Homework 1 solutions

2.40. Representing linear functions as matrix multiplication. Suppose that $f : \mathbb{R}^n \longrightarrow \mathbb{R}^m$ is linear. Show that there is a matrix $A \in \mathbb{R}^{m\times n}$ such that for all $x \in \mathbb{R}^n$, $f(x) = Ax$. (Explicitly describe how you get the coefficients A_{ij} from f, and then verify that $f(x) = Ax$ for any $x \in \mathbb{R}^n$.) Is the matrix A that represents f unique? In other words, if $\tilde{A} \in \mathbb{R}^{m\times n}$ is another matrix such that $f(x) = \tilde{A}x$ for all $x \in \mathbb{R}^n$, then do we have $\tilde{A} = A$? Either show that this is so, or give an explicit counterexample.

Solution. Any $x \in \mathbb{R}^n$ can be written as

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = x_1e_1 + x_2e_2 + \cdots + x_ne_n,$$

where e_i is the ith standard unit vector in $\mathbb{R}^n$. From linearity of f we have

$$f(x) = x_1f(e_1) + x_2f(e_2) + \cdots + x_nf(e_n),$$

or in (block) matrix form

$$f(x) = \begin{bmatrix} f(e_1) & f(e_2) & \cdots & f(e_n) \end{bmatrix} x.$$

Therefore, we simply take

$$A := \begin{bmatrix} f(e_1) & f(e_2) & \cdots & f(e_n) \end{bmatrix}.$$

So to determine A we only need to find $f(e_i)$, for $i = 1, \ldots, n$.

Suppose the matrix A is not unique, which means there is another $\tilde{A} \in \mathbb{R}^{m\times n}$ such that $f(x) = \tilde{A}x$. Then $Ax = \tilde{A}x$ or $(A - \tilde{A})x = 0$ for all $x \in \mathbb{R}^n$. When $x = e_i$, $(A - \tilde{A})e_i = 0$ implies that the ith column of $(A - \tilde{A})$ is zero. Repeating this argument for $i = 1, 2, \ldots, n$ proves that *all* columns of $(A - \tilde{A})$ are zero and hence $A = \tilde{A}$. Therefore the choice of A is *unique*.

2.70. Matrix representation of linear systems. Consider the (discrete-time) linear dynamical system

$$x(t + 1) = A(t)x(t) + B(t)u(t), \quad y(t) = C(t)x(t) + D(t)u(t).$$

Find a matrix G such that

$$\begin{bmatrix} y(0) \\ y(1) \\ \vdots \\ y(N) \end{bmatrix} = G \begin{bmatrix} x(0) \\ u(0) \\ \vdots \\ u(N) \end{bmatrix}.$$

The matrix G shows how the output at $t = 0, \ldots, N$ depends on the initial state $x(0)$ and the sequence of inputs $u(0), \ldots, u(N)$.

Solution. For $t = 0$, $x(t+1) = A(t)x(t) + B(t)u(t)$ becomes

$$x(1) = A(0)x(0) + B(0)u(0).$$

For $t = 1$

$$\begin{aligned} x(2) &= A(1)x(1) + B(1)u(1) \\ &= A(1)\left[A(0)x(0) + B(0)u(0)\right] + B(1)u(1) \\ &= A(1)A(0)x(0) + A(1)B(0)u(0) + B(1)u(1). \end{aligned}$$

For $t = 2$

$$\begin{aligned} x(3) &= A(2)x(2) + B(2)u(2) \\ &= A(2)\left[A(1)A(0)x(0) + A(1)B(0)u(0) + B(1)u(1)\right] + B(2)u(2) \\ &= A(2)A(1)A(0)x(0) + A(2)A(1)B(0)u(0) + A(2)B(1)u(1) + B(2)u(2). \end{aligned}$$

Now it is easy to guess the general expression for $x(i)$ in terms of $x(0), u(0), \ldots, u(i-1)$ (which can be later proved inductively) as follows

$$\begin{aligned} x(i) = {} & A(i-1)A(i-2)\cdots A(0)x(0) \\ & + A(i-1)A(i-2)\cdots A(1)B(0)u(0) \\ & + A(i-1)A(i-2)\cdots A(2)B(1)u(1) \\ & + A(i-1)A(i-2)\cdots A(3)B(2)u(2) \\ & \vdots \\ & + A(i-1)B(i-2)u(i-2) \\ & + B(i-1)u(i-1). \end{aligned}$$

Given $x(i)$ as above, $y(i)$ is simply found from $y(i) = C(i)x(i) + D(i)u(i)$ in terms of $x(0), u(0), \ldots, u(i)$ as

$$\begin{aligned} y(i) = {} & C(i)A(i-1)A(i-2)\cdots A(0)x(0) \\ & + C(i)A(i-1)A(i-2)\cdots A(1)B(0)u(0) \\ & + C(i)A(i-1)A(i-2)\cdots A(2)B(1)u(1) \\ & + C(i)A(i-1)A(i-2)\cdots A(3)B(2)u(2) \\ & \vdots \\ & + C(i)A(i-1)B(i-2)u(i-2) \\ & + C(i)B(i-1)u(i-1) \\ & + D(i)u(i). \end{aligned}$$

Therefore

$$G_{i1} = \begin{cases} C(0) & \text{if } i = 1 \\ C(i-1)A(i-2)\cdots A(0) & \text{if } 2 \le i \le N+1 \end{cases}$$

and for $i = 1, 2, \ldots, N+1$

$$G_{ij} = \begin{cases} 0 & \text{if } N+2 \geq j > i+1 \\ D(i-1) & \text{if } j = i+1 \\ C(i-1)B(i-2) & \text{if } j = i \\ C(i-1)A(i-2)\cdots A(j-1)B(j-2) & \text{if } 2 \leq j < i \end{cases}$$

Note: A number of students used product notation "$\prod_{k=1}^{n} A_k$" to denote the product $A_n \cdots A_1$. It's advisable to avoid this notation with matrices, since it's not clear (unless you specify explicitly) whether this means that or $A_1 \cdots A_n$, and the two aren't equal, because matrix multiplication is not commutative.

2.100. A mass subject to applied forces. Consider a unit mass subject to a time-varying force $f(t)$ for $0 \leq t \leq n$. Let the initial position and velocity of the mass both be zero. Suppose that the force has the form $f(t) = x_j$ for $j-1 \leq t < j$ and $j = 1, \ldots, n$. Let y_1 and y_2 denote, respectively, the position and velocity of the mass at time $t = n$.

a) Find the matrix $A \in \mathbb{R}^{2\times n}$ such that $y = Ax$.

b) For $n = 4$, find a sequence of input forces $x_1, \ldots, x_n$ that moves the mass to position 1 with velocity 0 at time n.

Solution. Let $p(t)$ and $v(t)$ denote, respectively, the position and velocity of the mass at time t.

a) The velocity is the integral of the applied force:

$$\begin{aligned} v(t) &= v(0) + \int_0^t f(\tau)\,d\tau \\ &= v(0) + \sum_{j=1}^{\lfloor t \rfloor} \int_{j-1}^{j} f(\tau)\,d\tau + \int_{\lfloor t \rfloor}^{t} f(\tau)\,d\tau \\ &= v(0) + \sum_{j=1}^{\lfloor t \rfloor} \int_{j-1}^{j} x_j\,d\tau + \int_{\lfloor t \rfloor}^{t} x_{\lfloor t \rfloor + 1}\,d\tau \\ &= v(0) + \sum_{j=1}^{\lfloor t \rfloor} (\tau x_j)|_{\tau=j-1}^{\tau=j} + (\tau x_{\lfloor t \rfloor+1})|_{\tau=\lfloor t \rfloor}^{\tau=t} \\ &= v(0) + \sum_{j=1}^{\lfloor t \rfloor} x_j + (t - \lfloor t \rfloor) x_{\lfloor t \rfloor + 1}. \end{aligned}$$

In particular, because the mass is initially at rest (that is, $v(0) = 0$), the final velocity is

$$y_2 = v(n) = \sum_{j=1}^{n} x_j.$$

Similarly, the position is the integral of the velocity:

$$
\begin{aligned}
p(t) &= p(0) + \int_0^t v(\tau)\,d\tau \\
&= p(0) + \int_0^t (v(0) + (v(\tau) - v(0)))\,d\tau \\
&= p(0) + v(0)t + \int_0^t (v(\tau) - v(0))\,d\tau \\
&= p(0) + v(0)t + \sum_{j=1}^{\lfloor \tau \rfloor} \int_{j-1}^{j} (v(\tau) - v(0))\,d\tau + \int_{\lfloor t \rfloor}^{t} (v(\tau) - v(0))\,d\tau \\
&= p(0) + v(0)t + \sum_{j=1}^{\lfloor t \rfloor} \int_{j-1}^{j} \left(\sum_{k=1}^{\lfloor \tau \rfloor} x_k + (\tau - \lfloor \tau \rfloor) x_{\lfloor \tau \rfloor + 1} \right) d\tau \\
&\quad + \int_{\lfloor t \rfloor}^{t} \left(\sum_{k=1}^{\lfloor \tau \rfloor} x_k + (\tau - \lfloor \tau \rfloor) x_{\lfloor \tau \rfloor} + 1 \right) d\tau \\
&= p(0) + v(0)t + \sum_{j=1}^{\lfloor t \rfloor} \int_{j-1}^{j} \left(\sum_{k=1}^{j-1} x_k + (\tau - (j-1)) x_j \right) d\tau \\
&\quad + \int_{\lfloor t \rfloor}^{t} \left(\sum_{k=1}^{\lfloor t \rfloor} x_k + (\tau - \lfloor t \rfloor) x_{\lfloor t \rfloor + 1} \right) d\tau \\
&= p(0) + v(0)t + \sum_{k=1}^{\lfloor t \rfloor} \left(\sum_{k=1}^{j-1} \tau x_k + \frac{1}{2}(\tau - (j-1))^2 x_j \right) \Bigg|_{\tau=j-1}^{\tau=j} \\
&\quad + \left(\sum_{k=1}^{\lfloor t \rfloor} \tau x_k + \frac{1}{2}(\tau - \lfloor t \rfloor)^2 x_{\lfloor t \rfloor + 1} \right) \Bigg|_{\tau=\lfloor t \rfloor}^{\tau=t} \\
&= p(0) + v(0)t + \sum_{j=1}^{\lfloor t \rfloor} \left(\sum_{k=1}^{j-1} x_k + \frac{1}{2} x_j \right) + \left(\sum_{k=1}^{\lfloor t \rfloor} x_k + \frac{1}{2}(t - \lfloor t \rfloor)^2 x_{\lfloor t \rfloor + 1} \right) \\
&= p(0) + v(0)t + \sum_{j=1}^{\lfloor t \rfloor} \sum_{k=1}^{j-1} x_k + \sum_{j=1}^{\lfloor t \rfloor} \frac{1}{2} x_j + \sum_{k=1}^{\lfloor t \rfloor} x_k + \frac{1}{2}(t - \lfloor t \rfloor)^2 x_{\lfloor t \rfloor + 1} \\
&= p(0) + v(0)t + \sum_{k=1}^{\lfloor t \rfloor} \sum_{j=k+1}^{\lfloor t \rfloor} x_k + \sum_{k=1}^{\lfloor t \rfloor} \frac{1}{2} x_k + \sum_{k=1}^{\lfloor t \rfloor} x_k + \frac{1}{2}(t - \lfloor t \rfloor)^2 x_{\lfloor t \rfloor + 1} \\
&= p(0) + v(0)t + \sum_{k=1}^{\lfloor t \rfloor} (\lfloor t \rfloor - k) x_k + \sum_{k=1}^{\lfloor t \rfloor} \frac{1}{2} x_k + \sum_{k=1}^{\lfloor t \rfloor} x_k + \frac{1}{2}(t - \lfloor t \rfloor)^2 x_{\lfloor t \rfloor + 1} \\
&= p(0) + v(0)t + \sum_{k=1}^{\lfloor t \rfloor} \left((\lfloor t \rfloor - k) + \frac{1}{2} + (t - \lfloor t \rfloor) \right) x_k + \frac{1}{2}(t - \lfloor t \rfloor)^2 x_{\lfloor t \rfloor + 1}
\end{aligned}
$$

$$= p(0) + v(0)t + \sum_{k=1}^{\lfloor t \rfloor} \left(t - k + \frac{1}{2} \right) x_k + \frac{1}{2}(t - \lfloor t \rfloor)^2 x_{\lfloor t \rfloor + 1}.$$

In particular, because the mass is initially at rest at the origin (that is, $p(0) = 0$ and $v(0) = 0$), the final position is

$$y_1 = p(n) = \sum_{j=1}^{n} \left(n - k + \tfrac{1}{2} \right) x_j.$$

Thus, we obtain the following system of linear equations:

$$y_1 = \sum_{j=1}^{n} (n - j + \tfrac{1}{2}) x_j,$$
$$y_2 = \sum_{j=1}^{n} x_j.$$

Since A_{ij} gives the coefficient of x_j in our expression for y_i, we have that

$$A_{1j} = n - j + \tfrac{1}{2} \qquad \text{and} \qquad A_{2j} = 1, \qquad j = 1, \ldots, n.$$

More concretely, we have that

$$A = \begin{bmatrix} n - \frac{1}{2} & n - \frac{3}{2} & \cdots & \frac{3}{2} & \frac{1}{2} \\ 1 & 1 & \cdots & 1 & 1 \end{bmatrix}.$$

b) We want to solve the following system of linear equations:

$$\begin{bmatrix} \frac{7}{2} & \frac{5}{2} & \frac{3}{2} & \frac{1}{2} \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

This system is underdetermined, and has infinitely many solutions. Suppose we choose $x_2 = x_3 = 0$. Then, we are left with the system

$$\begin{bmatrix} \frac{7}{2} & \frac{1}{2} \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_4 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

The second equation implies that $x_4 = -x_1$. Then, the first equation becomes

$$\tfrac{7}{2}x_1 + \tfrac{1}{2}x_4 = \tfrac{7}{2}x_1 - \tfrac{1}{2}x_1 = 3x_1 = 1.$$

Solving this equation, we find that $x_1 = \frac{1}{3}$. Substituting this value into our expression for x_4 gives $x_4 = -x_1 = -\frac{1}{3}$. Thus, one sequence of input forces that moves the mass to position 1 with velocity 0 at time n is

$$x = \begin{bmatrix} \frac{1}{3} \\ 0 \\ 0 \\ -\frac{1}{3} \end{bmatrix}.$$

2.120. Counting sequences in a language or code. We consider a language or code with an alphabet of n symbols $1, 2, \ldots, n$. A sentence is a finite sequence of symbols, $k_1, \ldots, k_m$ where $k_i \in \{1, \ldots, n\}$. A language or code consists of a set of sequences, which we will call the *allowable sequences*. A language is called *Markov* if the allowed sequences can be described by giving the allowable transitions between consecutive symbols. For each symbol we give a set of symbols which are allowed to follow the symbol. As a simple example, consider a Markov language with three symbols $1, 2, 3$. Symbol 1 can be followed by 1 or 3; symbol 2 must be followed by 3; and symbol 3 can be followed by 1 or 2. The sentence 1132313 is allowable (*i.e.*, in the language); the sentence 1132312 is not allowable (*i.e.*, not in the language). To describe the allowed symbol transitions we can define a matrix $A \in \mathbb{R}^{n\times n}$ by

$$A_{ij} = \begin{cases} 1 & \text{if symbol } i \text{ is allowed to follow symbol } j \\ 0 & \text{if symbol } i \text{ is not allowed to follow symbol } j \end{cases}.$$

a) Let $B = A^r$. Give an interpretation of B_{ij} in terms of the language.

b) Consider the Markov language with five symbols $1, 2, 3, 4, 5$, and the following transition rules:

- 1 must be followed by 2 or 3
- 2 must be followed by 2 or 5
- 3 must be followed by 1
- 4 must be followed by 4 or 2 or 5
- 5 must be followed by 1 or 3

Find the total number of allowed sentences of length 10. Compare this number to the simple code that consists of all sequences from the alphabet (*i.e.*, all symbol transitions are allowed). In addition to giving the answer, you must explain how you solve the problem. Do not hesitate to use Julia.

Solution.

a) If $B = A^k$, then B_{ij} is the number of sequences of length $k+1$ that start with symbol j and end with symbol i.

Here is a formal proof. Let $S_L(i,j)$ denote the set of sentences of length L that start with symbol j and end with symbol i:

$$S_L(i,j) = \{(k_1 = j, k_2, \ldots, k_{L-1}, k_L = i) \in \mathbb{N}_n^L \mid A_{k_\ell k_{\ell+1}} = 1 \text{ for all } \ell = 1, \ldots, L-1\}.$$

We claim that

$$(A^p)_{ij} = |S_{p+1}(i,j)|$$

for all $i, j \in \mathbb{N}_n$ and $p \in \mathbb{N}$. First, we introduce some notation. Given a finite sequence of symbols $(k_1, \ldots, k_L)$ and a symbol k_{L+1}, we define

$$(k_1, \ldots, k_L) + k_{L+1} = (k_1, \ldots, k_L, k_{L+1}).$$

(Thus, + denotes the operation of appending a symbol to a finite sequence of symbols.) Similarly, given a set S of finite sequences of symbols, we define

$$S + j = \{s + j \mid s \in S\}.$$

Finally, we define

$$\phi(i) = \{j \in \mathbb{N}_n \mid \text{symbol } i \text{ is allowed to follow symbol } j\} = \{j \in \mathbb{N}_n \mid A_{ij} = 1\}.$$

First, we prove the claim when $p = 1$:

$$(A^1)_{ij} = |S_2(i, j)|.$$

Note that $|S_2(i, j)| = 1$ if (j, i) is a sentence in the language (that is, symbol i is allowed to follow symbol j; or, equivalently, $A_{ij} = 1$), and $|S_2(i, j)| = 0$ otherwise (that is, symbol i is not allowed to follow symbol j; or, equivalently, $a_{ij} = 0$). In either case, we have that $|S_2(i, j)| = A_{ij} = (A^1)_{ij}$. Now suppose that $(A^p)_{ij} = |S_{p+1}(i, j)|$ for some $p \in \mathbb{N}$. We can partition $S_{p+2}(i, j)$ based on the penultimate symbol:

$$S_{p+2}(i, j) = \bigsqcup_{k \in \phi(i)} (S_{p+1}(k, j) + i).$$

Since the size of a disjoint union is equal to the sum of the sizes of the sets forming the union, we have that

$$|S_{p+2}(i, j)| = \sum_{k \in \phi(i)} |S_{p+1}(k, j) + i| = \sum_{k \in \phi(i)} |S_{p+1}(k, j)|.$$

Because $A_{ik} = 1$ for $k \in \phi(i)$, and $A_{ik} = 0$ for $k \notin \phi(i)$, we have that

$$|S_{p+2}(i, j)| = \sum_{k=1}^{n} A_{ik} |S_{p+1}(k, j)|.$$

Using the induction hypothesis, we have that $|S_{p+1}(k, j)| = (A^p)_{kj}$, and hence that

$$|S_{p+2}(i, j)| = \sum_{k=1}^{n} A_{ik} (A^p)_{kj} = (AA^p)_{ij} = (A^{p+1})_{ij}.$$

By induction, this proves that our interpretation of $(A^p)_{ij}$ is correct for all $p \in \mathbb{N}$.

b) For the given Markov language we can find the number of allowed sequences of length 10 by simply adding all the entries of the matrix A^9. Form the description of the rules we have

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix}.$$

Using the Julia command `B = A^9` we find

$$B = A^9 = \begin{bmatrix} 41 & 49 & 24 & 113 & 37 \\ 55 & 65 & 31 & 150 & 49 \\ 42 & 49 & 23 & 113 & 37 \\ 0 & 0 & 0 & 1 & 0 \\ 31 & 37 & 18 & 86 & 28 \end{bmatrix}.$$

(Note that there is only one word that ends with 4 (*i.e.*, 4444444444, because 4 can only follow 4.) Adding all the elements of B, using for example the Julia command `sum(B)`, we find that the total number of allowed sequences of length 10 is 1079. Finally, we can compare this number to the simple code that consists of all sequences from the alphabet. Of course there are $5^{10} = 9765625$ such sequences. Just to check our method, we can also compute this number the same way as above, by forming the matrix

$$A = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix},$$

(which means any symbol can follow any symbol), then find $B = A^9$, and adding all the entries of the matrix B. (Yes, you do get the same number as above ...)

2.230. Population dynamics. An ecosystem consists of n species that interact (say, by eating other species, eating each other's food sources, eating each other's predators, and so on). We let $x(t) \in \mathbb{R}^n$ be the vector of deviations of the species populations (say, in thousands) from some equilibrium values (which don't matter here), in time period (say, month) t. In this model, time will take on the discrete values $t = 0, 1, 2, \ldots$. Thus $x_3(4) < 0$ means that the population of species 3 in time period 4 is below its equilibrium level. (It does not mean the population of species 3 is negative in time period 4.) The population (deviations) follows a discrete-time linear dynamical system:

$$x(t+1) = Ax(t).$$

We refer to $x(0)$ as the *initial population perturbation*.

The questions below pertain to the specific case with $n = 10$ species, with matrix A given in `pop_dyn_data.json`.

a) Suppose the initial perturbation is $x(0) = e_4$ (meaning, we inject one thousand new creatures of species 4 into the ecosystem at $t = 0$). How long will it take to affect the other species populations? In other words, report a vector s, where s_i is the smallest t for which $x_i(t) \neq 0$. (We have $s_4 = 0$).

b) *Population control.* We can choose any initial perturbation that satisfies $|x_i(0)| \leq 1$ for each $i = 1, \ldots, 10$. (We achieve this by introducing additional creatures and/or hunting and fishing.) What initial perturbation $x(0)$ would you choose in order to maximize the population of species 1 at time $t = 10$? Explain your reasoning. Give the initial perturbation, and using your selected initial perturbation, give $x_1(10)$ and plot $x_1(t)$ versus t for $t = 0, \ldots, 40$.

Solution.

a) We can simply simulate the first few time periods by iterating $x(t+1) = Ax(t)$ and check when the elements of the vectors $x(t)$ first become nonzero. The code for this method is given below.

We find that $s = (4, 2, 1, 0, 4, 3, 3, 2, 4, 3)$.

Alternatively, since the system evolves by repeated application of the matrix A, $x(t) = A^t x(0)$. Since $x(0) = e_4$, we will have $x_i(t) \neq 0$ exactly when $(A^t)_{i,4} \neq 0$. We can thus solve this problem by examining the 4th columns of each of the first few powers (say, 10) of A, looking to see when the elements first become nonzero.

b) Let $B = A^{10}$, so $x(10) = A^{10}x(0) = Bx(0)$. We want to maximize $x_1(10)$, which is given by the inner product of the 1st row of B with $x(0)$. That is,

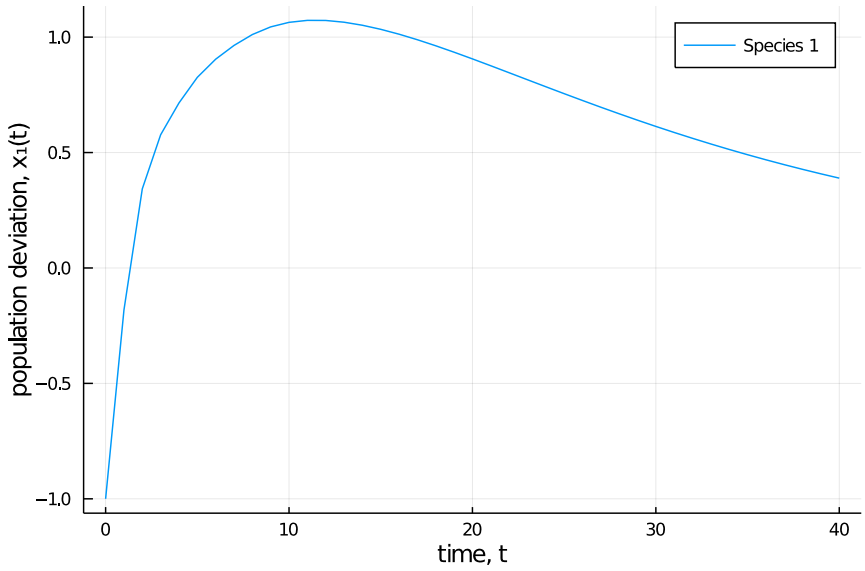
$$x_1(10) = \sum_{j=1}^{10} b_{1,j} x_j(0).$$

This sum is called *separable* because the jth term depends only on the variable $x_j(0)$. That means that the sum is maximized when each term is maximized separately. Since we are constrained to having the elements $|x_j(0)| \leq 1$, it's easy to see that the term $b_{1,j}x_j(0)$ is maximized when $x_j(0) = \text{sign}(b_{1,j})$, *i.e.*, $x_j(0)$ is either $+1$ or -1, with the sign matching the sign of $b_{1,j}$.

The code and the plot of $x_1(t)$ is given below. We find that

$$x(0) = (-1, -1, 1, -1, 1, 1, 1, 1, 1, -1)$$

and $x_1(10) = 1.06$. Note that, counterintuitively, the $x(0)$ which maximizes $x_1(10)$ sets the initial perturbation of species 1 to -1!



```
# cell 1
include("readclassjson.jl")
data = readclassjson("pop_dyn_data.json")
A = data["A"]
n = data["n"]

# cell 2
s = -ones(Int64, n)
x = zeros(n)
x[4] = 1.0
for t = 0:10
    s[(x .!= 0) .& (s .== -1)] .= t
    if all(s .!= -1)
        break
    end
    x = A*x
end
s

# cell 3
B = A^10
xat0 = sign.(B[1,:])

# cell 4
xat10 = B*xat0
xat10[1]

# cell 5
using Plots
x1history = zeros(41)
x = xat0
x1history[1] = x[1]
for t = 1:40
    x = A*x
    x1history[t+1] = x[1]
end
plot(0:40, x1history, label="Species 1",
    ylabel="population deviation, x1(t)", xlabel="time, t")
```