imshow(model_two_b_quadratic.P)
plt.title("P matrix with quadratic cost function")
```

Out[33]:
```
Text(0.5, 1.0, 'P matrix with quadratic cost function')
```



In [34]:
```
plt.imshow(model_two_b_cubic.P)
plt.title("P matrix with cubic cost function")
```

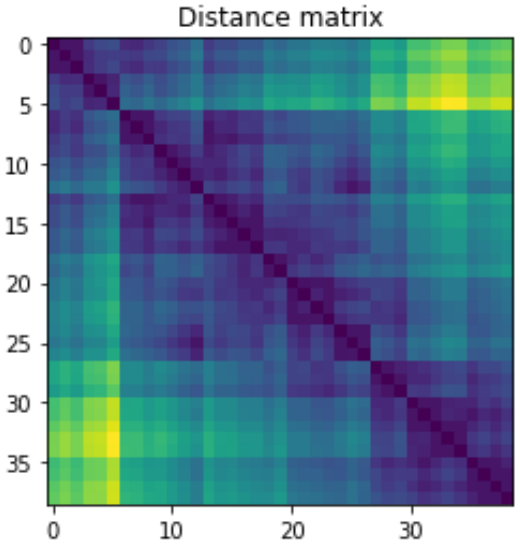
Out[34]:
```
Text(0.5, 1.0, 'P matrix with cubic cost function')
```



Visualization of distance matrix

In [35]:
```
plt.imshow(model_two_b_linear.G)
plt.title("Distance matrix")
```

Out[35]:
```
Text(0.5, 1.0, 'Distance matrix')
```



Sampling a sequence of button presses

In [137…
```
### Sampling a sequence of buttons with linear cost function

### Sampling without specifying the length of sequence
model_two_sample_one = sample(model_two_b_linear)

### Sampling with length of sequence
model_two_sample_two = sample(model_two_b_linear, 4)
```

In [138…
```
print(get_sound(model_two_sample_one))

get_sound(model_two_sample_two)
```

```
(['binnen', 'binnen'], 0.0013040620980479902)
```

Out[138… `(['boos', 'ja', 'wat', 'spelen'], 8.893045510363819e-08)`

In [139…
```
### Sampling a sequence of buttons with quadratic cost function

### Sampling without specifying the length of sequence
model_two_sample_three = sample(model_two_b_quadratic)

### Sampling with length of sequence
model_two_sample_four = sample(model_two_b_quadratic, 4)
```

In [140…
```
print(get_sound(model_two_sample_three))

get_sound(model_two_sample_four)
```

```
(['spelen', 'bal'], 0.0004604056328731926)
```

Out[140… `(['boos', 'oeps', 'blij', 'karlijn'], 1.6381970839091073e-07)`

In [141…

```
### Sampling a sequence of buttons with cubic cost function

### Sampling without specifying the length of sequence
model_two_sample_five = sample(model_two_b_cubic)

### Sampling with length of sequence
model_two_sample_six = sample(model_two_b_cubic, 4)
```

In [142…

```
print(get_sound(model_two_sample_five))

get_sound(model_two_sample_six)
```

```
(['eten', 'oeps', 'pim'], 2.405888707611927e-08)
```

Out[142… `(['wat', 'wil', 'knuffel', 'straks'], 5.420930795873049e-08)`

In [ ]:

Using survey_id 44

In [42]:

```
survey_44 = model_df.loc[model_df['survey_id'] == 44]
survey_44.head()
```

Out[42]:

	id	survey_id	x	y	sound	concept	concept_other	date_intro
157	166	44	928.565217	343.636364	want	Want	None	2020-00:00:00+
158	167	44	1280.739130	357.734266	where	WHERE	None	2020-00:00:00+
159	168	44	1473.260870	705.482517	no	NO	None	2020-00:00:00+
160	169	44	1294.826087	987.440559	go	GO	None	2019 00:00:00+
161	170	44	1092.913043	672.587413	love you	LOVE YOU	None	2019 00:00:00+

In [43]:

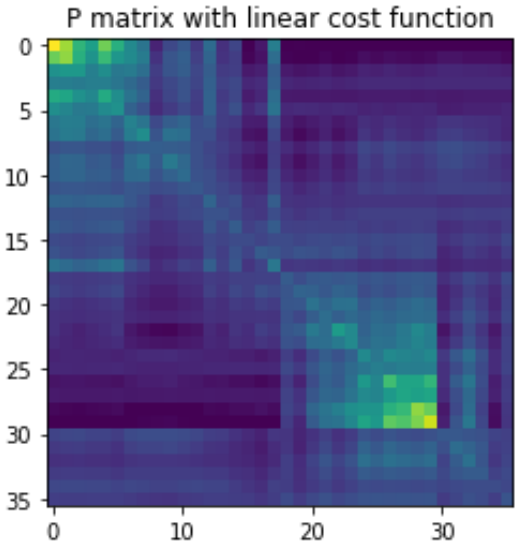
```
survey_44_buttons = [Button(row["sound"], row["concept"], row["x"], row["y

model_three_b_linear = ButtonsBiGramLanguageModel(survey_44_buttons,  line
model_three_b_quadratic = ButtonsBiGramLanguageModel(survey_44_buttons,  q
model_three_b_cubic = ButtonsBiGramLanguageModel(survey_44_buttons,  cubic
```

Visualization of P matrix with different cost functions

In [44]:

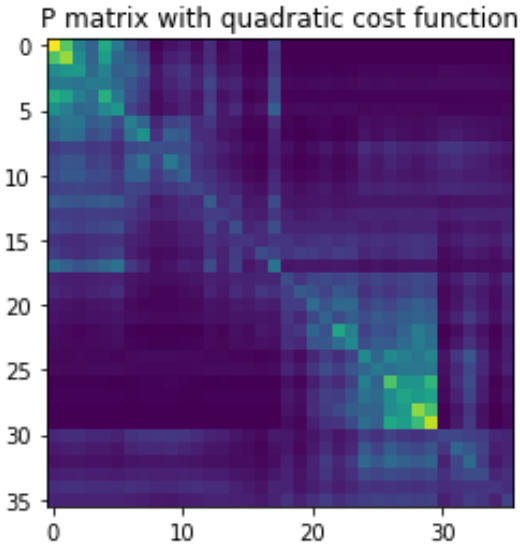
```
plt.imshow(model_three_b_linear.P)
plt.title("P matrix with linear cost function")
```

Out[44]: Text(0.5, 1.0, 'P matrix with linear cost function')



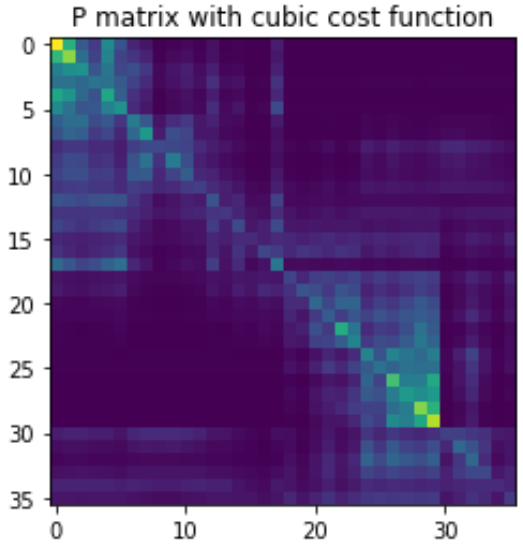
In [45]:
```
plt.imshow(model_three_b_quadratic.P)
plt.title("P matrix with quadratic cost function")
```

Out[45]: Text(0.5, 1.0, 'P matrix with quadratic cost function')



In [46]:
```
plt.imshow(model_three_b_cubic.P)
plt.title("P matrix with cubic cost function")
```

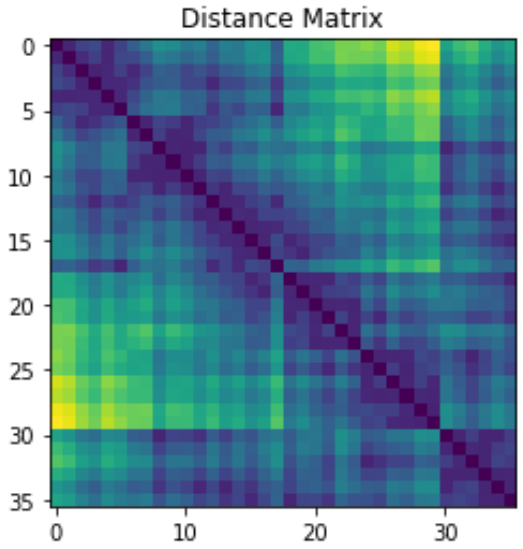
Out[46]: Text(0.5, 1.0, 'P matrix with cubic cost function')



Visualization of distance matrix

In [47]:
```
plt.imshow(model_three_b_linear.G)
plt.title("Distance Matrix")
```

Out[47]: Text(0.5, 1.0, 'Distance Matrix')



Sampling a sequence of button presses

In [143…
```
### Sampling a sequence of buttons with linear cost function

### Sampling without specifying the length of sequence
model_three_sample_one = sample(model_three_b_linear)

### Sampling with length of sequence
model_three_sample_two = sample(model_three_b_linear, 4)
```

In [144…
```
print(get_sound(model_three_sample_one))
```

```
get_sound(model_three_sample_two)
```

```
(['peepee'], 0.027777777777777776)
```

Out[144… `(['poopoo', 'no', 'look', 'mom'], 2.6496224584723004e-07)`

In [145…
```
### Sampling a sequence of buttons with quadratic cost function

### Sampling without specifying the length of sequence
model_three_sample_three = sample(model_three_b_quadratic)

### Sampling with length of sequence
model_three_sample_four = sample(model_three_b_quadratic, 5)
```

In [146…
```
print(get_sound(model_three_sample_three))

get_sound(model_three_sample_four)
```

```
(['want'], 0.027777777777777776)
```

Out[146… `(['where', 'dog park', 'bed', 'blair', 'poopoo'], 5.091447595130204e-13)`

In [147…
```
### Sampling a sequence of buttons with cubic cost function

### Sampling without specifying the length of sequence
model_three_sample_five = sample(model_three_b_cubic)

### Sampling with length of sequence
model_three_sample_six = sample(model_three_b_cubic, 3)
```

In [148…
```
print(get_sound(model_three_sample_five))

get_sound(model_three_sample_six)
```

```
(['blair', 'no'], 0.0026351230587808292)
```

Out[148… `(['more', 'go', 'porch'], 4.093593027622508e-07)`

Generate 10 2-button sequences of buttons with word "outside"

In [54]:
```
generate_ten_sequences(2, model_three_b_linear, 'outside')
```

```
Total = 101 sequences
```

Out[54]:
```
[[['dog park', 'outside'], 0.002550342284595942],
 [['outside', 'dog park'], 0.002156916262342359],
 [['later', 'outside'], 0.0015480542979676125],
 [['outside', 'ice'], 0.0009186104583179742],
 [['eleanor', 'outside'], 0.0007557441747198373],
 [['play', 'outside'], 0.0006981770502018892],
 [['mom', 'outside'], 0.0006128841448749021],
```

```
[['outside', 'frisbee'], 0.0003437145204707483],
[['outside', 'eleanor'], 0.0002641075916823815],
[['outside', 'no'], 0.00026006986524188994]]
```

Generate 10 3-button sequences of buttons with word "outside"

In [55]:
```
generate_ten_sequences(3, model_three_b_linear, 'outside')
```

```
Total = 142 sequences
```

Out[55]:
```
[[['more', 'outside', 'food'], 6.135486039939528e-05],
 [['outside', 'food', 'peepee'], 2.4160968639372223e-05],
 [['outside', 'outside', 'want'], 2.1586110141998667e-05],
 [['outside', 'tug', 'play'], 1.6685116963840408e-05],
 [['love you', 'later', 'outside'], 1.4919674884462857e-05],
 [['outside', 'now', 'treat'], 1.4046834639223517e-05],
 [['outside', 'want', 'play'], 1.0004328495779033e-05],
 [['outside', 'poopoo', 'outside'], 5.732328467132568e-06],
 [['outside', 'where', 'eleanor'], 5.328209496715246e-06],
 [['outside', 'dog park', 'no'], 3.2907661277753934e-06]]
```

In [ ]:
```
```

Using survey_id 61

In [56]:
```
survey_61 = model_df.loc[model_df['survey_id'] == 61]
survey_61.head()
```

Out[56]:

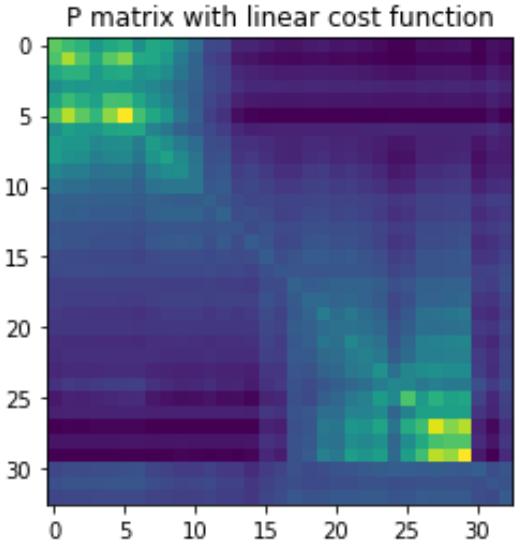
	id	survey_id	x	y	sound	concept	concept_other	date_introd
343	393	61	619.267561	528.575733	no	NO	None	2021- 00:00:00+
344	394	61	543.747126	854.184853	bye	BYE	None	2020- 00:00:00+
345	395	61	746.573436	744.210912	yes	YES	None	2020- 00:00:00+
346	396	61	947.242018	845.559446	hi	HI	None	2020- 00:00:00+
347	397	61	746.573436	966.315147	want	WANT	None	2020- 00:00:00+

In [57]:
```
survey_61_buttons = [Button(row["sound"], row["concept"], row["x"], row["y

model_four_b_linear = ButtonsBiGramLanguageModel(survey_61_buttons,  linea
model_four_b_quadratic = ButtonsBiGramLanguageModel(survey_61_buttons,  qu
model_four_b_cubic = ButtonsBiGramLanguageModel(survey_61_buttons,  cubic_
```

Visualizations of P matrix with different cost functions

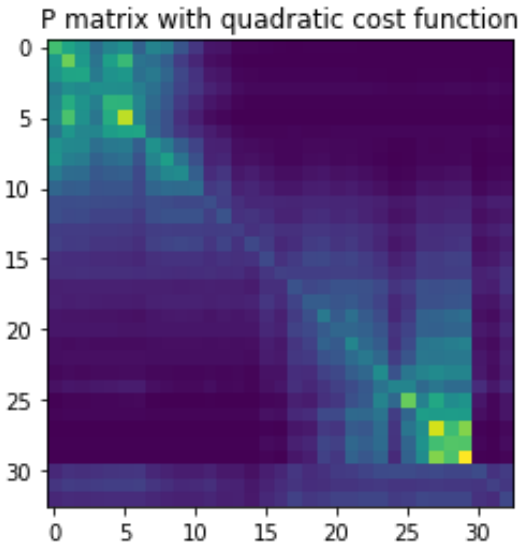
In [58]:
```
plt.imshow(model_four_b_linear.P)
plt.title("P matrix with linear cost function")
```

Out[58]:
```
Text(0.5, 1.0, 'P matrix with linear cost function')
```



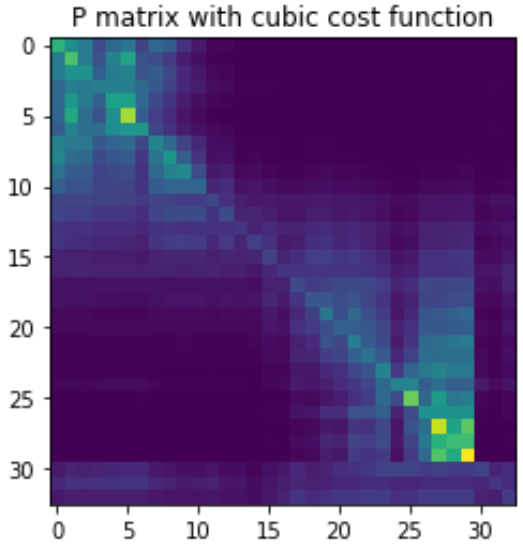
In [59]:
```
plt.imshow(model_four_b_quadratic.P)
plt.title("P matrix with quadratic cost function")
```

Out[59]:
```
Text(0.5, 1.0, 'P matrix with quadratic cost function')
```



In [60]:
```
plt.imshow(model_four_b_cubic.P)
plt.title("P matrix with cubic cost function")
```

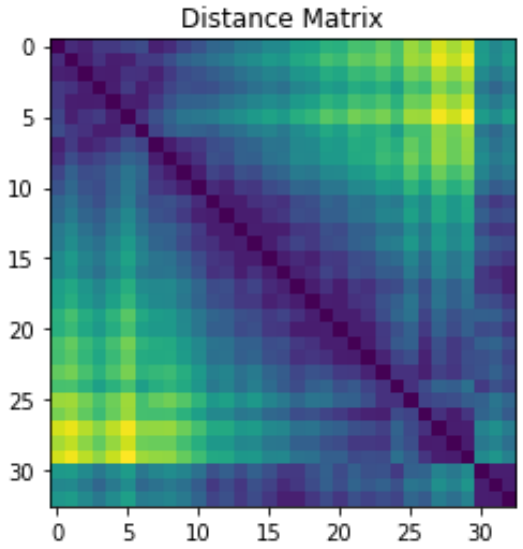
Out[60]:
```
Text(0.5, 1.0, 'P matrix with cubic cost function')
```



Visualization of distance matrix

In [61]:
```
plt.imshow(model_four_b_linear.G)
plt.title("Distance Matrix")
```

Out[61]: Text(0.5, 1.0, 'Distance Matrix')



Sampling a sequence of button presses

In [149…
```
### Sampling a sequence of buttons with linear cost function

### Sampling without specifying the length of sequence
model_four_sample_one = sample(model_four_b_linear)

### Sampling with length of sequence
model_four_sample_two = sample(model_four_b_linear, 4)
```

In [150…
```
print(get_sound(model_four_sample_one))
```

```
get_sound(model_four_sample_two)
```

```
(['water', 'want', 'later'], 4.453645664328919e-06)
```

Out[150… `(['copper', 'play', 'savannah', 'no'], 2.9267908288059106e-06)`

In [151…
```
### Sampling a sequence of buttons with quadratic cost function

### Sampling without specifying the length of sequence
model_four_sample_three = sample(model_four_b_quadratic)

### Sampling with length of sequence
model_four_sample_four = sample(model_four_b_quadratic, 3)
```

In [152…
```
print(get_sound(model_four_sample_three))

get_sound(model_four_sample_four)
```

```
(['outside', 'want', 'copper'], 6.856317655264684e-07)
```

Out[152… `(['copper', 'laser', 'savannah'], 5.836808927032919e-06)`

In [153…
```
### Sampling a sequence of buttons with cubic cost function

### Sampling without specifying the length of sequence
model_four_sample_five = sample(model_four_b_cubic)

### Sampling with length of sequence
model_four_sample_six = sample(model_four_b_cubic, 5)
```

In [154…
```
print(get_sound(model_four_sample_five))

get_sound(model_four_sample_six)
```

```
(['love you'], 0.030303030303030304)
```

Out[154… `(['pig', 'water', 'laser', 'laser', 'where'], 2.7475274994771115e-08)`

Generate 10 2-button sequences of buttons with word "outside"

In [68]:
```
generate_ten_sequences(2, model_four_b_linear, 'outside')
```

```
Total = 99 sequences
```

Out[68]:
```
[[['now', 'outside'], 0.0018961897634257358],
 [['outside', 'puzzle'], 0.0014462690054916317],
 [['chill out', 'outside'], 0.0010235434415697526],
 [['box', 'outside'], 0.0009678244103701688],
 [['box', 'outside'], 0.0009678244103701688],
 [['outside', 'box'], 0.0002334877245630953],
 [['outside', 'mommy'], 0.00022283714959105147],
```

```
[['outside', 'bye'], 0.00022245386943044182],
[['outside', 'scratch'], 0.00021817637704966357],
[['where', 'outside'], 1.4234079408408579e-06]]
```

Generate 10 3-button sequences of buttons with word "outside"

In [69]:
```
generate_ten_sequences(3, model_four_b_linear, 'outside')
```

```
Total = 87 sequences
```

Out[69]:
```
[[['outside', 'outside', 'pool'], 0.0002580860416731534],
 [['snake', 'outside', 'later'], 5.132637116867116e-05],
 [['ice', 'puzzle', 'outside'], 4.776964948955948e-05],
 [['hi', 'pool', 'outside'], 4.255907433551538e-05],
 [['copper', 'eat', 'outside'], 2.5871716577649713e-05],
 [['mommy', 'pig', 'outside'], 2.1053340379020278e-05],
 [['outside', 'potty', 'potty'], 9.214783714271353e-06],
 [['snake', 'outside', 'mommy'], 8.965778964969222e-06],
 [['outside', 'daddy', 'outside'], 4.181545056543139e-06],
 [['bye', 'outside', 'copper'], 5.143291128966149e-07]]
```

In [ ]:
```
```

Using survey_id 53

In [70]:
```
survey_53 = model_df.loc[model_df['survey_id'] == 53]
survey_53.head()
```

Out[70]:

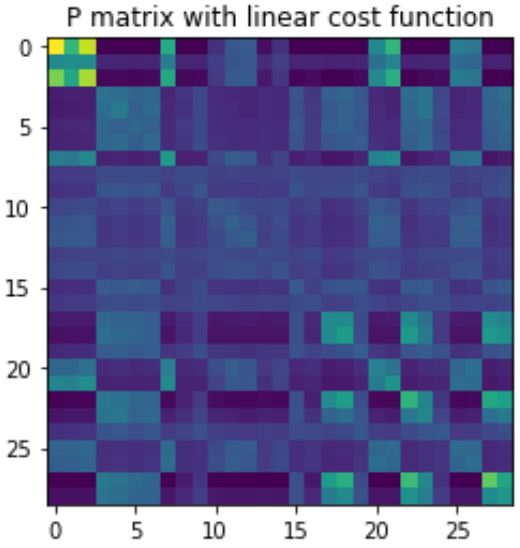
	id	survey_id	x	y	sound	concept	concept_other	date_int
59	319	53	2186.545299	278.409052	later	LATER	None	202 00:00:0
60	320	53	1911.636208	851.136325	all done	ALL-DONE	None	202 00:00:0
61	321	53	1881.090754	298.772727	tired	other	None	202 00:00:0
62	322	53	1445.818026	3119.136368	roxanne	learner's name	None	202 00:00:0
63	323	53	1186.181663	3103.863641	jim	PERSON-PARENT	None	202 00:00:0

In [71]:
```
survey_53_buttons = [Button(row["sound"], row["concept"], row["x"], row["y

model_five_b_linear = ButtonsBiGramLanguageModel(survey_53_buttons,  linea
model_five_b_quadratic = ButtonsBiGramLanguageModel(survey_53_buttons,  qu
model_five_b_cubic = ButtonsBiGramLanguageModel(survey_53_buttons,  cubic_
```

Visualization of P matrix with different cost functions

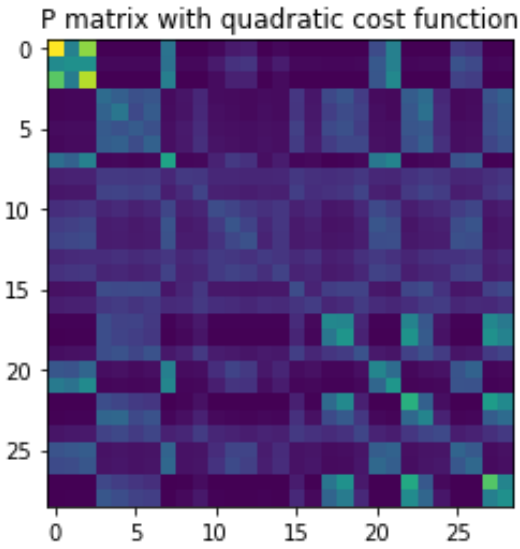
In [72]:
```
plt.imshow(model_five_b_linear.P)
plt.title("P matrix with linear cost function")
```

Out[72]:
```
Text(0.5, 1.0, 'P matrix with linear cost function')
```



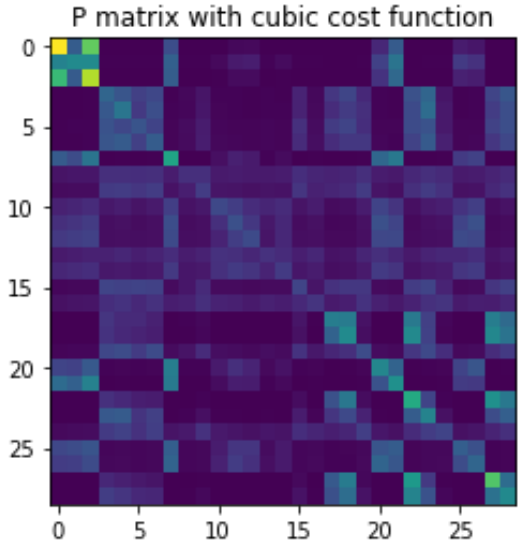
In [73]:
```
plt.imshow(model_five_b_quadratic.P)
plt.title("P matrix with quadratic cost function")
```

Out[73]:
```
Text(0.5, 1.0, 'P matrix with quadratic cost function')
```



In [74]:
```
plt.imshow(model_five_b_cubic.P)
plt.title("P matrix with cubic cost function")
```

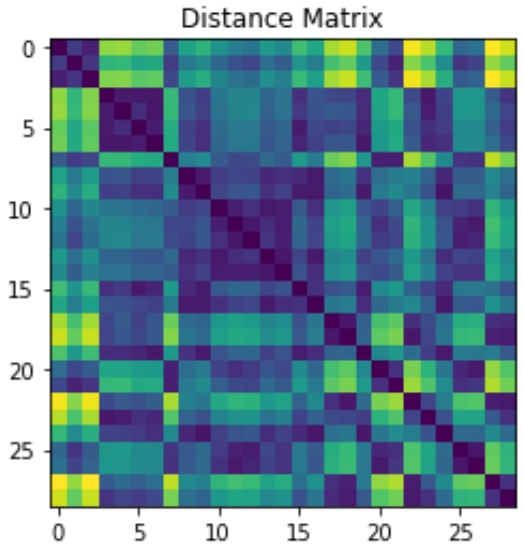
Out[74]:
```
Text(0.5, 1.0, 'P matrix with cubic cost function')
```



Visualization of distance matrix

In [75]:
```
plt.imshow(model_five_b_linear.G)
plt.title("Distance Matrix")
```

Out[75]: Text(0.5, 1.0, 'Distance Matrix')



In [155…
```
### Sampling a sequence of buttons with linear cost function

### Sampling without specifying the length of sequence
model_five_sample_one = sample(model_five_b_linear)

### Sampling with length of sequence
model_five_sample_two = sample(model_five_b_linear, 5)
```

In [156…
```
print(get_sound(model_five_sample_one))

get_sound(model_five_sample_two)
```

```
(['walk', 'ok'], 0.0003187452640892506)
```

Out[156…
```
(['want', 'outside', 'share', 'cookie', 'kong'], 7.430802002508796e-09)
```

In [157…
```
### Sampling a sequence of buttons with quadratic cost function

### Sampling without specifying the length of sequence
model_five_sample_three = sample(model_five_b_quadratic)

### Sampling with length of sequence
model_five_sample_four = sample(model_five_b_quadratic, 4)
```

In [158…
```
print(get_sound(model_five_sample_three))

get_sound(model_five_sample_four)
```

```
(['jim', 'grammy brenda'], 0.002559610645538265)
```

Out[158…
```
(['walk', 'ride', 'help', 'jim'], 4.034754211520864e-07)
```

In [159…
```
### Sampling a sequence of buttons with cubic cost function

### Sampling without specifying the length of sequence
model_five_sample_five = sample(model_five_b_cubic)

### Sampling with length of sequence
model_five_sample_six = sample(model_five_b_cubic, 3)
```

In [160…
```
print(get_sound(model_five_sample_five))

get_sound(model_five_sample_six)
```

```
(['tennis ball'], 0.034482758620689655)
```

Out[160…
```
(['all done', 'jim', 'ok'], 2.3856762696122334e-06)
```

Generate 10 2-button sequences of buttons with word "outside"

In [82]:
```
generate_ten_sequences(2, model_five_b_linear, 'outside')
```

```
Total = 105 sequences
```

Out[82]:
```
[[['tired', 'outside'], 0.00233267742607981],
 [['outside', 'outside'], 0.0020104437954279 48],
 [['outside', 'walk'], 0.0020024959599409006],
 [['sunshine', 'outside'], 0.0018425747987853734],
 [['tennis ball', 'outside'], 0.0016314523582681 39],
 [['snuggle', 'outside'], 0.0010822668427637539],
 [['share', 'outside'], 0.0009511540461331458],
 [['outside', 'roxanne'], 0.0007984209886999943],
 [['roxanne', 'outside'], 0.0005832461432685612],
 [['friend', 'outside'], 0.0004665465701090077]]
```

Generate 10 3-button sequences of buttons with word "outside"

In [83]:
```
generate_ten_sequences(3, model_five_b_linear, 'outside')
```

```
Total = 119 sequences
```

Out[83]:
```
[[['outside', 'walk', 'later'], 0.000141530443714765],
 [['tennis ball', 'outside', 'sunshine'], 8.872926550516833e-05],
 [['garden', 'outside', 'kong'], 5.797141652627938e-05],
 [['roxanne', 'walk', 'outside'], 3.649610950868179e-05],
 [['jim', 'roxanne', 'outside'], 3.468206327545865e-05],
 [['roxanne', 'eat', 'outside'], 3.321450924001142e-05],
 [['roxanne', 'kong', 'outside'], 2.428235766487895e-05],
 [['outside', 'tired', 'share'], 9.850826451754313e-06],
 [['outside', 'ok', 'cookie'], 7.952949249102558e-06],
 [['cookie', 'love you', 'outside'], 3.571594716419012e-06]]
```

In [ ]:
```
```

Using survey_id 13

In [84]:
```
survey_13 = model_df.loc[model_df['survey_id'] == 13]
survey_13.head()
```

Out[84]:

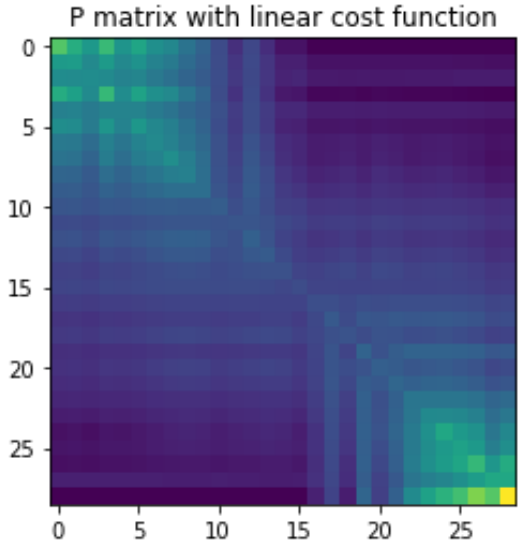
	id	survey_id	x	y	sound	concept	concept_other	date_introduced
0	29	13	36.372925	195.413374	where	other	where	Na
1	30	13	76.832695	176.607903	want	WANT	None	Na
2	31	13	110.753512	196.639818	love you	LOVE-YOU	None	Na
3	32	13	39.642401	154.531915	bye	BYE	None	Na
4	33	13	114.022989	155.758359	hi	HI	None	Na

In [85]:
```
survey_13_buttons = [Button(row["sound"], row["concept"], row["x"], row["y

model_six_b_linear = ButtonsBiGramLanguageModel(survey_13_buttons,  linear
model_six_b_quadratic = ButtonsBiGramLanguageModel(survey_13_buttons,  qua
model_six_b_cubic = ButtonsBiGramLanguageModel(survey_13_buttons,  cubic_c
```

Visualizations of P matrix with different cost functions

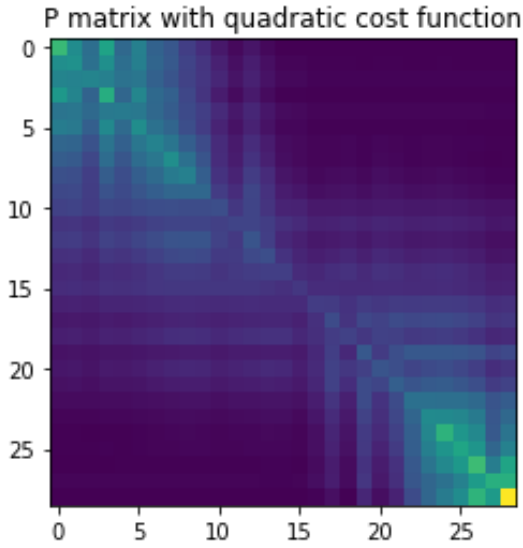
In [86]:
```
plt.imshow(model_six_b_linear.P)
plt.title("P matrix with linear cost function")
```

Out[86]:
```
Text(0.5, 1.0, 'P matrix with linear cost function')
```



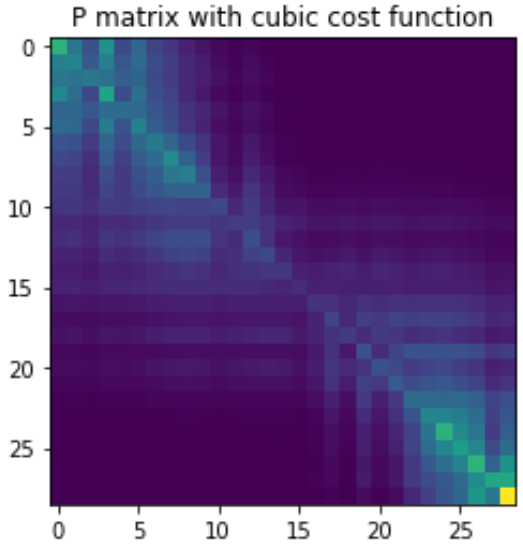
In [87]:
```
plt.imshow(model_six_b_quadratic.P)
plt.title("P matrix with quadratic cost function")
```

Out[87]:
```
Text(0.5, 1.0, 'P matrix with quadratic cost function')
```



In [88]:
```
plt.imshow(model_six_b_cubic.P)
plt.title("P matrix with cubic cost function")
```

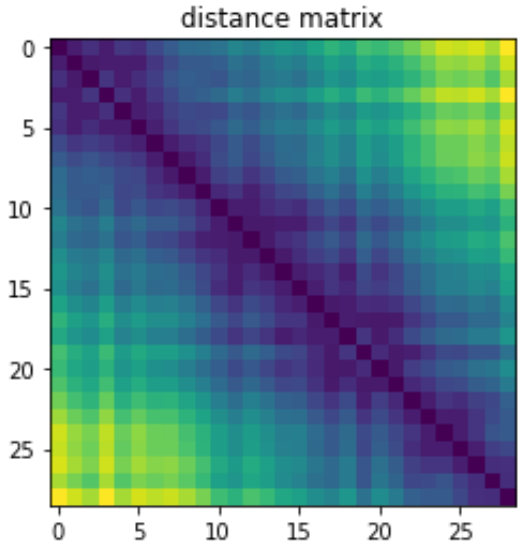
Out[88]:
```
Text(0.5, 1.0, 'P matrix with cubic cost function')
```



Visualization of distance matrix

In [89]:
```
plt.imshow(model_six_b_linear.G)
plt.title("distance matrix")
```

Out[89]: Text(0.5, 1.0, 'distance matrix')



Sampling a sequence of button presses

In [161…
```
### Sampling a sequence of buttons with linear cost function

### Sampling without specifying the length of sequence
model_six_sample_one = sample(model_six_b_linear)

### Sampling with length of sequence
model_six_sample_two = sample(model_six_b_linear, 4)
```

In [162…
```
print(get_sound(model_six_sample_one))
```

```
get_sound(model_six_sample_two)
```

```
(['copper', 'hi', 'chill out'], 3.139157625403163e-05)
```

Out[162… `(['pool', 'upstairs', 'ride', 'pool'], 1.7482339222790102e-06)`

In [163…
```
### Sampling a sequence of buttons with quadratic cost function

### Sampling without specifying the length of sequence
model_six_sample_three = sample(model_six_b_quadratic)

### Sampling with length of sequence
model_six_sample_four = sample(model_six_b_quadratic, 4)
```

In [164…
```
print(get_sound(model_six_sample_three))

get_sound(model_six_sample_four)
```

```
(['yes', 'water', 'scratch'], 1.533566895664364e-06)
```

Out[164… `(['scratch', 'later', 'water', 'walk'], 5.709126908498857e-07)`

In [165…
```
### Sampling a sequence of buttons with cubic cost function

### Sampling without specifying the length of sequence
model_six_sample_five = sample(model_six_b_cubic)

### Sampling with length of sequence
model_six_sample_six = sample(model_six_b_cubic, 4)
```

In [166…
```
print(get_sound(model_six_sample_five))

get_sound(model_six_sample_six)
```

```
(['chill out', 'outside', 'all done'], 9.496824992738756e-05)
```

Out[166… `(['chill out', 'pool', 'where', 'savannah'], 1.2822639103929367e-08)`

Generate 10 2-button sequences containing the word "outside"

In [96]:
```
generate_ten_sequences(2, model_six_b_linear, 'outside')
```

```
Total = 110 sequences
```

Out[96]:
```
[[['all done', 'outside'], 0.0032036984452090086],
 [['ball', 'outside'], 0.0018029832812720323],
 [['donut', 'outside'], 0.0017816997012687103],
 [['outside', 'water'], 0.0013962164060700077],
 [['pig', 'outside'], 0.0013571747611065281],
 [['eat', 'outside'], 0.0010126592832796146],
 [['scratch', 'outside'], 0.0009957349730368315],
 [['outside', 'scratch'], 0.0006670551404176407],
```

```
 [['outside', 'yes'], 0.00054132351998215 21],
 [['outside', 'yes'], 0.00054132351998215 21]]
```

Generate 10 3-button sequences containing the word "outside"

In [97]:
```
generate_ten_sequences(3, model_six_b_linear, 'outside')
```

```
Total = 57 sequences
```

Out[97]:
```
[[['water', 'outside', 'want'], 2.1676709729803845e-05],
 [['outside', 'play', 'chill out'], 1.8440555166264214e-05],
 [['outside', 'copper', 'chill out'], 1.6864078349589427e-05],
 [['outside', 'play', 'ball'], 1.589280545001994e-05],
 [['outside', 'daddy', 'upstairs'], 1.1184563414420438e-05],
 [['outside', 'copper', 'all done'], 1.036112476015947e-05],
 [['savannah', 'outside', 'eat'], 9.231049673579213e-06],
 [['pool', 'mommy', 'outside'], 6.589244154291781e-06],
 [['want', 'outside', 'scratch'], 5.2334709118729886e-06],
 [['ball', 'bye', 'outside'], 1.920161267626837e-06]]
```

In [ ]:
```
```

Using survey_id 57

In [98]:
```
survey_57 = model_df.loc[model_df['survey_id'] == 57]
survey_57.head()
```

Out[98]:

	id	survey_id	x	y	sound	concept	concept_other	da
302	352	57	447.006085	405.808219	want	WANT	None	0
303	353	57	661.691684	326.027397	what	what	None	0
304	354	57	864.109533	510.136986	no	NO	None	0
305	355	57	618.754564	577.643836	hmm	INTERROGATIVE-QUESTION	None	0
306	356	57	808.904665	743.342466	yes	YES	None	0

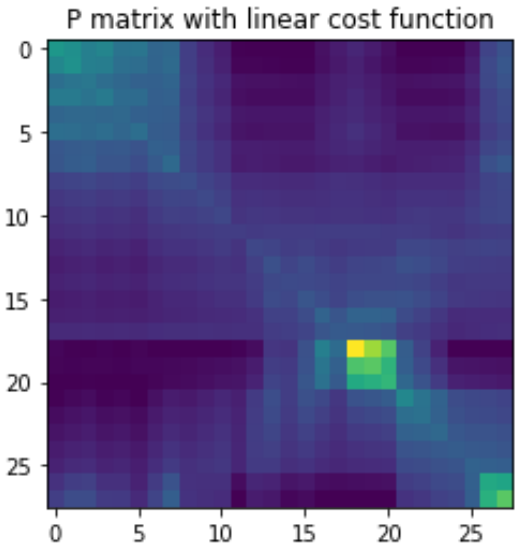
In [99]:
```
survey_57_buttons = [Button(row["sound"], row["concept"], row["x"], row["y

model_seven_b_linear = ButtonsBiGramLanguageModel(survey_57_buttons, linea
model_seven_b_quadratic = ButtonsBiGramLanguageModel(survey_57_buttons, qu
model_seven_b_cubic = ButtonsBiGramLanguageModel(survey_57_buttons, cubic_
```

Visualizations of P matrix with different cost functions

In [ ]:
```
```

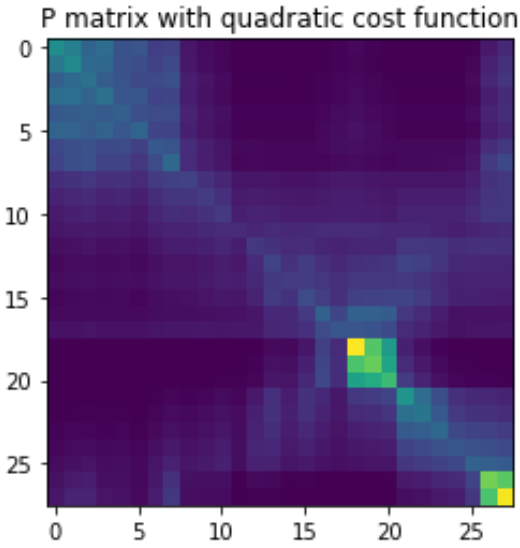
In [100…
```
plt.imshow(model_seven_b_linear.P)
plt.title("P matrix with linear cost function")
```

Out[100…
```
Text(0.5, 1.0, 'P matrix with linear cost function')
```



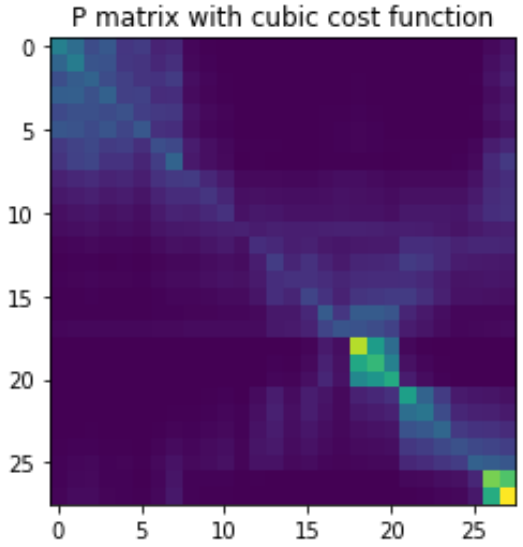
In [101…
```
plt.imshow(model_seven_b_quadratic.P)
plt.title("P matrix with quadratic cost function")
```

Out[101…
```
Text(0.5, 1.0, 'P matrix with quadratic cost function')
```



In [102…
```
plt.imshow(model_seven_b_cubic.P)
plt.title("P matrix with cubic cost function")
```

Out[102…
```
Text(0.5, 1.0, 'P matrix with cubic cost function')
```

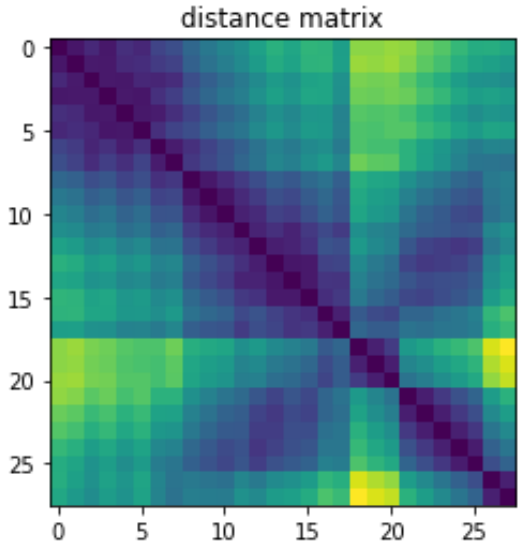


Visualization of distance matrix

In [103…

```
plt.imshow(model_seven_b_linear.G)
plt.title("distance matrix")
```

Out[103… Text(0.5, 1.0, 'distance matrix')



Sampling a sequence of button presses

In [167…

```
### Sampling a sequence of buttons with linear cost function

### Sampling without specifying the length of sequence
model_seven_sample_one = sample(model_seven_b_linear)

### Sampling with length of sequence
model_seven_sample_two = sample(model_seven_b_linear, 4)
```

In [168…

```
print(get_sound(model_seven_sample_one))
```

```
get_sound(model_seven_sample_two)
```

```
(['catnip', 'back', 'mad'], 5.4919328092929866e-05)
```

Out[168… `(['yes', 'mad', 'water', 'play'], 8.439261170476288e-07)`

In [170…

```
### Sampling a sequence of buttons with quadratic cost function

### Sampling without specifying the length of sequence
model_seven_sample_three = sample(model_seven_b_quadratic)

### Sampling with length of sequence
model_seven_sample_four = sample(model_seven_b_quadratic, 4)
```

In [171…

```
print(get_sound(model_seven_sample_three))

get_sound(model_seven_sample_four)
```

```
(['no', 'ouch', 'play'], 2.2005148314529285e-05)
```

Out[171… `(['worm', 'chin', 'ouch', 'bunny'], 1.4286489671627452e-08)`

In [172…

```
### Sampling a sequence of buttons with cubic cost function

### Sampling without specifying the length of sequence
model_seven_sample_five = sample(model_seven_b_cubic)

### Sampling with length of sequence
model_seven_sample_six = sample(model_seven_b_cubic, 4)
```

In [173…

```
print(get_sound(model_seven_sample_five))

get_sound(model_seven_sample_six)
```

```
(['mom'], 0.03571428571428571)
```

Out[173… `(['ear', 'want', 'chin', 'outside'], 6.65540441313717e-18)`

Generate 10 2-button sequences containing the word "outside"

In [110…

```
generate_ten_sequences(2, model_seven_b_linear, "outside")
```

```
Total = 138 sequences
```

Out[110…

```
[[['mad', 'outside'], 0.001800633468261209],
 [['outside', 'bunny'], 0.0016413471712027766],
 [['outside', 'ear'], 0.001553626443542876],
 [['outside', 'where'], 0.0012743004058766783],
 [['outside', 'food'], 0.0010782431588971509],
 [['pets', 'outside'], 0.0009170160329411447],
 [['mom', 'outside'], 0.0005894662226096595],
 [['no', 'outside'], 0.000301910622147414],
```

```
 [['want', 'outside'], 1.310293546401228e-06],
 [['want', 'outside'], 1.310293546401228e-06]]
```

Generate 10 3-button sequences containing the word "outside"

In [111…

```
generate_ten_sequences(3, model_seven_b_linear, "outside")
```

```
Total = 78 sequences
```

Out[111…

```
[[['outside', 'noise', 'bye'], 0.0003456468056113231],
 [['outside', 'outside', 'water'], 0.00010180338970335886],
 [['outside', 'no', 'want'], 5.5184815631324245e-05],
 [['food', 'worm', 'outside'], 5.191315637122996e-05],
 [['outside', 'mom', 'mom'], 5.137397281139447e-05],
 [['outside', 'bunny', 'pets'], 4.766796309976258e-05],
 [['outside', 'back', 'play'], 4.2675479298277324e-05],
 [['water', 'outside', 'what'], 3.2026696080311434e-05],
 [['love you', 'where', 'outside'], 1.2857904934943876e-05],
 [['mom', 'outside', 'yes'], 1.0874975300604973e-05]]
```

Using survey_id 63

In [112…

```
survey_63 = model_df.loc[model_df['survey_id'] == 63]
survey_63.head()
```

Out[112…

	id	survey_id	x	y	sound	concept	concept_other	date_in
460	509	63	82.113665	71.740674	all done	All Done	None	20 00:00:
461	510	63	242.608557	78.667850	find	Find an object	None	20 00:00:
462	511	63	816.337803	90.390763	outside	Anywhere immediately outside our apartment	None	20 00:00:
463	512	63	135.967433	214.547068	game	This starts a training session where Bertie ge...	None	20 00:00:
464	513	63	214.348659	211.349910	play	This starts a play session of fetch or tug	None	20 00:00:

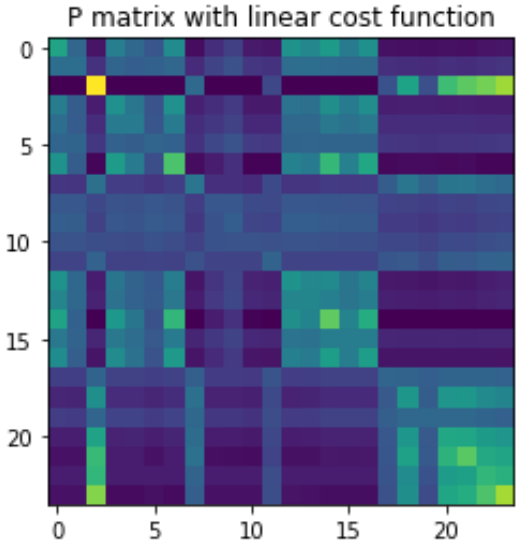
In [113…

```
survey_63_buttons = [Button(row["sound"], row["concept"], row["x"], row["y

model_eight_b_linear = ButtonsBiGramLanguageModel(survey_63_buttons, linea
model_eight_b_quadratic = ButtonsBiGramLanguageModel(survey_63_buttons, qu
model_eight_b_cubic = ButtonsBiGramLanguageModel(survey_63_buttons, cubic_
```

Visualizations of P matrix with different cost functions

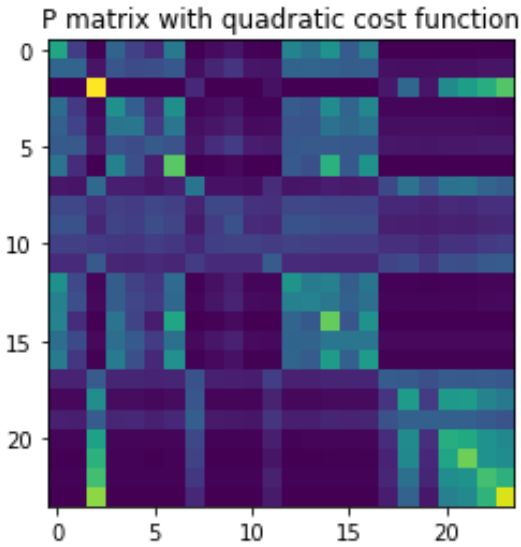
In [114…
```
plt.imshow(model_eight_b_linear.P)
plt.title("P matrix with linear cost function")
```

Out[114… `Text(0.5, 1.0, 'P matrix with linear cost function')`



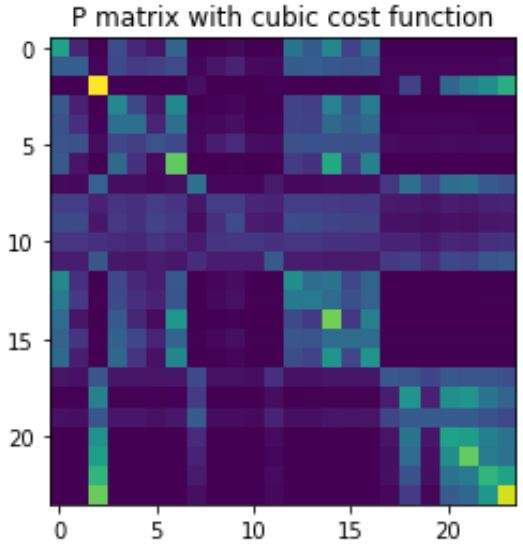
In [115…
```
plt.imshow(model_eight_b_quadratic.P)
plt.title("P matrix with quadratic cost function")
```

Out[115… `Text(0.5, 1.0, 'P matrix with quadratic cost function')`



In [116…
```
plt.imshow(model_eight_b_cubic.P)
plt.title("P matrix with cubic cost function")
```

Out[116… `Text(0.5, 1.0, 'P matrix with cubic cost function')`

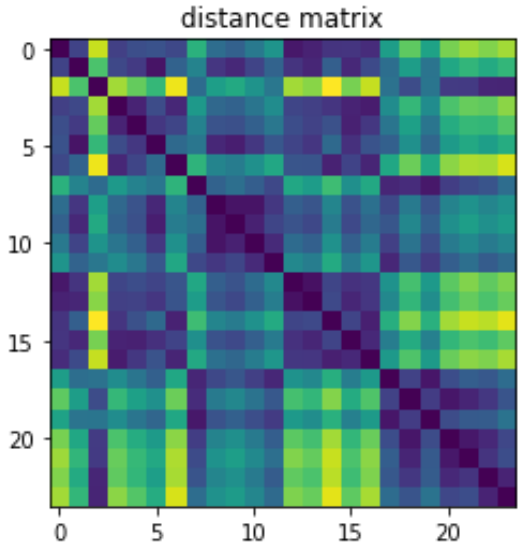


Visualization of distance matrix

In [117…

```
plt.imshow(model_eight_b_linear.G)
plt.title("distance matrix")
```

Out[117… Text(0.5, 1.0, 'distance matrix')



Sampling a sequence of button presses

In [174…

```
### Sampling a sequence of buttons with linear cost function

### Sampling without specifying the length of sequence
model_eight_sample_one = sample(model_eight_b_linear)

### Sampling with length of sequence
model_eight_sample_two = sample(model_eight_b_linear, 4)
```

In [175…

```
print(get_sound(model_eight_sample_one))
```

```
get_sound(model_eight_sample_two)
```

```
(['ball', 'bertie', 'outside'], 0.00014822491463298051)
```

Out[175… `(['hungry', 'errands', 'happy', 'play'], 4.287042979035088e-08)`

In [176…
```
### Sampling a sequence of buttons with quadratic cost function

### Sampling without specifying the length of sequence
model_eight_sample_three = sample(model_eight_b_quadratic)

### Sampling with length of sequence
model_eight_sample_four = sample(model_eight_b_quadratic, 4)
```

In [177…
```
print(get_sound(model_eight_sample_three))

get_sound(model_eight_sample_four)
```

```
(['love you', 'yes!'], 0.002747248358746435)
```

Out[177… `(['play', 'hungry', 'outside', 'walk'], 2.318530418655494e-13)`

In [178…
```
### Sampling a sequence of buttons with cubic cost function

### Sampling without specifying the length of sequence
model_eight_sample_five = sample(model_eight_b_cubic)

### Sampling with length of sequence
model_eight_sample_six = sample(model_eight_b_cubic, 4)
```

In [179…
```
print(get_sound(model_eight_sample_five))

get_sound(model_eight_sample_six)
```

```
(['outside'], 0.041666666666666664)
```

Out[179… `(['later', 'jack', 'love you', 'concerned'], 7.081035744119974e-08)`

Generate 10 2-button sequences containing the word "outside"

In [124…
```
generate_ten_sequences(2, model_eight_b_linear, "outside")
```

```
Total = 79 sequences
```

Out[124…
```
[[['outside', 'park'], 0.005733990527253551],
 [['bertie', 'outside'], 0.0041833092736542205],
 [['noise', 'outside'], 0.002282607355845538],
 [['noise', 'outside'], 0.002282607355845538],
 [['outside', 'hmm?'], 0.0018241900399970697],
 [['jack', 'outside'], 0.0014178536035540595],
 [['game', 'outside'], 0.0005940705721942011],
 [['all done', 'outside'], 0.0003997375070610104],
```

```
 [['outside', 'ball'], 9.038336909686924e-06],
 [['outside', 'find'], 9.038336909686924e-06]]
```

Generate 10 3-button sequences containing the word "outside"

In [125…

```
generate_ten_sequences(3, model_eight_b_linear, "outside")
```

```
Total = 77 sequences
```

Out[125…

```
[[['ball', 'outside', 'mama'], 0.0001586582109685782],
 [['jack', 'outside', 'mama'], 0.00014360830327212192],
 [['outside', 'hmm?', 'help'], 5.960027626258766e-05],
 [['hungry', 'happy', 'outside'], 4.2119203901651105e-05],
 [['outside', 'play', 'game'], 6.431853076313892e-07],
 [['outside', 'later', 'help'], 4.461374105145138e-07],
 [['outside', 'jack', 'mama'], 2.7440033667608446e-07],
 [['find', 'outside', 'ball'], 2.2754676994556996e-07],
 [['later', 'outside', 'jack'], 1.5786904666839505e-07],
 [['concerned', 'outside', 'now'], 1.4810773635145633e-09]]
```

In [ ]: