EECS 16A Designing Information Devices and Systems I
Spring 2018 Homework 5

**All problems on this homework are practice, however all concepts covered here are fair game for the exam.**

1. **Sports Rank**

Every year in college sports, specifically football and basketball, debate rages over team rankings. The rankings determine who will get to compete for the ultimate prize, the national championship. However, ranking teams is quite challenging in the setting of college sports for a few reasons: there is uneven paired competition (not every team plays each other), there is a sparsity of matches (every team plays a small subset of all the teams available), and there is no well-ordering (team A beats team B who beats team C who beats A). In this problem, we will come up with an algorithm to rank the teams with real data drawn from the 2014 Associated Press (AP) poll of the top 25 college football teams.

Given $N$ teams we want to determine the rating $r_i$ for the $i^{th}$ team for $i = 1, 2, \ldots, N$, after which the teams can be ranked in order from highest to lowest rating. Given the wins and losses of each team, we can assign each team a score $s_i$.

$$s_i = \sum_j^N q_{ij} r_j,$$

where $q_{ij}$ represents the number of times team $i$ has beaten team $j$ divided by the number [1] of games played by team $i$. If we define the vectors $\vec{s} = \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_N \end{bmatrix}$, and $\vec{r} = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_N \end{bmatrix}$ we can express their relationship as a system of equations

$$\vec{s} = \mathbf{Q}\vec{r},$$

where $\mathbf{Q} = \begin{bmatrix} q_{11} & q_{12} & \cdots & q_{1N} \\ q_{21} & q_{22} & \cdots & q_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ q_{N1} & q_{N2} & \cdots & q_{NN} \end{bmatrix}$ is an $N \times N$ matrix.

(a) Consider a specific case where we have three teams, team A, team B, and team C. Team A beats team C twice and team B once. Team B beats team A twice and never beats team C. Team C beats team B three times. What is the matrix $\mathbf{Q}$?

**Solution:**

[1] We normalize by the number of games played to prevent teams from getting a high score by just repeatedly playing against weak opponents.

$$\mathbf{Q} = \begin{bmatrix} 0 & \frac{1}{5} & \frac{2}{5} \\ \frac{2}{6} & 0 & 0 \\ 0 & \frac{3}{5} & 0 \end{bmatrix}$$

(b) Returning to the general setting, if our scoring metric is good, then it should be the case that teams with better ratings have higher scores. Let's make the assumption that $s_i = \lambda r_i$ with $\lambda > 0$. Show that $\vec{r}$ is an eigenvector of $\mathbf{Q}$.

**Solution:**

With our assumption, we have $\vec{s} = \lambda \vec{r}$ and thus $\lambda \vec{r} = \mathbf{Q}\vec{r}$.

To find our rating vector, we need to find an eigenvector of $\mathbf{Q}$ with all nonnegative entries (ratings can't be negative) and a positive eigenvalue. If the matrix $\mathbf{Q}$ satisfies certain conditions (beyond the scope of this course), the dominant eigenvalue $\lambda_D$, i.e. the largest eigenvalue in absolute value, is positive and real. In addition, the dominant eigenvector, i.e. the eigenvector associated with the dominant eigenvalue, is unique and has all positive entries. We will now develop a method for finding the dominant eigenvector for a matrix if it is unique.

(c) Given $\vec{v}$, an eigenvector of $\mathbf{Q}$ with eigenvalue $\lambda$, and a real nonzero number $c$, express $\mathbf{Q}^n c\vec{v}$ in terms of $\vec{v}$, $c$,$n$, and $\lambda$.

**Solution:**

$$\mathbf{Q}^n c\vec{v} = \lambda^n c\vec{v}$$

This is because $\mathbf{Q}^n c\vec{v} = c\mathbf{Q}^n\vec{v} = c\mathbf{Q}^{n-1}\lambda\vec{v} = \lambda c\mathbf{Q}^{n-1}\vec{v} = \cdots = \lambda^{n-1}c\mathbf{Q}\vec{v} = \lambda^n c\vec{v}$.

(d) Now given multiple eigenvectors $\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_m$ of $\mathbf{Q}$, their eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_m$, and real nonzero numbers $c_1, c_2, \ldots, c_m$, express $\mathbf{Q}^n(\sum_{i=1}^m c_i\vec{v}_i)$ in terms of $\vec{v}$'s, $\lambda$'s, and $c$'s.

**Solution:**

First, we distribute $\mathbf{Q}$ to get

$$\sum_{i=1}^m \mathbf{Q}^n c_i\vec{v}_i.$$

From the previous part, we know that we can express each term in the summation with $\lambda_i^n c_i\vec{v}_i$, and thus we get

$$\sum_{i=1}^m \lambda_i^n c_i\vec{v}_i.$$

(e) Assuming that $|\lambda_1| > |\lambda_i|$ for $i = 2, \ldots, m$, argue or prove that

$$\lim_{n\to\infty} \frac{1}{\lambda_1^n}\mathbf{Q}^n \left(\sum_{i=1}^m c_i\vec{v}_i\right) = c_1\vec{v}_1.$$

*Hints*:

i. For sequences of vectors $\{\vec{a}_n\}$ and $\{\vec{b}_n\}$, $\lim_{n\to\infty}\left(\vec{a}_n + \vec{b}_n\right) = \lim_{n\to\infty}\vec{a}_n + \lim_{n\to\infty}\vec{b}_n$.

ii. For a scalar $w$ with $|w| < 1$, $\lim_{n\to\infty} w^n = 0$.

**Solution:**

From the previous part, we can conclude that

$$\frac{1}{\lambda_1^n}\mathbf{Q}^n\left(\sum_{i=1}^m c_i\vec{v}_i\right) = \frac{1}{\lambda_1^n}\sum_{i=1}^m \lambda_i^n c_i\vec{v}_i,$$

which may be rewritten as

$$c_1\vec{v}_1 + \sum_{i=2}^{m}\left(\frac{\lambda_i}{\lambda_1}\right)^n c_i\vec{v}_i,$$

where $\left|\frac{\lambda_i}{\lambda_1}\right| < 1$ for $i = 2,\ldots,m$. Therefore, $\lim_{n\to\infty}\left(\frac{\lambda_i}{\lambda_1}\right)^n = 0$ and

$$\lim_{n\to\infty}\frac{1}{\lambda_1^n}\mathbf{Q}^n\left(\sum_{i=1}^{m} c_i\vec{v}_i\right) = c_1\vec{v}_1.$$

(f) Now further assuming that $\lambda_1$ is positive, show that

$$\lim_{n\to\infty}\frac{\mathbf{Q}^n(\sum_{i=1}^m c_i\vec{v}_i)}{\|\mathbf{Q}^n(\sum_{i=1}^m c_i\vec{v}_i)\|} = \frac{c_1\vec{v}_1}{\|c_1\vec{v}_1\|}$$

*Hints:*

i. Divide the numerator and denominator by $\lambda_1^n$ and use the result from the previous part.

ii. For the sequence of vectors $\{\vec{a}_n\}$, $\lim_{n\to\infty}\|\vec{a}_n\| = \|\lim_{n\to\infty}\vec{a}_n\|$.

iii. For a sequence of vectors $\{\vec{a}_n\}$ and a sequence of scalars $\{\alpha_n\}$, if $\lim_{n\to\infty}\alpha_n$ is not equal to zero then the $\lim_{n\to\infty}\frac{\vec{a}_n}{\alpha_n} = \frac{\lim_{n\to\infty}\vec{a}_n}{\lim_{n\to\infty}\alpha_n}$.

**Solution:**

First, we use the hints and write the expression

$$\lim_{n\to\infty}\frac{\frac{1}{\lambda_1^n}\mathbf{Q}^n(\sum_{i=1}^m c_i\vec{v}_i)}{\frac{1}{\lambda_1^n}\|\mathbf{Q}^n(\sum_{i=1}^m c_i\vec{v}_i)\|}.$$

Using the fact that $\lambda_1$ is positive, we get

$$\lim_{n\to\infty}\frac{\frac{1}{\lambda_1^n}\mathbf{Q}^n(\sum_{i=1}^m c_i\vec{v}_i)}{\left\|\frac{1}{\lambda_1^n}\mathbf{Q}^n(\sum_{i=1}^m c_i\vec{v}_i)\right\|}.$$

Since the denominator does not converge to zero, we get

$$\lim_{n\to\infty}\frac{\frac{1}{\lambda_1^n}\mathbf{Q}^n(\sum_{i=1}^m c_i\vec{v}_i)}{\left\|\frac{1}{\lambda_1^n}\mathbf{Q}^n(\sum_{i=1}^m c_i\vec{v}_i)\right\|} = \frac{\lim_{n\to\infty}\frac{1}{\lambda_1^n}\mathbf{Q}^n(\sum_{i=1}^m c_i\vec{v}_i)}{\left\|\lim_{n\to\infty}\frac{1}{\lambda_1^n}\mathbf{Q}^n(\sum_{i=1}^m c_i\vec{v}_i)\right\|}.$$

Finally, using our result from the previous part, we obtain

$$\lim_{n\to\infty}\frac{\mathbf{Q}^n(\sum_{i=1}^m c_i\vec{v}_i)}{\|\mathbf{Q}^n(\sum_{i=1}^m c_i\vec{v}_i)\|} = \frac{c_1\vec{v}_1}{\|c_1\vec{v}_1\|}.$$

Let's assume that any vector $\vec{b}$ in $\mathbb{R}^N$ can be expressed as a linear combination of the eigenvectors of any square matrix $\mathbf{A}$ in $\mathbb{R}^{N\times N}$, i.e. $\mathbf{A}$ has $N$ rows and $N$ columns.

Let's tie it all together. Given the eigenvectors of $\mathbf{Q}$, $\vec{v}_1,\vec{v}_2,\ldots,\vec{v}_N$, we arbitrarily choose the dominant eigenvector to be $\vec{v}_1 = \vec{v}_D$. If we can find a vector $\vec{b} = \sum_{i=1}^m c_i\vec{v}_i$, such that $c_1$ is nonzero, then [2]

[2] If we select a vector at random, $c_1$ will be almost certainly non-zero.

$$\lim_{n\to\infty} \frac{\mathbf{Q}^n \vec{b}}{\|\mathbf{Q}^n \vec{b}\|} = \frac{c_1 \vec{v}_D}{\|c_1 \vec{v}_D\|}.$$

This is the idea behind the power iteration method, which is a method for finding the unique dominant eigenvector (up to scale) of a matrix whenever one exists. In the IPython notebook, we will use this method to rank our teams.

Note: For this application we know the rating vector (which will be the dominant eigenvector) has all positive entries, but $c_1$ might be negative resulting in our method returning a vector with all negative entries. If this happens, we simply multiply our result by $-1$.

(g) From the method you implemented in the IPython notebook, name the top five teams, the fourteenth team, and the seventeenth team.

**Solution:**

Oregon, Alabama, Arizona, Mississippi, UCLA, LSU, USC.

Here is an example of the code that could have been entered for the power iteration method:

```
b = np.dot(Q, b) / np.linalg.norm(np.dot(Q, b))
```

**2. The Dynamics of Romeo and Juliet's Love Affair**

In this problem, we will study a discrete-time model of the dynamics of Romeo and Juliet's love affair—adapted from Steven H. Strogatz's original paper, *Love Affairs and Differential Equations*, Mathematics Magazine, 61(1), p.35, 1988, which describes a continuous-time model.

Let $R[n]$ denote Romeo's feelings about Juliet on day $n$, and let $J[n]$ quantify Juliet's feelings about Romeo on day $n$. If $R[n] > 0$, it means that Romeo loves Juliet and inclines toward her, whereas if $R[n] < 0$, it means that Romeo is resentful of her and inclines away from her. A similar interpretation holds for $J[n]$, which represents Juliet's feelings about Romeo.

A larger $|R[n]|$ represents a more intense feeling of love (if $R[n] > 0$) or resentment (if $R[n] < 0$). If $R[n] = 0$, it means that Romeo has neutral feelings toward Juliet on day $n$. Similar interpretations hold for larger $|J[n]|$ and the case of $J[n] = 0$.

We model the dynamics of Romeo and Juliet's relationship using the following coupled system of linear evolutionary equations:

$$R[n+1] = a R[n] + b J[n], \quad n = 0, 1, 2, \ldots$$

and

$$J[n+1] = c R[n] + d J[n], \quad n = 0, 1, 2, \ldots,$$

which we can rewrite as

$$\vec{s}[n+1] = \mathbf{A} \vec{s}[n],$$

where

$$\vec{s}[n] = \begin{bmatrix} R[n] \\ J[n] \end{bmatrix}$$

denotes the state vector and

$$\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

the state transition matrix for our dynamic system model.

The parameters $a$ and $d$ capture the linear fashion in which Romeo and Juliet respond to their own feelings, respectively, about the other person. It's reasonable to assume that $a, d > 0$, to avoid scenarios of fluctuating day-to-day mood swings. Within this positive range, if $0 < a < 1$, then the effect of Romeo's own feelings about Juliet tend to fizzle away with time (in the absence of influence from Juliet to the contrary), whereas if $a > 1$, Romeo's feelings about Juliet intensify with time (in the absence of influence from Juliet to the contrary). A similar interpretation holds when $0 < d < 1$ and $d > 1$.

The parameters $b$ and $c$ capture the linear fashion in which the other person's feelings influence $R[n]$ and $J[n]$, respectively. These parameters may or may not be positive. If $b > 0$, it means that the more Juliet shows affection for Romeo, the more he loves her and inclines toward her. If $b < 0$, it means that the more Juliet shows affection for Romeo, the more resentful he feels and the more he inclines away from her. A similar interpretation holds for the parameter $c$.

All in all, each of Romeo and Juliet has four romantic styles, which makes for a combined total of sixteen possible dynamic scenarios. The fate of their interactions depends on the romantic style each of them exhibits, the initial state, and the values of the entries in the state transition matrix $\mathbf{A}$. In this problem, we'll explore a subset of the possibilities.

(a) Consider the case where $a+b = c+d$ in the state-transition matrix

$$\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}.$$

i. Show that

$$\vec{v}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

is an eigenvector of $\mathbf{A}$, and determine its corresponding eigenvalue $\lambda_1$. Also determine the other eigenpair $(\lambda_2, \vec{v}_2)$. Your expressions for $\lambda_1$, $\lambda_2$, and $\vec{v}_2$ must be in terms of one or more of the parameters $a$, $b$, $c$, and $d$.

**Solution:**

$$\begin{aligned} \mathbf{A}\begin{bmatrix} 1 \\ 1 \end{bmatrix} &= \begin{bmatrix} a+b \\ c+d \end{bmatrix} \\ &= (a+b)\begin{bmatrix} 1 \\ 1 \end{bmatrix} \\ &= (c+d)\begin{bmatrix} 1 \\ 1 \end{bmatrix} \end{aligned}$$

Let $\mu = a+b = c+d$. Then it's clear that $\begin{bmatrix} 1 & 1 \end{bmatrix}^T$ is an eigenvector of $\mathbf{A}$ corresponding to the eigenvalue $\mu$. Therefore, the following is an eigenpair of $\mathbf{A}$:

$$\left(\lambda_1 = a+b = c+d, \vec{v}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right)$$

To determine the other eigenpair $(\lambda_2, \vec{v}_2)$, we determine the other eigenvalue $\lambda_2$ first. We can do this in one of two ways:

**Method I:** Use the fact that $\text{tr}(\mathbf{A}) = \lambda_1 + \lambda_2$. We therefore have

$$\begin{aligned} a + d &= \lambda_1 + \lambda_2 \\ &= a + b + \lambda_2. \end{aligned}$$

Therefore,

$$\begin{aligned} \lambda_2 &= a + d - \lambda_1 \\ &= a + d - (a + b) \\ &= d - b. \end{aligned}$$

If we use the expression $\lambda_1 = c + d$, then an identical approach yields $\lambda_2 = a - c$.

**Method II:** An alternative approach is to determine the second eigenvalue $\lambda_2$ by solving the characteristic polynomial

$$\begin{aligned} \det(\lambda \mathbf{I} - \mathbf{A}) &= \det\left(\begin{bmatrix} \lambda - a & -b \\ -c & \lambda - d \end{bmatrix}\right) \\ &= (\lambda - a)(\lambda - d) - bc \\ &= \lambda^2 - (a + d)\lambda - bc \\ &= 0. \end{aligned}$$

We know from theory of quadratic polynomials that the sum of the roots equals the negative of the coefficient of the linear term $\lambda$. So, $\lambda_1 + \lambda_2 = a + d$. Notice that for a $2 \times 2$ matrix, the coefficient of $\lambda$ is $-\text{tr}(\mathbf{A})$. And we can now use the same steps of Method I from here on.

Once we have the second eigenvalue, we use it to build the matrix $\lambda_2 \mathbf{I} - \mathbf{A}$. However, we do this in a smart way. We use the expression $\lambda_1 = a - c$ for the first row, and $\lambda_1 = d - b$ for the second row. That is,

$$\begin{aligned} \lambda_2 \mathbf{I} - \mathbf{A} &= \begin{bmatrix} (a - c) - a & -b \\ -c & (d - b) - d \end{bmatrix} \\ &= \begin{bmatrix} -c & -b \\ -c & -b \end{bmatrix}. \end{aligned}$$

Clearly, $\lambda_2 \mathbf{I} - \mathbf{A}$ has linearly dependent columns, and the vector

$$\vec{v}_2 = \begin{bmatrix} b \\ -c \end{bmatrix}$$

lies in its nullspace. Therefore, we have our second eigenpair:

$$\left(\lambda_2 = a - c = d - b, \vec{v}_2 = \begin{bmatrix} b \\ -c \end{bmatrix}\right).$$

**Observation:** You should note that any matrix whose row sums are a constant, say $\mu$, must have $(\mu, \vec{1})$ as an eigenpair, where $\vec{1}$ is the all-ones vector of appropriate size.

ii. Consider the following state-transition matrix:

$$\mathbf{A} = \begin{bmatrix} 0.75 & 0.25 \\ 0.25 & 0.75 \end{bmatrix}$$

i. Determine the eigenpairs for this system.

**Solution:**

Notice that in this matrix, $a = d = 0.75$ and $b = c = 0.25$. So $\mu = a - c = d - b = 0.5$. Clearly, this is a row-stochastic matrix—each of its rows sums to 1. From the results of part (a)(i), we know that the eigenpairs of this matrix are

$$\left(\lambda_1 = 1, \vec{v}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right) \qquad \text{and} \qquad \left(\lambda_2 = 0.5, \vec{v}_2 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}\right).$$

**Observation:** Notice that the eigenvectors $\vec{v}_1$ and $\vec{v}_2$ are orthogonal. This is not a coincidence. It turns out that the eigenvectors of a symmetric matrix are mutually orthogonal.

ii. Determine all the *fixed points* of the system. That is, find the set of points such that if Romeo and Juliet start at, or enter, any of those points, they'll stay in place forever: $\{\vec{s}_* \mid \mathbf{A}\vec{s}_* = \vec{s}_*\}$. Show these points on a diagram where the $x$ and $y$-axes are $R[n]$ and $J[n]$.

**Solution:**

Any point along vector $\vec{s}_* = v_1 = \vec{1}$ is a fixed point, because $\vec{v}_1 = \vec{1}$ corresponds to the eigenvalue $\lambda_1 = 1$.

iii. Determine representative points along the state trajectory $\vec{s}[n]$, $n = 0, 1, 2, \ldots$, if Romeo and Juliet start from the initial state

$$\vec{s}[0] = \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

**Solution:**

The general solution is given by:

$$\begin{aligned} \vec{s}[n] &= \alpha_1 \lambda_1^n \vec{v}_1 + \alpha_2 \lambda_2^n \vec{v}_2 \\ &= \alpha_1 1^n \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \alpha_2 0.5^n \begin{bmatrix} 1 \\ -1 \end{bmatrix} \end{aligned}$$

Since $\vec{v}_1 \perp \vec{v}_2$ and since $\vec{s}[0] = \vec{v}_2$, we know that $\alpha_1 = 0$ and $\alpha_2 = 1$. Therefore,

$$\vec{s}[n] = 0.5^n \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

Since $0.5^n$ decays to zero as $n \to \infty$, the state trajectory stays along the second eigenvector and decays to the origin:

$$\lim_{n\to\infty} (R[n], J[n]) = (0,0)$$

In particular, the state vector obeys the following trajectory:

$$\begin{bmatrix} R[n] \\ J[n] \end{bmatrix} = \begin{bmatrix} \left(\frac{1}{2}\right)^n \\ -\left(\frac{1}{2}\right)^n \end{bmatrix}, \quad n = 0, 1, 2, \ldots$$

This means that, ultimately, Romeo and Juliet will become neutral to each other.

iv. Suppose the initial state is $\vec{s}[0] = \begin{bmatrix} 3 & 5 \end{bmatrix}^T$. Determine a reasonably simple expression for the state vector $\vec{s}[n]$. Find the limiting state vector

$$\lim_{n\to\infty} \vec{s}[n].$$

**Solution:**

We must express the initial state vector as a linear combination of the eigenvectors. That is, we must solve the system of linear equations

$$\begin{bmatrix}\vec{v}_1 & \vec{v}_2\end{bmatrix}\vec{\alpha} = \vec{s}[0]$$
$$\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}\alpha_1\\ \alpha_2\end{bmatrix} = \begin{bmatrix}3\\ 5\end{bmatrix}.$$

It's straightforward to find the solution:

$$\vec{\alpha} = \begin{bmatrix}\alpha_1\\ \alpha_2\end{bmatrix} = \begin{bmatrix}4\\ -1\end{bmatrix}.$$

Therefore, the state vector is given by

$$\begin{aligned}\vec{s}[n] &= \alpha_1\,\lambda_1^n\,\vec{v}_1 + \alpha_2\,\lambda_2^n\,\vec{v}_2\\ &= 4\begin{bmatrix}1\\ 1\end{bmatrix} - \left(\frac{1}{2}\right)^n\begin{bmatrix}1\\ -1\end{bmatrix}\\ &= \begin{bmatrix}4 - \left(\frac{1}{2}\right)^n\\ \\ 4 + \left(\frac{1}{2}\right)^n\end{bmatrix}\end{aligned}$$

Clearly,

$$\lim_{n\to\infty}\vec{s}[n] = \begin{bmatrix}4\\ 4\end{bmatrix}.$$

(b) Consider the setup in which

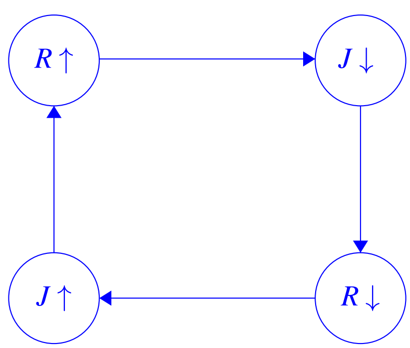
$$\mathbf{A} = \begin{bmatrix}a & b\\ c & d\end{bmatrix} = \begin{bmatrix}0 & 1\\ -1 & 0\end{bmatrix}.$$

In this scenario, if Juliet shows affection toward Romeo, Romeo's love for her increases, and he inclines toward her. The more intensely Romeo inclines toward her, the more Juliet distances herself. The more Juliet withdraws, the more Romeo is discouraged and retreats into his cave. But the more Romeo inclines away, the more Juliet finds him attractive and the more intensely she conveys her affection toward him. Juliet's increasing warmth increases Romeo's interest in her, which prompts him to incline toward her—again!

Predict the outcome of this scenario before you write down a single equation.

**Solution:**

We expect a never-ending cycle—an oscillation. The following diagram shows a qualitative picture of what happens.



Beginning with the top left node, we see that Romeo's affection increases. As a result, Juliet retreats, as depicted by the node on the top-right. In turn, this causes Romeo to lose hope and retreat, as shown in the bottom-right node. When Romeo pulls away, Juliet finds him mystically attractive and gravitates toward him, as shown by the bottom-left node. This causes Romeo to turn toward Juliet, which takes us back to the top-left node again, for yet another cycle.

Then determine a complete solution $\vec{s}[n]$ in the simplest of terms, assuming an initial state given by $\vec{s}[0] = \begin{bmatrix} 1 & 0 \end{bmatrix}^T$. As part of this, you must determine the eigenvalues and eigenvectors of the $\mathbf{A}$.

**Solution:**

The eigenvalues are the roots of the equation

$$\begin{aligned} \det(\lambda \mathbf{I} - \mathbf{A}) &= \det \begin{bmatrix} \lambda & -1 \\ 1 & \lambda \end{bmatrix} \\ &= \lambda^2 + 1 = 0. \end{aligned}$$

So, $\lambda_1 = i$ and $\lambda_2 = -i$. Constructing the matrices $\lambda_1 \mathbf{I} - \mathbf{A}$ and $\lambda_2 \mathbf{I} - \mathbf{A}$, we find the corresponding eigenvectors by inspection:

$$\lambda_1 \mathbf{I} - \mathbf{A} = \begin{bmatrix} i & -1 \\ 1 & i \end{bmatrix} \quad \Longrightarrow \quad \vec{v}_1 = \begin{bmatrix} 1 \\ i \end{bmatrix}$$

and

$$\lambda_2 \mathbf{I} - \mathbf{A} = \begin{bmatrix} -i & -1 \\ 1 & -i \end{bmatrix} \quad \Longrightarrow \quad \vec{v}_2 = \begin{bmatrix} 1 \\ -i \end{bmatrix}.$$

The matrix $\mathbf{A}$ has complex-valued eigenvalues and eigenvectors. Specifically, it has purely imaginary eigenvalues. This is not a coincidence. It turns out that if a matrix $\mathbf{A}$ has odd symmetry—that is, if $\mathbf{A}^T = -\mathbf{A}$—then its eigenvalues are purely imaginary.

Before we determine the general solution $\vec{s}[n]$, we must decompose the initial-state vector in terms of the two eigenvectors. The equation is

$$\underbrace{\begin{bmatrix} 1 & 1 \\ i & -i \end{bmatrix}}_{[\vec{v}_1 \;\; \vec{v}_2]} \underbrace{\begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix}}_{\vec{\alpha}} = \underbrace{\begin{bmatrix} 1 \\ 0 \end{bmatrix}}_{\vec{s}[0]},$$

which yields the coefficient vector

$$\begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = \begin{bmatrix} 1/2 \\ 1/2 \end{bmatrix}.$$

The general solution is given by

$$\begin{aligned} \vec{s}[n] &= \alpha_1 \lambda_1^n \vec{v}_1 + \alpha_2 \lambda_2^n \vec{v}_2 \\ &= \frac{1}{2} i^n \begin{bmatrix} 1 \\ i \end{bmatrix} + \frac{1}{2} (-i)^n \begin{bmatrix} 1 \\ -i \end{bmatrix}. \end{aligned}$$

Since the two terms on the right-hand side are complex conjugates of one another, we have

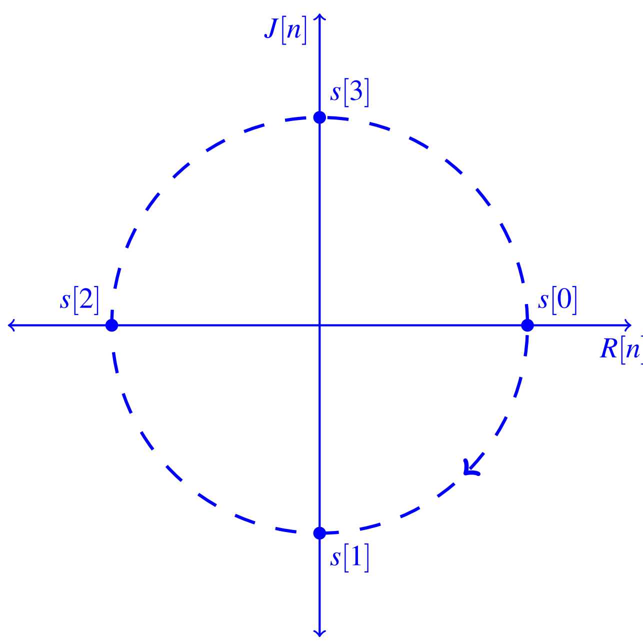
$$\begin{aligned}\vec{s}[n] &= 2\,\mathrm{Re}\left\{\frac{1}{2}i^n\begin{bmatrix}1\\i\end{bmatrix}\right\}\\ &= \mathrm{Re}\left\{\begin{bmatrix}i^n\\i^{n+1}\end{bmatrix}\right\}\end{aligned}$$

$$\vec{s}[n] = \begin{cases}\begin{bmatrix}1\\0\end{bmatrix} & \text{if } n \geq 0 \text{ and } n \mod 4 = 0\\ \begin{bmatrix}0\\-1\end{bmatrix} & \text{if } n \geq 0 \text{ and } n \mod 4 = 1\\ \begin{bmatrix}-1\\0\end{bmatrix} & \text{if } n \geq 0 \text{ and } n \mod 4 = 2\\ \begin{bmatrix}0\\1\end{bmatrix} & \text{if } n \geq 0 \text{ and } n \mod 4 = 3\end{cases}$$

Plot (by hand, or otherwise without the assistance of any scientific computing software package), on a two-dimensional plane (called a *phase plane*)—where the horizontal axis denotes $R[n]$ and the vertical axis denotes $J[n]$—representative points along the trajectory of the state vector $\vec{s}[n]$, starting from the initial state given in this part. Describe, in plain words, what Romeo and Juliet are doing in this scenario. In other words, what does their state trajectory look like? Determine $\|\vec{s}[n]\|^2$ for all $n = 0, 1, 2, \ldots$ to corroborate your description of the state trajectory.

**Solution:**

Romeo and Juliet are going around in a clockwise circle. Note that $\|\vec{s}[n]\|^2 = 1$ for all $n = 0, 1, 2, 3, \ldots$.




### 3. Image Compression

In this question, we explore how eigenvalues and eigenvectors can be used for image compression. A grayscale image can be represented as a data grid. Say a symmetric, square image is represented by a symmetric matrix $\mathbf{A}$, such that $\mathbf{A}^T = \mathbf{A}$. We can transform the images to vectors to make it easier to process them as data, but here, we will understand them as 2D data. Let $\lambda_1, \ldots, \lambda_n$ be the eigenvalues of $\mathbf{A}$ with corresponding eigenvectors $\vec{v}_1, \ldots, \vec{v}_n$. Also, let these eigenvectors be normalized (unit norm). Then, the matrix can be represented as the expansion

$$\mathbf{A} = \lambda_1 \vec{v}_1 \vec{v}_1^T + \lambda_2 \vec{v}_2 \vec{v}_2^T + \cdots + \lambda_n \vec{v}_n \vec{v}_n^T.$$

Note that the eigenvectors must be normalized for this expansion to be valid because we know that if $\vec{v}_i$ is an eigenvector, then any scalar multiple $\alpha \vec{v}_i$ is also an eigenvector. If we scaled every eigenvector on the right hand side of the equation by $\alpha$, then the left hand side would change from $\mathbf{A}$ to $\alpha^2 \mathbf{A}$.

The previous expansion shows that the matrix $\mathbf{A}$ can be synthesized by its $n$ eigenvalues and eigenvectors. However, $\mathbf{A}$ can also be *approximated* with the $k$ largest eigenvalues and the corresponding eigenvectors. That is,

$$\mathbf{A} \approx \lambda_1 \vec{v}_1 \vec{v}_1^T + \lambda_2 \vec{v}_2 \vec{v}_2^T + \cdots + \lambda_k \vec{v}_k \vec{v}_k^T.$$

(a) Construct appropriate matrices $\mathbf{V}$, $\mathbf{W}$ (using $\vec{v}_i$'s as rows and columns) and a matrix $\mathbf{\Lambda}$ with the eigenvalues $\lambda_i$ as components such that

$$\mathbf{A} = \mathbf{V} \mathbf{\Lambda} \mathbf{W}.$$

**Solution:**

First, we try to create a matrix operation that results in $\vec{v}_1 \vec{v}_1^T + \vec{v}_2 \vec{v}_2^T + \cdots + \vec{v}_n \vec{v}_n^T$ with matrices $\mathbf{V}$ and $\mathbf{W}$. Note that $\vec{v}_i \vec{v}_i^T$ has to be a matrix, so $\vec{v}_i$ has to be a column vector. Therefore,

$$\begin{bmatrix} | & & | \\ \vec{v}_1 & \cdots & \vec{v}_n \\ | & & | \end{bmatrix} \begin{bmatrix} \vec{v}_1^T \\ \vdots \\ \vec{v}_n^T \end{bmatrix} = \vec{v}_1 \vec{v}_1^T + \vec{v}_2 \vec{v}_2^T + \cdots + \vec{v}_n \vec{v}_n^T.$$

This is called an "outer product."

We can let

$$\mathbf{V} = \begin{bmatrix} | & & | \\ \vec{v}_1 & \cdots & \vec{v}_n \\ | & & | \end{bmatrix}; \mathbf{W} = \mathbf{V}^T.$$

Now to add the scalar $\lambda_i$s for each $\vec{v}_i \vec{v}_i^T$ pair, we can simply scale one set of vectors:

$$\begin{bmatrix} | & & | \\ \lambda_1 \vec{v}_1 & \cdots & \lambda_n \vec{v}_n \\ | & & | \end{bmatrix} \begin{bmatrix} \vec{v}_1^T \\ \vdots \\ \vec{v}_n^T \end{bmatrix} = \lambda_1 \vec{v}_1 \vec{v}_1^T + \lambda_2 \vec{v}_2 \vec{v}_2^T + \cdots + \lambda_n \vec{v}_n \vec{v}_n^T$$

Notice that we can scale the columns of $\mathbf{V}$ by $\lambda_i$ by right-multiplying it with a diagonal matrix [3]:

$$\begin{bmatrix} | & & | \\ \lambda_1 \vec{v}_1 & \cdots & \lambda_n \vec{v}_n \\ | & & | \end{bmatrix} = \begin{bmatrix} | & & | \\ \vec{v}_1 & \cdots & \vec{v}_n \\ | & & | \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 & \ldots & 0 \\ 0 & \lambda_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \lambda_n \end{bmatrix}$$

[3] This has to be the case since $\lambda_i$ times the $i$th standard basis vector $\vec{e}_i$ will have $\lambda_i$ times the $i$th column of the matrix $\mathbf{U}$ as the image of the linear operation defined by $\mathbf{U}$. This is because matrix multiplication on the right deals with columns.

Therefore, let

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_n \end{bmatrix}.$$

And we have

$$\mathbf{A} = \lambda_1 \vec{v}_1 \vec{v}_1^T + \lambda_2 \vec{v}_2 \vec{v}_2^T + \cdots + \lambda_n \vec{v}_n \vec{v}_n^T = \mathbf{V}\mathbf{\Lambda}\mathbf{W}.$$

(b) Use the IPython notebook `prob5.ipynb` and the image file `pattern.npy`. Use the `numpy.linalg.eig` command to find the $\mathbf{V}$ and $\mathbf{\Lambda}$ matrices for the image. Note that `numpy.linalg.eig` returns normalized eigenvectors by default. Mathematically, how many eigenvectors are required to fully capture the information within the image?

**Solution:**

The shape function told us that the image is a $400 \times 400$ matrix. We can therefore expect that we can have as many as 400 eigenvalues, and indeed looking at the `eig_vals` variable in the notebook tells us that all 400 of them are non-zero. This tells us that to fully understand the action of the matrix, we would need to know about all 400 of the eigenspaces. Therefore, we would need 400 eigenvectors to fully understand them.

This is true without using deeper tricks. In reality, you might expect that the fact that this particular matrix/image is symmetric should allow us to exploit that symmetry somehow to reduce the amount of information required to perfectly capture the matrix. Indeed, this is true.

(c) In the IPython notebook, find an approximation for the image using the 100 largest eigenvalues and eigenvectors.

**Solution:**

See `sol5.ipynb`.

(d) Repeat part (c) with $k = 50$. By further experimenting with the code, what seems to be the lowest value of $k$ that retains most of the salient features of the given image?

**Solution:**

See `sol5.ipynb`.

The question of the lowest value of $k$ is a bit subjective, and it is fine whatever answer you gave for it. The image starts looking qualitatively different somewhere around 15 eigenvectors. Certainly below 7, it looks very different. The "resolution" seems to be dropping as we keep fewer and fewer eigenvalues and eigenvectors.

Look at the plot of the eigenvalues included in the solutions notebook. You will see how they fast they become small.

This effect of a reduction in quality as we save less information is something that all of you have experienced while using things like JPEG compression. We hope that seeing this example gives you some idea of why it could be possible to do such "lossy compression" in real-world applications. Later in the 16AB sequence, we will be learning more about why this works.

Along with lossless compression and error-correcting codes, lossy compression is one of the major advances that makes the modern age of multimedia possible. So, give a silent shoutout to eigenconcepts next time you watch a video online.

## 4. Noisy Images

In lab, we used the single pixel camera to successfully capture a few images with various masks. The only restriction on a mask is that the mask only could only consist of 0's or 1's as elements and that the "mask matrix," that is, the matrix with the masks as row vectors, is invertible. There are many such matrices that satisfy this condition. One simple example of such a matrix is the identity matrix. In this problem, we are going to explore the design space of matrices that can be used as masks and see the effects of our choice of the matrix. Specifically, we will be analyzing what the effect of noise will be in our system and how we can pick matrices that mitigate the effects of noise.

Suppose that we are trying to capture $10 \times 10$ images using the single pixel camera from lab. Each mask is then a $10 \times 10$ image, represented by a vector of length 100. Since there are 100 unknowns, there will be 100 such vectors leading to a $100 \times 100$ mask matrix. We will call this matrix of masks $\mathbf{A}$. The image we are trying to capture will be $\vec{x}$, and we will refer to the measurements we make (i.e., after applying all the different masks to the object we are trying to image) as $\vec{b}$. Thus, $\mathbf{A}\vec{x} = \vec{b}$.

(a) Suppose that the measurement process adds noise. Then, rather than measuring $\vec{b}$, we measure $\vec{b} + \vec{n}$, where $\vec{n}$ is a vector representing the added noise. Express $\vec{x}$ in terms of $\mathbf{A}$, $\vec{b}$, and $\vec{n}$.

**Solution:**

The noise is added to the vector $\vec{b}$. Therefore,

$$\begin{aligned} \mathbf{A}\vec{x} &= \vec{b} + \vec{n} \\ \vec{x} &= \mathbf{A}^{-1}\left(\vec{b} + \vec{n}\right) \\ \vec{x} &= \mathbf{A}^{-1}\vec{b} + \mathbf{A}^{-1}\vec{n} \end{aligned}$$

(b) We are going to try different $\mathbf{A}$ matrices in this problem and compare how they deal with noise. Run the associated cells in the attached IPython notebook. What special matrix is $\mathbf{A}_1$? Are there any differences between the matrices $\mathbf{A}_2$ and $\mathbf{A}_3$?

**Solution:**

See `sol5.ipynb`.

The matrix $\mathbf{A}_1$ is the identity matrix. Notice that there are almost no visibile differences between the matrices $\mathbf{A}_2$ and $\mathbf{A}_3$.

(c) Run the associated cells in the attached IPython notebook. Notice that each plot returns the result of trying to image a noisy image as well as the minimum absolute value of the eigenvalue of each matrix. Comment on the effect of small eigenvalues on the noise in the image.

**Solution:**

See `sol5.ipynb`.

Notice that we are printing the eigenvalue with the smallest absolute value. As the absolute value of the smallest eigenvalue decreases, the noise in the result increases.

(d) The associated IPython notebook also prints out how many eigenvectors each matrix has. Notice each matrix has 100 eigenvectors. What does this imply about the span of the eigenvectors? Can the noise vector be written as a linear combination of the eigenvectors?

**Solution:**

Recall that eigenvectors of different eigenvalues must be linearly independent. Since we have 100 linearly independent vectors, we know that we have a basis for $\mathbb{R}^{100}$. Therefore, the noise vector can be represented as a linear combination of the basis vectors.

(e) Small eigenvalues of $\mathbf{A}$ seem to cause problems for our imaging system. Inverting the matrix $\mathbf{A}$ turns these small eigenvalues into large eigenvalues. Show that if $\lambda$ is an eigenvalue of a matrix $\mathbf{A}$, then $\frac{1}{\lambda}$ is an eigenvalue of the matrix $\mathbf{A}^{-1}$.

*Hint:* Start with an eigenvalue $\lambda$ and one corresponding eigenvector $\vec{v}$, such that they satisfy $\mathbf{A}\vec{v} = \lambda\vec{v}$.

**Solution:**

For some eigenvector $\vec{v}$ and associated eigenvalue $\lambda$, we know that:

$$\begin{aligned}
\mathbf{A}\vec{v} &= \lambda\vec{v} \\
\vec{v} &= \mathbf{A}^{-1}(\lambda\vec{v}) \\
\vec{v} &= \lambda\mathbf{A}^{-1}\vec{v} \\
\frac{1}{\lambda}\vec{v} &= \mathbf{A}^{-1}\vec{v} \\
\mathbf{A}^{-1}\vec{v} &= \frac{1}{\lambda}\vec{v}
\end{aligned}$$

Therefore, $\frac{1}{\lambda}$ is an eigenvalue of $\mathbf{A}^{-1}$ with the corresponding eigenvector $\vec{v}$.

## 5. Is There A Steady State?

So far, we've seen that for a conservative state transition matrix $\mathbf{A}$, we can find the eigenvector, $\vec{v}$, corresponding to the eigenvalue $\lambda = 1$. This vector is the steady state since $\mathbf{A}\vec{v} = \vec{v}$. However, we've so far taken for granted that the state transition matrix even has the eigenvalue $\lambda = 1$. Let's try to prove this fact.

(a) Show that if $\lambda$ is an eigenvalue of a matrix $\mathbf{A}$, then it is also an eigenvalue of the matrix $\mathbf{A}^T$.

*Hint:* Recall that the determinants of $\mathbf{A}$ and $\mathbf{A}^T$ are the same.

**Solution:**

Recall that we find the eigenvalues of a matrix $\mathbf{A}$ by setting the determinant of $\mathbf{A} - \lambda\mathbf{I}$ to 0.

$$\det(\mathbf{A} - \lambda\mathbf{I}) = \det\left((\mathbf{A} - \lambda\mathbf{I})^T\right) = \det\left(\mathbf{A}^T - \lambda\mathbf{I}\right) = 0$$

Since the two determinants are equal, the characteristic polynomicals of the two matrices must also be equal. Therefore, they must have the same eigenvalues.

(b) Let a square matrix $\mathbf{A}$ have rows that sum to one. Show that $\vec{1} = \begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix}^T$ is an eigenvector of $\mathbf{A}$. What is the corresponding eigenvalue?

**Solution:**

Recall that if the rows of $\mathbf{A}$ sum to one, then $\mathbf{A}\vec{1} = \vec{1}$. Therefore, the corresponding eigenvalue is $\lambda = 1$.

(c) Let's put it together now. From the previous two parts, show that any conservative state transition matrix will have the eigenvalue $\lambda = 1$. Recall that conservative state transition matrices are those that have columns that sum to 1.

**Solution:**

If we tranpose a conservative state transition matrix $\mathbf{A}$, then the rows of $\mathbf{A}^T$ (or the columns of $\mathbf{A}$) sum to one by definition of a conservative system. Then, from part (b), we know that $\mathbf{A}^T$ has the eigenvalue $\lambda = 1$. Furthermore, from part (a), we know that the $\mathbf{A}$ and $\mathbf{A}^T$ have the same eigenvalues, so $\mathbf{A}$ also has the eigenvalue $\lambda = 1$.

**6. Fibonacci Sequence**

One of the most useful things about diagonalization is that it allows us to easily compute polynomial functions of matrices. This in turn lets us do far more, including solving many linear recurrence relations. This problem shows you how this can be done for the Fibonacci numbers, but you should notice that the same exact technique can be applied far more generally.

(a) The Fibonacci sequence can be constructed according to the following relation. The $N$th number in the Fibonacci sequence, $F_N$, is computed by adding the previous two numbers in the sequence together:

$$F_N = F_{N-1} + F_{N-2}$$

We select the first two numbers in the sequence to be $F_1 = 0$ and $F_2 = 1$ and then we can compute the following numbers as

$$0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, \ldots$$

Write the operation of computing the next Fibonacci numbers from the previous two using matrix multiplication:

$$\begin{bmatrix} F_N \\ F_{N-1} \end{bmatrix} = \mathbf{A} \begin{bmatrix} F_{N-1} \\ F_{N-2} \end{bmatrix}$$

**Solution:**

$$\begin{bmatrix} F_N \\ F_{N-1} \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}}_{\mathbf{A}} \begin{bmatrix} F_{N-1} \\ F_{N-2} \end{bmatrix}$$

(b) Diagonalize $\mathbf{A}$ to show that

$$F_N = \frac{1}{\sqrt{5}} \left( \frac{1+\sqrt{5}}{2} \right)^{N-1} - \frac{1}{\sqrt{5}} \left( \frac{1-\sqrt{5}}{2} \right)^{N-1}$$

is an analytical expression for the $N$th Fibonacci number.

**Solution:**

Note that $\mathbf{A}$ has the following eigenvalues and corresponding eigenvectors:

$$\left\{ \lambda_1 = \frac{1+\sqrt{5}}{2}, \lambda_2 = \frac{1-\sqrt{5}}{2} \right\} \qquad \left\{ \vec{p}_1 = \begin{bmatrix} \frac{1+\sqrt{5}}{2} \\ 1 \end{bmatrix}, \vec{p}_2 = \begin{bmatrix} \frac{1-\sqrt{5}}{2} \\ 1 \end{bmatrix} \right\}$$

And recall the $2 \times 2$ inverse formula:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

Using these facts, first we diagonalize $\mathbf{A}$

$$\mathbf{P} = \begin{bmatrix} \frac{1+\sqrt{5}}{2} & \frac{1-\sqrt{5}}{2} \\ 1 & 1 \end{bmatrix} \qquad \mathbf{P}^{-1} = \frac{1}{\sqrt{5}} \begin{bmatrix} 1 & \frac{-1+\sqrt{5}}{2} \\ -1 & \frac{1+\sqrt{5}}{2} \end{bmatrix}$$

$$\mathbf{A} = \mathbf{PDP}^{-1}$$
$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} \frac{1+\sqrt{5}}{2} & \frac{1-\sqrt{5}}{2} \\ 1 & 1 \end{bmatrix} \begin{bmatrix} \frac{1+\sqrt{5}}{2} & 0 \\ 0 & \frac{1-\sqrt{5}}{2} \end{bmatrix} \left( \frac{1}{\sqrt{5}} \begin{bmatrix} 1 & \frac{-1+\sqrt{5}}{2} \\ -1 & \frac{1+\sqrt{5}}{2} \end{bmatrix} \right)$$

Then, we have that $F_N$ is equal to the first element of $\mathbf{A}^{N-2} \begin{bmatrix} 1 \\ 0 \end{bmatrix}$.

$$\begin{aligned} F_N &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^{N-2} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \frac{1+\sqrt{5}}{2} & \frac{1-\sqrt{5}}{2} \\ 1 & 1 \end{bmatrix} \begin{bmatrix} \frac{1+\sqrt{5}}{2} & 0 \\ 0 & \frac{1-\sqrt{5}}{2} \end{bmatrix}^{N-2} \left( \frac{1}{\sqrt{5}} \begin{bmatrix} 1 & \frac{-1+\sqrt{5}}{2} \\ -1 & \frac{1+\sqrt{5}}{2} \end{bmatrix} \right) \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\ &= \begin{bmatrix} \frac{1+\sqrt{5}}{2} & \frac{1-\sqrt{5}}{2} \end{bmatrix} \begin{bmatrix} \left(\frac{1+\sqrt{5}}{2}\right)^{N-2} & 0 \\ 0 & \left(\frac{1-\sqrt{5}}{2}\right)^{N-2} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{5}} \\ -\frac{1}{\sqrt{5}} \end{bmatrix} \\ &= \frac{1}{\sqrt{5}} \left( \frac{1+\sqrt{5}}{2} \right)^{N-1} - \frac{1}{\sqrt{5}} \left( \frac{1-\sqrt{5}}{2} \right)^{N-1} \end{aligned}$$