On Indicator Functions

Let $\mathbf{1}_{i,d}$ be an indicator function on preference lists which is 1 when the ith car is displaced by exactly d.

Note some helpful properties of this indicator function:

- Constant on cosets of the diagonal subgroup. This means that the Fourier transform will have at most n^m nonzero coefficients indexed by group elements summing to $0 (\text{mod } n)$ (see 6/12 pg 3ish) (ie $\text{supp}(\widehat{\mathbf{1}_{i,d}}) \subset \{g \in C_n^m \mid \sum_{j=0}^m g_j \equiv 0 \pmod{n}\}$)

- Since the behavior of the cars after the ith car has no bearing on the displacement of the ith car, the indices of the characters with non-zero fourier coefficient will have a 0 in the ith spot. (ie $\text{supp}(\widehat{\mathbf{1}_{i,d}}) \subset \{g \in C_n^m \mid g_j = 0 \text{ for all } i < j \leq n\}$)

  Proof outline: (dimensional argument)
  If the preference of the jth car can never change $\mathbf{1}_{i,d}$, then $\mathbf{1}_{i,d}$ must be constant on cosets of the cyclic subgroup generated by g where $g_k = \begin{cases} 1 & k = j \\ 0 & else \end{cases}$. Note that χ_h where $h_j = 0$ satisfies this criteria. Since the space of functions constant on cosets of $\langle g \rangle$ is an n^{m-1} dimensional vector space, and the set of characters we identified as part of this space span an n^{m-1} dimensional space, these characters must span the whole space.

- Regardless of their order, the parking outcome of the first $i-1$ cars will be the same, so these indicator functions are invariant under the action of S_{i-1} on the first $i-1$ cars. This means that for $\sigma \in S_{i-1}$, $\mathbf{1}_{i,d}(g) = \mathbf{1}_{i,d}(\sigma \cdot g)$ and $\widehat{\mathbf{1}_{i,d}}(g) = \widehat{\mathbf{1}_{i,d}}(\sigma \cdot g)$ for all $g \in C_n^m$ where σ acts on the first $i-1$ cars. Proof outline is 2 steps.

  - Justify why the parking outcome of the first $i-1$ cars is order independent (see argument from first week of research - use pairwise transpositions as generators)
  - Justify why action of the symmetric group is the same on both the delta basis and the fourier basis (see pg1 6/14 for details)

- Note that if $d \geq i$, then $\mathbf{1}_{i,d}(g) = 0$ for all g since the most that the ith car could be displaced is $i-1$

What are functions of interest that can be written in terms of these indicator functions?

- Lucky: $\sum_{i=0}^m \mathbf{1}_{i,0}$
- Number of cars displaced exactly d: $\sum_{i=0}^m \mathbf{1}_{i,d}$
- Total displacement/area: $\sum_{d=0}^{n-1} \sum_{i=0}^m \mathbf{1}_{i,d}$
- The constant function (but written in an interesting way). Note that $\sum_{d=0}^{i-1} \mathbf{1}_{i,d} = \mathbf{1}$ since at worst, the ith car will have displacement $i-1$

Some further observations from computation

- In making choices around which orbits to ignore for linear independence, you can make these choices to play nicely with the indicator functions by prioritizing what orbits to keep based on the number of 0s that they have at the end. The result of constructing the basis in this way is that the basis vectors required to construct $\mathbf{1}_{i,d}$ will only be sums over orbits of S_i and S_{i-1} for the character indices. Additionally, no indicator function will ever need to use a singleton basis vector (one of the basis vectors in the new basis which is just the same as one in the old basis). These would be good to prove rather than just observing

More notes from computational observations

- It seems like if you want to decompose the function repeats, the result only involves 2 distinct non-zero values for the fourier coefficients. The coefficeint for the trivial character is $n^m\left(\frac{m-1}{n}\right) = (m-1)n^{m-1}$ (this can be proved using expected value of the function). The coefficient for any character which is 0 everywhere except for 2 adjacent values which sum to n is n^{m-1} (the summing to n part is because repeats is constant on cosets of the diagonal subgroup. I don't know why there are only 2 nonzero elements of the character and they are always next to eachother or why the value of the coefficient is n^{m-1}).

  In short, now I have a little conjecture: let S be a subset of the group C_n^m where for some i, $g_i + g_{i+1} = n$ and for all $j \neq i, i+1$, $g_j = 0$. Then

  $$repeats = (m-1)n^{m-1}\chi_0 + n^{m-1}\sum_{g\in S}\chi_g$$

  or in other words

  $$\widehat{repeats}(g) = \begin{cases} (m-1)n^{m-1} & g = 0 \\ n^{m-1} & g \in S \\ 0 & \text{otherwise} \end{cases}$$

- I was curious how defect decomposed, which is the same as asking how many cars passed the nth spot in the parking procedure. Kind of shockingly, defect decomposes using a Very similar set of coefficients as total displacement. It seems that the only difference in the set of non-zero coefficients for orbits of S_m is the additional inclusion of orbits for the all 0s with a single nonzero element. WHYYYY?

  Recall that this is surprising because total displacement is constant under reordering AND cosets of the diagonal subgroup. In my 6/12 journal, I counted the number of distinct non-zero fourier coefficients, and it turned out to be the sequence a003239, which grows more slowly than the total number of nonzero fourier coefficients.

```
In [503]: stat = CnmStat(3,3, stats.passed_k[-1])

In [504]: stat.print_by_value("basis")
6
-4.5-2.598076j       : S_3*002,
-4.5+2.598076j       : S_3*001,
-1.5-2.598076j       : S_3*111,
-1.5+2.598076j       : S_3*222,
3.+0.j               : S_3*021,
12.+0.j              : S_3*000,
6
```

```
In [13]: stats = IterateStats(3,3, True)
. . . loading from file . . .

In [14]: stat = CnmStat(3,3, stats.total_disp)

In [15]: stat.print_by_value("basis")
4
-4.5-7.794229j       : S_3*111,
-4.5+7.794229j       : S_3*222,
9.+0.j               : S_3*102,
36.+0.j              : S_3*000,
4
```

```
In [500]: stat = CnmStat(4,4, stats.passed_k[-1])

In [501]: stat.print_by_value("basis")
12
-32.-32.j            : S_4*0003,
-32.+0.j             : S_4*0020,
-32.+32.j            : S_4*1000,
-4.-16.j             : S_4*3333,
-4.-8.j              : S_4*0211,
-4.+0.j              : S_4*3113, S_4*2222,
-4.+8.j              : S_4*3320,
-4.+16.j             : S_4*1111,
4.+0.j               : S_4*2231,
12.+0.j              : S_4*2002,
20.+0.j              : S_4*0130,
156.+0.j             : S_4*0000,
13
```

```
In [10]: stats = IterateStats(4,4, True)
. . . loading from file . . .

In [11]: stat = CnmStat(4,4, stats.total_disp)

In [12]: stat.print_by_value("basis")
9
-16.-64.j            : S_4*3333,
-16.-32.j            : S_4*0112,
-16.+0.j             : S_4*1313, S_4*2222,
-16.+32.j            : S_4*2033,
-16.+64.j            : S_4*1111,
16.+0.j              : S_4*3122,
48.+0.j              : S_4*2020,
80.+0.j              : S_4*0130,
624.+0.j             : S_4*0000,
10
```

```
In [497]: stat = CnmStat(5,5, stats.passed_k[-1])

In [498]: stat.print_by_value("basis")
30
-312.5-430.11935j    : S_5*04000,
-312.5-101.537405j   : S_5*30000,
-312.5+101.537405j   : S_5*02000,
-312.5+430.11935j    : S_5*00100,
-26.18034-0.j        : S_5*11044,
-24.045085-54.124465j : S_5*01301,
-24.045085+54.124465j : S_5*42004,
-18.454915-40.022398j : S_5*10202,
-18.454915+40.022398j : S_5*40330,
-15.815595-45.042683j : S_5*40344,
-15.815595+45.042683j : S_5*12011,
-11.545085-10.633135j : S_5*11422,
-11.545085+10.633135j : S_5*34143,
-8.401699-20.307481j : S_5*22222,
-8.401699+20.307481j : S_5*33333,
-5.954915-6.571639j  : S_5*31231,
-5.954915+6.571639j  : S_5*42234,
-3.81966-0.j         : S_5*03322,
4.40983-4.061496j    : S_5*24333,
4.40983+4.061496j    : S_5*32122,
10.+0.j              : S_5*40312,
15.59017-17.204774j  : S_5*21444,
15.59017+17.204774j  : S_5*14113,
23.315595-1.551354j  : S_5*33301,
23.315595+1.551354j  : S_5*24220,
97.188471+0.j        : S_5*02030,
103.401699-86.02387j : S_5*11111,
103.401699+86.02387j : S_5*44444,
197.811529-0.j       : S_5*00014,
2360.+0.j            : S_5*00000,
30
```

```
In [7]: stat = CnmStat(5,5, stats.total_disp)

In [8]: stat.print_by_value("basis")
26
-130.901699-0.j       : S_5*10414,
-120.225425-270.622324j : S_5*01301,
-120.225425+270.622324j : S_5*40204,
-92.274575-200.111989j : S_5*12002,
-92.274575+200.111989j : S_5*43003,
-79.077974-225.213416j : S_5*40443,
-79.077974+225.213416j : S_5*10121,
-57.725425-53.165676j : S_5*24121,
-57.725425+53.165676j : S_5*34314,
-42.008497-101.537405j : S_5*22222,
-42.008497+101.537405j : S_5*33333,
-29.774575-32.858195j : S_5*32311,
-29.774575+32.858195j : S_5*42342,
-19.098301-0.j        : S_5*03223,
22.04915-20.307481j   : S_5*34332,
22.04915+20.307481j   : S_5*23212,
50.+0.j               : S_5*13420,
77.95085-86.02387j    : S_5*44214,
77.95085+86.02387j    : S_5*14131,
116.577974-7.756768j  : S_5*03313,
116.577974+7.756768j  : S_5*24202,
485.942353+0.j        : S_5*02003,
517.008497-430.11935j : S_5*11111,
517.008497+430.11935j : S_5*44444,
989.057647+0.j        : S_5*40001,
11800.+0.j            : S_5*00000,
26

In [9]: stats = IterateStats(5,5, True)
. . . loading from file . . .
```

Some notes for n where my nice basis conversion fails:

Also, for $n = m = 6$, the number of distinct non-zero fourier coefficients was 72 for total displacement and 77 for defect. Note that 72 is 8 smaller than the total possible number of distinct nonzero coefficients.

For defect the following coefficents are causing the additinal collapsing from 80 to 72

- 222000 and 555300 have the same fourier coefficient
- 444000 and 111300 have the same fourier coefficient

- 444222 and 333324 have the same fourier coefficient
- 332220 and 333555 have the same fourier coefficient
- 444330 and 111333 have the same fourier coefficient
- 433200 and 544221 have the same fourier coefficient
- 333333 and 001245 have the same fourier coefficient
- 123345 has a coefficient of 0

Also, for $n = m = 7$ the number of distinct nonzero coefficients for total displacement is 246 (exactly the maximum possible number of distinct non-zero coefficeints for something constant on cosets of D and orbits of S_m). And the number of distinct nonzero coefficients for defect is 252 which is exactly $246 + 6$. This leads to the **very suspicious** hypothesis that for prime $n = m$, the number of nonzero coefficients for total displacement achieves it's maximum bound, and for composite $n = m$, there are some overlaping orbits (which ones you might ask? great question).

Another observation regarding the decomposition of defect

For all of the computations that I have looked at, the additional coefficients for defect come from orbits of the form $g0 \cdots 0$ where $g \in C_n$. And their values are all relatively large negative coefficients. And they all have the same real part.

That made me look more closely at their real part, and I realized that for defect, the real part was $-\frac{n^{n-1}}{2}$. For $n = 3, 4, 5, 6, 7$. Then I was looking at the complex part, and I happened to plot the arctangent of the ratio of the complex part to the real part it because I was curious what was going on with the phase. The change in phase is constant for every step up in the nonzero group element. That's also surprising.

- Since I was looking at defect, I was looking at statistics regarding the number of cars who attempted to park in a particular spot and moved on to the next spot. Notably, these statistics had the same fourier coefficients for all of the fourier coefficients which were indexed by group elements summing to 0. For a moment, let $s_i, d \in \mathbb{C}C_n^m$. Let s_i be the number of cars who attempted to park in the ith spot and moved on to the next spot, and let d be the total displacement. Each of the s_is were expressible as d/n plus a linear combination of characters with exactly 1 nonzero entry where that non-zero entry determines the coefficient. This is surprising.

  Written more symbolically, the little conjecture is that

$$s_i = \frac{d}{n} + \sum_{i=1}^{n-1} \hat{s}_i(g_i)(\chi_{S_m \cdot g_i})$$

  where g_i is a group element which has i for the first componenet and 0 for the remaining $m - 1$ components. And the character indexed by the orbit of S_m is really just a sum of all of the characters indexed by group elements which are a part of that orbit

  This means that to understand the number of cars that pass a given spot, you really only need to understand total displacement for linear probing and an additional $n - 2$ values: $\hat{s}_i(g_i)$.

Indicator functions - part 2 - parking lot's perspective

Let $\mathbf{1'}_{i,s} \in \mathbb{C}C_n^m$ be an indicator function on preference lists which is 1 when the s spot was passed by the ith car.

Some properties:

- This indicator function behaves nicely with the action of the diagonal subgroup. Let $(d, \cdots, d) \in D \subset C_n^m$. Then

$$(d, \cdots, d)\mathbf{1'}_{i,s} = \mathbf{1'}_{i,ds}$$

- Just as before, since the behavior of the cars after c doesn't matter at all, there is a constraint on the indices of the nonzero Fourier coefficients.

$$\mathrm{supp}(\widehat{\mathbf{1'}_{i,s}}) \subset \{g \in C_n^m | g_j = 0 \text{ for all } i < j \leq n\}$$

- Similar to before, the parking result of the first cars doesn't depend on the order of those cars, so $\mathbf{1'}_{i,s}$ is constant on orbits of S_{i-1} where S_{i-1} acts on the first i cars.

Some uses:

- Defect : $\sum_{i=1}^{m} \mathbf{1'}_{i,n}$
- Relatives of Defect. Let $g = (1, \cdots, 1)$ be the generator for the diagonal subgroup. Then let d_k be a function on preference lists where $d_k(p)$ is the defect of $g^k p$. Then

$$d_k = \sum_{i=1}^{m} \mathbf{1'}_{i,n-k}$$

- Total Displacement : $\sum_{s=1}^{m} \mathbf{1'}_{m,s}$

I wonder if this could help explain the connections between defect and total displacement that I found experimentally...

There were a lot of little conjectures mixed in with observations, so I'm just going to make a list of the ones I don't know how to show and questions that showed up along the way. (The theorem type statements that I do know how to show are on page 1 and 5 and are about indicator functions of various types)

Conjectures/questions

- To construct $\mathbf{1}_{i,d}$ using the modified basis where the choice of which orbit of S_i to remove to achieve linear independence is made as in my code currently implements it (by prioritizing keeping orbits whose characters end with 0s), then $\mathbf{1}_{i,d}$ will be a linear combination of orbits of S_i and S_{i-1}. (in answering this question please please come up with better/more concise mathematical notation for the modified basis, I'm getting tired of typing out wordy sentences that are only half explanations probably only readable to me)

- Given free choice for which orbits of S_i to remove to achieve linear independence, what is the best choice to minimize $\text{supp}(\mathbf{1}_{i,d})$? What if you are trying to make $\text{supp}(\mathbf{1}_{i,d})$ small for all i and d at the same time? (I think this leads you to my current implementation, but can you show that? Is there still choice? I think there is still choice, where the key thing is the number of trailing 0s, and the rest of picking the last in lexicographic order of the reverse string is just computational convenience, but there could be more optimization left in this space of *shrug* or I could just be wrong about the tailing 0s thing.)

- While I'm on questions about the modified basis. How many orbits of S_m are there for n spots? (relates to total multiplicity of tabloid representations in the S_m module decomposition of the full parking space, relates to A003239 for $n = m$, but we care about $n \neq m$) How many basis vectors do you start with when you just look at orbits of $S_1, \cdots S_m$ acting on each preference list? How many orbits do you end up throwing out to achieve linear independence? Can you write an expression using the combinatorial quantities involved in constructing the basis which is also a combinatorial proof that this number is n^m?

- **Repeats** conjecture (pg 2). Let S be the subset of C_n^m where for some i, $g_i + g_{i+1} = n$ and for all $j \neq i, i+1$, $g_j = 0$. Then

$$\widehat{repeats}(g) = \begin{cases} (m-1)n^{m-1} & g = 0 \\ n^{m-1} & g \in S \\ 0 & \text{otherwise} \end{cases}$$

- Let d be the total displacement statistic. I suspect that for prime $n = m$, the number of distinct nonzero fourier coefficients attains the maximum number of distinct nonzero fourier coefficients for a function which is constant on cosets of D and constant on orbits of S_m.

  This comes with further questions. For composite $n = m$, it seems that this is only off by a little bit. Can you identify which orbits of S_m for the fourier coefficients summing to 0 will be the same? Or 0?

- **Defect** and extensions conjecture. As before, let d be the total displacement statistic. Let s_k be the number of cars who passed the kth spot statistic. Let the group element $e_1 = (1, 0, \cdots, 0) \in C_n^m$. Then for $n = m$, it seems that

$$\begin{aligned} s_k &= \frac{d}{n} + \sum_{i=1}^{n-1} \widehat{s_k}(ie_1)\, \chi_{S_m \cdot ie_1} \\ &= \frac{d}{n} + \sum_{i=1}^{n-1} \sum_{j=1}^{m} \widehat{s_k}(ie_1)\, \chi_{ie_j} \end{aligned}$$

Further, specifically in the case of s_n which is defect, it seems that there is an explicit expression for $\widehat{s_n}(je_i)$

$$\widehat{s_n}(ie_1) = -\frac{n^{m-1}}{2} + \frac{n^{m-1}}{2\tan\left(\pi\left(\frac{i}{n} - \frac{1}{2}\right)\right)}$$

Similarly it seems that

$$\widehat{s_1}(ie_1) = \frac{n^{m-1}}{2} + \frac{n^{m-1}}{2\tan\left(\pi\left(\frac{i}{n} - \frac{1}{2}\right)\right)}$$

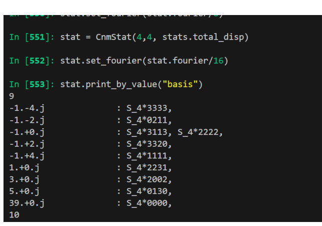
And possibly it seems that

$$\mathrm{Re}(\widehat{s_k}(je_1)) = \mathrm{Re}(\widehat{s_{n+1-k}(je_1)})$$

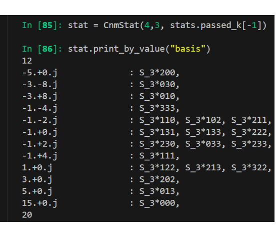
I wonder if there is a nice explicit expression for $\hat{s}_k(ie_1)$? And WHYYYYYY???

Possible directions to show these things: Are there properties of $\mathbf{1'}_{i}, s$ that you can leverage? Can you show that for $g \in C_n^m$ with $\sum_{i=1}^m g_i \equiv 0 (\mathrm{mod}\ n)$ that $\widehat{s_k}(g) = \widehat{s'_k}(g)$ (then showing the d/n part is workable using the fact that $\sum_i s_i = d$)? Can you find a clever way to explicitly compute $\widehat{s_k}(ie_1) = \langle s_k, \chi_{ie_1} \rangle$ (make sure the inner product isn't backwards)?

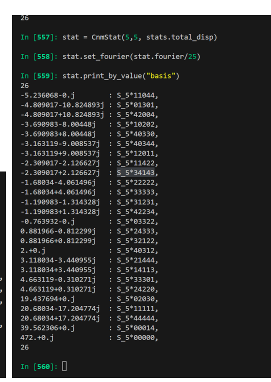
Also potentially interesting is figuring out how this works for $n \neq m$. It seems that defect still behaves somewhat nicely in its relation to displacement for n spots and n cars for most of the coefficeints and to displacement for n spots and m cars for the trivial character, and the second half of the expression seems to stay the same. But the d/n of this expression seems to become more chaotic for $k \neq n$. See screen shots



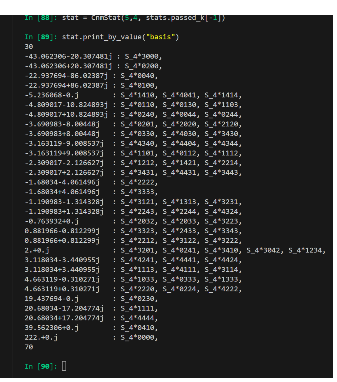
```
In [551]: stat = CnmStat(4,4, stats.total_disp)

In [552]: stat.set_fourier(stat.fourier/16)

In [553]: stat.print_by_value("basis")
9
-1.-4.j             : S_4*3333,
-1.-2.j             : S_4*0211,
-1.+0.j             : S_4*3113, S_4*2222,
-1.+2.j             : S_4*3320,
-1.+4.j             : S_4*1111,
1.+0.j              : S_4*2231,
3.+0.j              : S_4*2002,
5.+0.j              : S_4*0130,
39.+0.j             : S_4*0000,
10
```



```
In [85]: stat = CnmStat(4,3, stats.passed_k[-1])

In [86]: stat.print_by_value("basis")
12
-5.+0.j              : S_3*200,
-3.-8.j              : S_3*030,
-3.+8.j              : S_3*010,
-1.-4.j              : S_3*333,
-1.-2.j              : S_3*110, S_3*102, S_3*211,
-1.+0.j              : S_3*131, S_3*133, S_3*222,
-1.+2.j              : S_3*230, S_3*033, S_3*233,
-1.+4.j              : S_3*111,
1.+0.j               : S_3*122, S_3*213, S_3*322,
3.+0.j               : S_3*202,
5.+0.j               : S_3*013,
15.+0.j              : S_3*000,
20
```



```
26

In [557]: stat = CnmStat(5,5, stats.total_disp)

In [558]: stat.set_fourier(stat.fourier/25)

In [559]: stat.print_by_value("basis")
26
-5.236068-0.j        : S_5*11044,
-4.809017-10.824893j : S_5*01301,
-4.809017+10.824893j : S_5*42004,
-3.690983-8.00448j   : S_5*10202,
-3.690983+8.00448j   : S_5*40330,
-3.163119-9.008537j  : S_5*40344,
-3.163119+9.008537j  : S_5*12011,
-2.309017-2.126627j  : S_5*11422,
-2.309017+2.126627j  : S_5*34143,
-1.68034-4.061496j   : S_5*22222,
-1.68034+4.061496j   : S_5*33333,
-1.190983-1.314328j  : S_5*31231,
-1.190983+1.314328j  : S_5*42234,
-0.763932-0.j        : S_5*03322,
0.881966-0.812299j   : S_5*24333,
0.881966+0.812299j   : S_5*32122,
2.+0.j               : S_5*40312,
3.118034-3.440955j   : S_5*21444,
3.118034+3.440955j   : S_5*14113,
4.663119-0.310271j   : S_5*33301,
4.663119+0.310271j   : S_5*24220,
19.437694+0.j        : S_5*02030,
20.68034-17.204774j  : S_5*11111,
20.68034+17.204774j  : S_5*44444,
39.562306+0.j        : S_5*00014,
472.+0.j             : S_5*00000,
26

In [560]:
```



```
In [88]: stat = CnmStat(5,4, stats.passed_k[-1])

In [89]: stat.print_by_value("basis")
30
-43.062306-20.307481j : S_4*3000,
-43.062306+20.307481j : S_4*0200,
-22.937694-86.02387j : S_4*0040,
-22.937694+86.02387j : S_4*0100,
-5.236068-0.j         : S_4*1410, S_4*4041, S_4*1414,
-4.809017-10.824893j : S_4*0110, S_4*0130, S_4*1103,
-4.809017+10.824893j : S_4*0240, S_4*0044, S_4*0244,
-3.690983-8.00448j   : S_4*0201, S_4*2020, S_4*2120,
-3.690983+8.00448j   : S_4*0330, S_4*4030, S_4*3430,
-3.163119-9.008537j  : S_4*4340, S_4*4404, S_4*4344,
-3.163119+9.008537j  : S_4*1101, S_4*0112, S_4*1112,
-2.309017-2.126627j  : S_4*1212, S_4*1421, S_4*2214,
-2.309017+2.126627j  : S_4*3431, S_4*4431, S_4*3443,
-1.68034-4.061496j   : S_4*2222,
-1.68034+4.061496j   : S_4*3333,
-1.190983-1.314328j  : S_4*3121, S_4*1313, S_4*3231,
-1.190983+1.314328j  : S_4*2243, S_4*2244, S_4*4324,
-0.763932+0.j        : S_4*2032, S_4*2033, S_4*3223,
0.881966-0.812299j   : S_4*3323, S_4*2433, S_4*3343,
0.881966+0.812299j   : S_4*2212, S_4*3122, S_4*3222,
2.+0.j               : S_4*3201, S_4*0241, S_4*3410, S_4*3042, S_4*1234,
3.118034-3.440955j   : S_4*4241, S_4*4441, S_4*4424,
3.118034+3.440955j   : S_4*1113, S_4*4111, S_4*3114,
4.663119-0.310271j   : S_4*1033, S_4*0333, S_4*1333,
4.663119+0.310271j   : S_4*2220, S_4*0224, S_4*4222,
19.437694-0.j        : S_4*0230,
20.68034-17.204774j  : S_4*1111,
20.68034+17.204774j  : S_4*4444,
39.562306+0.j        : S_4*0410,
222.+0.j             : S_4*0000,
70

In [90]:
```

To Do

- Work on writing about sampling from defect d preference lists
- There's a whole list of conjectures! Lots of figuring out why things are true
- Can the first type of indicator function help with explicitly computing coefficients for the displaced by i family of functions?

More directions (maybe focus on the first things first though)

- Probably some more patterns to find if you look closely at other statistics. leading element/matches ith preference (should be on the simpler side)? longest prime? last start point? Anything for not circular parking (probably not nice)? More options (slightly more implementation required): relating to permutations, number of break points, average prime segment length?
- Use more detailed information about Fourier transforms to make statements about random walks? (Potentially the averaging trick for expected value for walks which treat all coordinates the same (ie drop all coefficients in the modified basis which don't correspond to orbits of S_m) in combination with the unproved observation about $\mathbf{1}_{i,d}$ only needing orbits of S_i and S_{i-1} could be helpful)
- What do the matrix representations of some elements of $C_n \wr S_m$ look like for the basis that you constructed? Basis is helpful, but very not orthonormal... What if it was? Can you enforce that in a way which respects the groups nicely? Can you use the algorithm prof O mentioned for if you already know the representation (do you know the representation?)?