6.S966: Exam 3, Spring 2025

Do not tear exam booklet apart!

- This is a closed book exam. One page (8 1/2 in. by 11 in.) of notes, front and back, are permitted. Calculators are not permitted.
- The total exam time is 1 hour and 50 minutes.
- The problems are not necessarily in any order of difficulty.
- Record all your answers in the places provided. If you run out of room for an answer, continue on a blank page and mark it clearly.
- If a question seems vague or under-specified to you, make an assumption, write it down, and solve the problem given your assumption.
- If you absolutely *have* to ask a question, come to the front.
- **Write your name on every piece of paper.**

Name: _____ MIT Email: _____

Question	Points	Score
1	30	
2	15	
3	55	
Total:	100	

Name: _____

Is it equivariant?

1. (30 points) Determine whether the following proposed operations are **equivariant** with respect to the group $SO(3)$ or not.

   (a) Apply a ReLU (`max(x, 0)`) to spherical harmonic coefficients transforming as $\bigoplus_{\ell=0}^{\ell_{\max}} \ell$, representing a scalar function on S^2. Is this operation equivariant under $SO(3)$?

   ◯ Equivariant

   ◯ Not equivariant

   Explain your answer:

   (b) Apply a ReLU (`max(x, 0)`) pointwise to a scalar signal on S^2. Is this operation equivariant under $SO(3)$?

   ◯ Equivariant

   ◯ Not equivariant

   Explain your answer:

   (c) Multiply two scalar functions on S^2 pointwise. Assume both rotate together under $SO(3)$. Is the result equivariant?

   ◯ Equivariant

   ◯ Not equivariant

   Explain your answer:

Name: _____

(d) Take the tensor product of spherical harmonic coefficients with themselves, where the coefficients transform as $\bigoplus_{\ell=0}^{\ell_{\max}} \ell$. Is this operation equivariant under $SO(3)$?

◯ Equivariant

◯ Not equivariant

Explain your answer:

(e) Multiply matching coefficients from a direct sum of spherical harmonics (e.g., $c_{\ell=1,m=0} \times c_{\ell=1,m=0}$). Is the result equivariant?

◯ Equivariant

◯ Not equivariant

Explain your answer:

(f) Given two copies of the same 1D irrep A of a group G, scale one by a and the other by b. Does this preserve equivariance?

◯ Equivariant

◯ Not equivariant

Explain your answer:

Name: _____

(g) Given a 2D irrep E of a group G, multiply the first component by a and the second by b. Is this operation equivariant?

◯ Equivariant

◯ Not equivariant

Explain your answer:

(h) Given two copies of a 2D irrep E of a group G, scale each copy by a scalar multiple of the identity (`a * np.eye(2)` and `b * np.eye(2)`), then sum them. Does this preserve equivariance?

◯ Equivariant

◯ Not equivariant

Explain your answer:

Name: _____

Vibrational Modes of Simple Lattices

2. (15 points) In class, we analyzed vibrational modes in point-symmetric objects. We now extend this to *periodic systems* with translational symmetry, analyzing vibrational modes in simple lattices.

   (a) Consider a 1D periodic lattice with identical atoms spaced by $\mathbf{a} = a\hat{x}$. The translation symmetries of this lattice consists of shifts by integer multiples of $\mathbf{a}$. The translation irreps are 1D (tranlations are Abelian) and indexed by a continuous label $k \in [-\pi/a, \pi/a)$. These irreps are defined by

   $$\rho_k(T_n) = e^{ikna}.$$

   where n is an integer (positive, negative or zero). Briefly explain why the irreps are labeled by values in the range $k \in [-\pi/a, \pi/a)$ (called the **first Brillouin zone**). *Hint: Let $k' = k + 2\pi/a$, where $k \in [-\pi/a, \pi/a)$. Expand $e^{ik'na}$.*

   (b) The 3D simple cubic lattice has identical atoms at positions defined by integer multiples of the lattice lattice vectors $(\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3) = (a\hat{x}, a\hat{y}, a\hat{z})$. Its vibrational modes are labeled by wavevector $\vec{k} \in [-\pi/a, \pi/a)^3$, i.e. inside the first *Brillouin zone* of the reciprocal lattice with lattice vectors $(\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3) = (\frac{2\pi}{a}\hat{x}, \frac{2\pi}{a}\hat{y}, \frac{2\pi}{a}\hat{z})$. The modes transform under the *little group* at $\vec{k} = K$, the subgroup of O_h satisfying

   $$K^{\text{vec}}(g)\vec{k} = \vec{k} + n_1\mathbf{b}_1 + n_2\mathbf{b}_2 + n_3\mathbf{b}_3.$$

   That is, symmetries that leave $\vec{k}$ invariant modulo a reciprocal lattice vector.
   At fixed $\vec{k} = K$, vibrational modes transform under

   $$K^{\text{perm}}(g) \otimes K^{\text{vec}}(g),$$

   where K^{perm} acts on atom positions and K^{vec} on 3D displacements. With one atom per unit cell in the 3D cubic lattice, there are 3 degrees of freedom per $\vec{k}$. Rigid motions are not subtracted, as they do not correspond to irreps at nonzero $\vec{k}$.

   At $\vec{k}_\Gamma = (0, 0, 0)$, the little group is all of O_h. The atom is fixed by all symmetries, so $\Gamma^{\text{perm}}(g) = A_{1g}$, and $\Gamma^{\text{vec}}(g) = T_{1u}$. What is the irrep decomposition of the vibrational modes at $\vec{k}_\Gamma$? Explain your reasoning.

(c) Vibrational modes belonging to different dimensions of the same copy of a given irrep must have the same "energy". How many distinct energies do the vibrational modes of the simple cubic lattice have at $\vec{k}_\Gamma = (0,0,0)$? Explain your reasoning.

(d) You will analyze how the vibrational modes of the simple cubic lattice transform under the *little group* at the following $\vec{k}$-points:

- $\vec{k}_Y = \frac{\pi}{a}(0,1,0)$ — invariant under the subgroup D_{4h}
- $\vec{k}_\Sigma = \frac{\pi}{a}(u,u,0)$ with $0 < u < 1$ — invariant under C_{2v}

Using the character tables for D_{4h}, and C_{2v} provided in the reference section, determine how the 3 vibrational modes of the simple cubic lattice decompose into irreps at each $\vec{k}$. Based on this, how many distinct vibrational "energies" are possible at each point?

i. How do the 3 simple cubic vibrational modes transform at $\vec{k}_Y$ (invariant under D_{4h})? How many distinct energies can the vibrational modes of the simple cubic lattice have at $\vec{k}_Y$? Explain your reasoning.

ii. How do the 3 simple cubic vibrational modes transform at $\vec{k}_\Sigma$ (invariant under C_{2v})? How many distinct energies can the vibrational modes of the simple cubic lattice have at $\vec{k}_\Sigma$? Explain your reasoning.

Name: _____

The Tesseract

3. (55 points) The tesseract has had a surprisingly active pop culture career — appearing in science fiction films like *Interstellar* (as a physical representation of time in higher dimensions) and superhero stories like *The Avengers* (as a mysterious cube-shaped energy source). But in mathematics, the tesseract is a 4D hypercube: a highly symmetric geometric object. In this problem, you'll use your knowledge of 2D square (D_4) and 3D cube (O_h) symmetries to construct and explore the discrete symmetry group of the tesseract — no special effects needed.

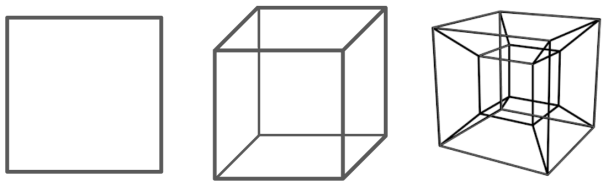

Figure 1: 2D Square, 3D Cube, 4D Tesseract

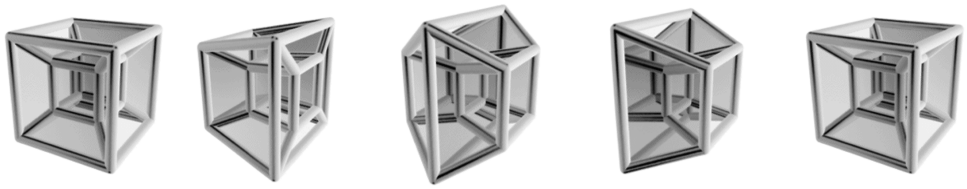

Figure 2: 3D projection of a tesseract while performing a simple rotation about a plane which bisects the figure from front-left to back-right and top to bottom. Source: Wikipedia, created by Jason Hise.

Name: _____

(a) For all code snippets below, you can assume the following libraries have been imported

```
1 import numpy as np
2 import scipy.linalg
3 from symm4ml import groups, rep, so3
```

i. Recall that we can generate the 8 element symmetry group of the square D_4 a 2D 4-fold (90°) rotation and a 2D mirror across the x axis.

```
4  so2_gen = np.array([[0, 1], [-1., 0]])
5  rot_90_2D = scipy.linalg.expm(2 * np.pi / 4 * so2_gen)
6
7  square_gen = np.stack([
8      rot_90_2D,  # 90 degree rotation
9      np.diag([1, -1])  # 2D mirror across x-axis
10 ])
11
12 square_group = groups.generate_group(square_gen)
13 print("Square group 2D vector rep: ", square_group.shape)
14 >> Square group 2D vector rep: (8, 2, 2)
```

Can the 2D mirror across the x-axis (line 9) be generated using the $SO(2)$ generator? Explain your reasoning.

ii. Describe how the function `groups.generate_group` uses `square_gen` to generate `square_group`.

Name: _____

iii. The symmetry group of the cube (O_h, same as the octahedra) of 48 elements can be generated with the generators of the square and one additional rotation, a 4-fold (90° rotation) in the yz plane (whose normal vector is along the 3D x-axis) or a 4-fold (90° rotation) in the xz plane (whose normal vector is along the 3D y-axis).

```
15 rot_90_yz_3D = so3.axis_angle_to_matrix(np.array([1, 0, 0]), np.pi/2)
16 rot_90_xz_3D = so3.axis_angle_to_matrix(np.array([0, 1, 0]), np.pi/2)
17
18 square_gen_to_3D = rep.direct_sum(square_gen, np.ones([len(square_gen), 1, 1]))
19
20 # This works
21 cube_gen = np.concatenate([square_gen_to_3D, rot_90_yz_3D[np.newaxis]])
22 # So does this
23 cube_gen = np.concatenate([square_gen_to_3D, rot_90_xz_3D[np.newaxis]])
24
25 cube_group = groups.generate_group(cube_gen)
26 print("Cube group 3D vector rep: ", cube_group.shape)
27 >> Cube group 3D vector rep: (48, 3, 3)
```

How does the function `so3.axis_angle_to_matrix` use the given arguements to construct the matrices `rot_90_yz_3D` and `rot_90_xz_3D`? What representation does this function return and which generators are used to do so?

iv. Describe what is happening in line 18 (`square_gen_to_3d = ...`) and why it's necessary to use to do to use `square_gen` to generate the symmetry of the cube?

Name: _____

v. We can follow an analogous procedure to generate the symmetry group of the tesseract (4D cube) B_4 with 384 elements. In this case, we need to add one additional generator corresponding to single plane 4-fold rotation.

```
28 def so4_generators():
29     ...
30
31 so4_gen = so4_generators() # Get the 6 generators
32 # Assume 4D coords are (x, y, z, w)
33 so4_gen_labels = ['Lxy', 'Lxz', 'Lxw', 'Lyz', 'Lyw', 'Lzw']
34
35 rot_90_4D = scipy.linalg.expm(np.pi/2 * so4_gen[so4_gen_labels.index(???)])
36
37 cube_gen_to_4d = rep.direct_sum(cube_gen, np.ones([len(cube_gen), 1, 1]))
38
39 tesseract_gen = np.concatenate([cube_gen_to_4d, rot_90_4D[np.newaxis]])
40
41 tesseract_group = groups.generate_group(tesseract_gen)
42 print("Tesseract group 4D vector rep: ", tesseract_group.shape)
43 >> Tesseract group 4D vector rep: (384, 4, 4)
```

Based on the procedure we used above to generate the symmetry of the cube from the symmetry of the square, give the labels of the three $SO(4)$ generators (three strings in `so4_gen_labels`) that we could plug into the `???`s in line 35 to generate the group of the tesseract. Explain your reasoning.

(b) Summarizing from above:

- D_4 is the symmetry group of the square in 2D (order 8),
- O_h is the symmetry group of the cube in 3D (order 48),
- B_4 is the symmetry group of the tesseract in 4D (order 384).

i. How does one construct the left cosets of a group G with respect to a subgroup H? Give the **number of cosets** and the **number of elements per coset** in terms of the size of G and H, i.e. $|G|$ and $|H|$.

ii. D_4 partitions the elements of the group O_h into 6 distinct cosets $|O_h|/|D_4| = 6$. Consider the square as one face of a cube. What do these cosets represent geometrically? Explain your reasoning.

iii. The tesseract is alternatively called the 8-cell because it can be thought of as built from 8 cubes. $|B_4|/|O_h| = 8$. What do these cosets represent geometrically?

Name: _____

(c) If we try to get irreps of B_4 using our function `rep.infer_irreps` our `python` kernel quickly runs out of memory. Let's see why.

i. If $|B_4|$ is 384, what are the 3 dimensions of regular representation of $|B_4|$?

ii. At some point, `rep.infer_irreps` calls `linalg.infer_change_of_basis(reg_rep, reg_rep)` where `reg_rep` is the regular representation. What is the shape (dimensions) of each matrix produced when `linalg.infer_change_of_basis` computes the **Kronecker sum** of each element of `reg_rep` (`reg_rep[i]` for `i` in `range(384)`) with itself? Explain your reasoning.

iii. Suppose we are using `float32` numbers (4 bytes) to represent this matrix. Note that $100^4 \times 4$ bytes $= 400 \times 10^6$ bytes $= 400$ megabytes (MB). Give a rough estimate (within an order of magnitude) for how much memory in gigabytes (GB) $= 10^9$ bytes each of these matrices takes to store.

(d) To compute the irreps of B_4, Prof. Smidt instead created a new function that takes inspiration from `lie.infer_irreps_from_tensor_products` to create a new function `infer_irreps_from_tensor_products_of_vector_rep` that uses the vector matrix representation rather than generators.

i. Why can't we use `lie.infer_irreps_from_tensor_products` directly to get the irreps of B_4?

ii. B_4 has 20 conjugacy classes. The Wonderful Orthogonality Theorem for Character tells us that the number of conjugacy classes directly tells us the number of irreps of a group. How many irreps must B_4 have? Explain how the Wonderful Orthogonality Theorem for Character gives this constraint.

Name: _____

(e) In Exam 2, you examined the irrep basis functions for the faces of the cube. The six basis functions spanned 3 distinct irreps of O_h: one 1D irrep, one 2D irrep and one 3D irrep. The irrep basis functions for the 8 cells of the tesseract look very analgous to these basis functions and also span 3 distinct irreps of B_4: one 1D irrep, one 3D irrep, and one 4D irrep. We give the plots of the cube face and tesseract cell coefficients side by side below.

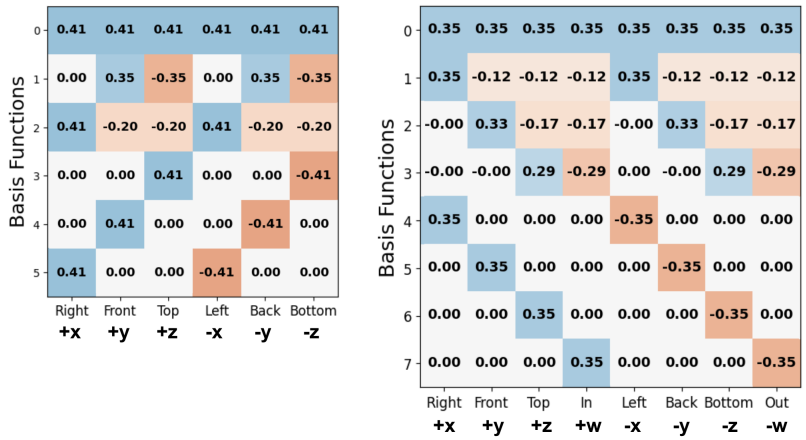

Figure 3: (Left) Cube face basis functions from Exam 2 (Right) Tesseract cell basis functions

i. If the 2D irrep basis functions 1 and 2 for the cube faces (Left) are proportional to the polynomials $y^2 - z^2$ and $2x^2 - y^2 - z^2$, respectively, what polynomials in (x, y, z, w) are the 3D irrep basis functions 1, 2, and 3 for the tesseract cells (Right) proportional to, according to the right plot above? Indicate which polynomial corresponds to each basis function.

ii. If the 3D irrep basis functions 3, 4, and 5 for the cube faces (Left) are proportional to the polynomials z, y, x, respectively, what polynomials in (x, y, z, w) are the 4D irrep basis functions 4, 5, 6, and 7 for the tesseract cells (Right) proportional to, according to the right plot above? Indicate which polynomial corresponds to each basis function.

Name: _____

symm4ml Docstring listing

Functions from modules listed in order: `groups, linalg, rep, lie, so3`

```
groups
----------------------
groups.generate_group:
Generate new group elements from matrices (group representations)
    Input:
        matrices: np.array of shape [n, d, d] of known elements
        decimals: int number of decimals to round to when comparing matrices
    Output:
        group: np.array of shape [m, d, d], where m is the size of the resultant
            group

linalg
----------------------
lie.infer_change_of_basis:
Compute the change of basis matrix from X1 to X2.
tip: Use the function nullspace
    Input:
        X1: an (n, d1, d1) array of n (d1, d1) matrices
        X2: an (n, d2, d2) array of n (d2, d2) matrices
    Output:
        Sols: An (m, d1, d2) array of m solutions.
        Each solution is a (d1, d2) matrix that satisfies X1 @ S = S @ X2,
        and together they form an orthognal basis for the set of solutions (under
            the inner product of the flattened versions).

rep
----------------------
rep.direct_sum
Computes direct sum of two representations.
    Input:
        rep1: np.array [n, d1, d1] representation of group. rep[i] is a matrix
            that
            represents i-th element of group.
        rep2: np.array [n, d2, d2] representation of group. rep[i] is a matrix
            that
            represents i-th element of group.
        You can assume that rep1 and rep2 are valid group representations.
    Output:
        Direct sum of representations. np.array [n, d1 + d2, d1 + d2].

rep.infer_irreps
```

```
Infers irreducible representations of group represented by multiplication table.
    Input:
        table: np.array [n, n] where table[i, j] = k means i * j = k.
    Output:
        Irreducible representations. List of np.array [n, d, d] where d is a
            dimension of irrep.
    Note: you can output the irreps in any order and in any basis.

lie
----------------------
lie.infer_irreps_from_tensor_products:
Infers irreducible representations from successive tensor products of a
    representation.
    Input:
        X: np.array [lie_dim, d, d] - generators of a representation.
        n: int - number of non-isomorphic irreducible representations to infer.
    Output:
        Ys: List[np.array] - list of n generators of irreducible representations,
            each an np.array of shape [lie_dim, d', d'] for some d'.
    """
    lie_dim, d, _ = X.shape


so3
----------------------
so3.axis_angle_to_matrix:
Convert an axis-angle representation to a rotation matrix.
    Input:
        axis: a unit vector representing the axis of rotation, np.array of shape
            [3]
        angle: the angle of rotation, in radians
    Output:
        R: a 3x3 rotation matrix, np.array [3, 3]
```

Name: _____

O_h	E	$8C_3$	$6C_2$	$6C_4$	$3C_2=(C_4)^2$	i	$6S_4$	$8S_6$	$3\sigma_h$	$6\sigma_d$	linear functions, rotations	quadratic functions
A_{1g}	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	-	$x^2+y^2+z^2$
A_{2g}	+1	+1	-1	-1	+1	+1	-1	+1	+1	-1	-	-
E_g	+2	-1	0	0	+2	+2	0	-1	+2	0	-	$(2z^2-x^2-y^2, x^2-y^2)$
T_{1g}	+3	0	-1	+1	-1	+3	+1	0	-1	-1	(R_x, R_y, R_z)	-
T_{2g}	+3	0	+1	-1	-1	+3	-1	0	-1	+1	-	(xz, yz, xy)
A_{1u}	+1	+1	+1	+1	+1	-1	-1	-1	-1	-1	-	-
A_{2u}	+1	+1	-1	-1	+1	-1	+1	-1	-1	+1	-	-
E_u	+2	-1	0	0	+2	-2	0	+1	-2	0	-	-
T_{1u}	+3	0	-1	+1	-1	-3	-1	0	+1	+1	(x, y, z)	-
T_{2u}	+3	0	+1	-1	-1	-3	+1	0	+1	-1	-	-

D_{4h}	E	$2C_4$ (z)	C_2	$2C'_2$	$2C''_2$	i	$2S_4$	σ_h	$2\sigma_v$	$2\sigma_d$	linear functions, rotations	quadratic functions	cubic functions
A_{1g}	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	-	x^2+y^2, z^2	-
A_{2g}	+1	+1	+1	-1	-1	+1	+1	+1	-1	-1	R_z	-	-
B_{1g}	+1	-1	+1	+1	-1	+1	-1	+1	+1	-1	-	x^2-y^2	-
B_{2g}	+1	-1	+1	-1	+1	+1	-1	+1	-1	+1	-	xy	-
E_g	+2	0	-2	0	0	+2	0	-2	0	0	(R_x, R_y)	(xz, yz)	-
A_{1u}	+1	+1	+1	+1	+1	-1	-1	-1	-1	-1	-	-	-
A_{2u}	+1	+1	+1	-1	-1	-1	-1	-1	+1	+1	z	-	$z^3, z(x^2+y^2)$
B_{1u}	+1	-1	+1	+1	-1	-1	+1	-1	-1	+1	-	-	xyz
B_{2u}	+1	-1	+1	-1	+1	-1	+1	-1	+1	-1	-	-	$z(x^2-y^2)$
E_u	+2	0	-2	0	0	-2	0	+2	0	0	(x, y)	-	(xz^2, yz^2) (xy^2, x^2y), (x^3, y^3)

C_{2v}	E	C_2 (z)	$\sigma_v(xz)$	$\sigma_v(yz)$	linear functions, rotations	quadratic functions	cubic functions
A_1	+1	+1	+1	+1	z	x^2, y^2, z^2	z^3, x^2z, y^2z
A_2	+1	+1	-1	-1	R_z	xy	xyz
B_1	+1	-1	+1	-1	x, R_y	xz	xz^2, x^3, xy^2
B_2	+1	-1	-1	+1	y, R_x	yz	yz^2, y^3, x^2y

Name: _____

The six generators of $SO(4)$ can be written as:

$$L_1 = L_{(xy)} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad L_2 = L_{(xz)} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad L_3 = L_{(xw)} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \end{bmatrix}$$

$$L_4 = L_{(yz)} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad L_5 = L_{(yw)} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix} \quad L_6 = L_{(zw)} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

where L_i enumerates the 6 generators and $L_{(jk)}$ specifies the plane of rotation.

Name: _____

Addendum: The Hypercube Rule of Food

This is NOT part of the exam.

In case you're curious: our analysis of the "Cubed Rule of Food" from Exam 2 can be extended to the hypercube (tesseract). For reasons even she cannot explain, Prof. Smidt spent her Friday night doing exactly that.

Using the basis functions provided at the end of the "Tesseract" problem and the power spectrum of all binary signals on the hypercube, she identified 15 distinct categories of food / not-food.

These 15 categories break down as follows:

- 1 category each for signals with 0, 1, 7, and 8 filled cells (salad, toast, hyperquiche, and hypercalzone, respectively).
- Signals with 2 filled cells correspond to either sandwich or not-sandwich.
- 3-cell signals fall into 2 categories: taco or not-taco.
- 4-cell signals split into 3 categories: hyper-not-taco, not-sushi, or sushi.
- 5-cell signals yield 2 categories: hypertaco or quiche.
- 6-cell signals also yield 2 categories: something resembling a hyper-sandwich-taco or hyper-sushi.

The degenerate categories are visualized below in this order using the "Dalí cross." The central cube corresponds to the cell at $(0, 0, 0, +1)$; the bottom cube corresponds to $(0, 0, 0, -1)$.

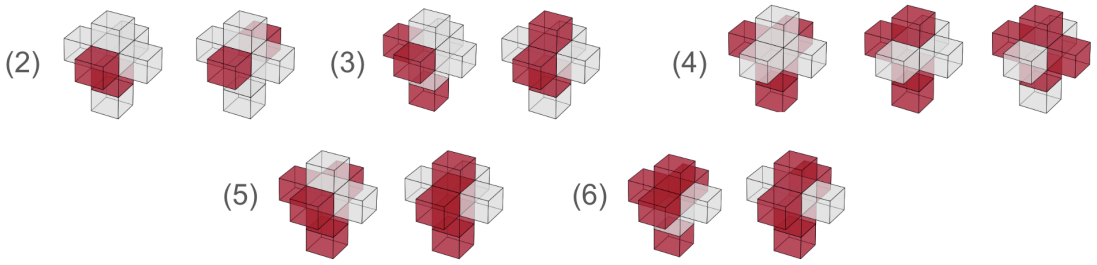

Figure 4: Distinct categories of hyperfood, organized by number of cells filled with structural hyperstarch.

Name: _____

Work space

Name: _____

Work space