ON THE FIBONACCI NUMBERS

Prepared by Kei Nakamura

The *Fibonacci numbers* are terms of the sequence defined in a quite simple recursive fashion. However, despite its simplicity, they have some curious properties which are worth attention. In this set of notes, we will look at some of the important features of these numbers.

In the first half of the notes, our attention shall be paid to the relationship of the Fibonacci numbers and the *Euclidean algorithm*. The main theorem is that of Gabriel Lamé, which relates the complexity of the Euclidean algorithm with the Fibonacci numbers.

In the second half of the notes, we will look at the Fibonacci numbers from linear algebraic point of view. Our treatment shall essentially be a glimpse of the more general theory of *linear recurrence*.

1 The Fibonacci Numbers

Definition 1 *The sequence* $\{f_n\}_{n=0}^{\infty}$ *of natural numbers defined by the equations* $f_0 = 0$, $f_1 = 1$, *and* $f_{n+2} = f_{n+1} + f_n$ *is called the* Fibonacci sequence *or the* sequence of the Fibonacci numbers. *The* n*-th term in the sequence is called the* n-th Fibonacci number.

Despite its simplicity, the Fibonacci sequence yields some substantial results in relation to other topics in mathematics. In many cases, the theory surrounding the Fibonacci numbers would often involve a special number $\frac{1}{2}(1+\sqrt{5})$. We will denote this number by φ in this set of notes without further mentioning. First, we will state some properties of this number φ in the following lemma.

Lemma 2 *For every natural number* n, *we have:*
1) $\varphi^{n+2} = \varphi^{n+1} + \varphi^n$, *and*
2) $(1-\varphi)^{n+2} = (1-\varphi)^{n+1} + (1-\varphi)^n$.

Proof: It is easy to verify $\varphi^2 = \varphi + 1$ and $(1-\varphi)^2 = (1-\varphi) + 1$ by calculating directly or by using the fact that φ and $1-\varphi$ are solutions to the quadratic equation $x^2 - x - 1 = 0$. Now, multiplying both sides of these equations by φ^n and $(1-\varphi)^n$ respectively, we obtain the desired results.

Note that the above equations resemble the equation $f_{n+2} = f_{n+1} + f_n$, which we saw in the definition of the Fibonacci numbers. This resemblance turns out to be far from a coincidence. However, we will not discuss further about this for now, and treat the above lemma as just useful facts in computations.

Besides this naive observation, a strong connection between the Fibonacci numbers and the number φ can be seen more explicitly in the next theorem, which gives an explicit formula for the Fibonacci number f_n in terms of φ.

Theorem 3 *For every natural number* n, $f_n = \frac{1}{\sqrt{5}}(\varphi^n - (1-\varphi)^n)$.

Proof: We will prove this by an induction. First, as base cases, we will consider the cases when $n = 0$ and $n = 1$. It is easy to verify

$\frac{1}{\sqrt{5}}(\varphi^0 - (1-\varphi)^0) = \frac{1}{\sqrt{5}}(1-1) = 0 = f_0$,

$\frac{1}{\sqrt{5}}(\varphi^1 - (1-\varphi)^1) = \frac{1}{\sqrt{5}}(2\varphi - 1) = \frac{1}{\sqrt{5}}((1+\sqrt{5}) - 1) = 1 = f_1$,

so the statement is indeed true for $n = 0$ and $n = 1$.

Now, as an induction hypothesis, suppose that $f_i = \frac{1}{\sqrt{5}}(\varphi^i - (1-\varphi)^i)$ for all i such that $0 \leq i \leq k+1$. Using the previous lemma together with the definition of the Fibonacci numbers and the induction hypothesis, we compute

$f_{k+2} = f_{k+1} + f_k$

$= \frac{1}{\sqrt{5}}(\varphi^{k+1} - (1-\varphi)^{k+1}) + \frac{1}{\sqrt{5}}(\varphi^k - (1-\varphi)^k)$

$= \frac{1}{\sqrt{5}}((\varphi^{k+1} + \varphi^k) - ((1-\varphi)^{k+1} + (1-\varphi)^k))$

$= \frac{1}{\sqrt{5}}(\varphi^{k+2} - (1-\varphi)^{k+2})$

as desired. Thus, by the mathematical induction principle, we conclude that $f_n = \frac{1}{\sqrt{5}}(\varphi^n - (1-\varphi)^n)$ for all natural number n.

Corollary 4 *For every natural number n, f_n is the closest integer to $\frac{1}{\sqrt{5}}\varphi^n$.*

Proof: We saw $f_n = \frac{1}{\sqrt{5}}(\varphi^n - (1-\varphi)^n) = \frac{1}{\sqrt{5}}\varphi^n - \frac{1}{\sqrt{5}}(1-\varphi)^n$. We would like to show that the second term is small in absolute value. Indeed, it suffices to show that this second term is less than $\frac{1}{2}$ in absolute value. Note that $\frac{1}{\sqrt{5}} < \frac{1}{2}$ and $|1-\varphi| < 1$. Thus, we observe $|\frac{1}{\sqrt{5}}(1-\varphi)^n| = \frac{1}{\sqrt{5}}|1-\varphi|^n < \frac{1}{2} \cdot 1^n < \frac{1}{2}$.

It is clear from the above corollary that f_n grows in somewhat exponential manner, in accordance with φ^n. The deviation of f_n from $\frac{1}{\sqrt{5}}\varphi^n$ is measured by $|\frac{1}{\sqrt{5}}(1-\varphi)^n| = \frac{1}{\sqrt{5}}|1-\varphi|^n$. This decreases fairly quickly as n grows, since it is exponential. We should note f_n is slightly more than or less than $\frac{1}{\sqrt{5}}\varphi^n$, depending on whether n is odd or even.

Another interesting connection between the Fibonacci numbers and φ is that the positive powers of φ are always between two consecutive Fibonacci numbers. This, in turn, gives upper bounds and lower bounds for most of the Fibonacci numbers in terms of powers of φ. We shall state this fact in the following lemma.

Lemma 5 *For every integer n such that $n \geq 1$, we have $f_{n+1} < \varphi^n < f_{n+2}$.*

Proof: Again, we will prove this by an induction. First, as base cases, we will consider the cases when $n = 1$ and $n = 2$. Note that $1 < \varphi < 2$. By adding 1 to each term in the inequality, we obtain $2 < \varphi + 1 < 3$. The two inequalities together yield

$1 < \varphi < 2 < \varphi + 1 < 3$.

Using the lemma and the first few Fibonacci numbers, we can write this as

$f_2 < \varphi < f_3 < \varphi^2 < f_4$

which shows that the statement is true for $n = 1$ and $n = 2$.

Now, as the induction hypothesis, suppose that $f_{i+1} < \varphi^i < f_{i+2}$ for all i such that $0 \leq i \leq k+1$. In particular, we have

$f_{k+2} < \varphi^{k+1} < f_{k+3}$ and $f_{k+1} < \varphi^k < f_{k+2}$.
Adding each term of the two inequalities, we obtain
$f_{k+2} + f_{k+1} < \varphi^{k+1} + \varphi^k < f_{k+3} + f_{k+2}$.
Using the lemma and the definition of the Fibonacci numbers, we can rewrite this inequality as
$f_{k+3} < \varphi^{k+2} < f_{k+4}$
which shows that the inequality holds for $n = k+2$. Thus, by the mathematical induction principle, we conclude that $f_{n+1} < \varphi^n < f_{n+2}$ for every natural number n such that $n \geq 1$.

2 Euclidean Algorithm

The *greatest common divisor* of a and b, denoted by $\gcd(a, b)$, is defined for integers a and b that are not both zero. The Euclidean Algorithm is a well known algorithm for computing this $\gcd(a, b)$, and is based on the following lemma. Indeed, the Euclidean Algorithm is essentially just repeated applications of this lemma.

Lemma 6 *Let a be an integer and b be a positive integer. Then, we have* $\gcd(a, b) = \gcd(b, (a \bmod b))$

Proof: It suffices to show that the set of common divisors of a and b coincides with the set of common divisors of b and $(a \bmod b)$. Recall that the division theorem gives $a = bq + (a \bmod b)$.

Now, suppose d divides a and b. Then, d also divides $a - bq = (a \bmod b)$. Thus, a common divisor of a and b must be a common divisor of b and $(a \bmod b)$.

On the other hand, suppose d divides b and $(a \bmod b)$. Then, d divides $bq + (a \bmod b) = a$. Thus, a common divisor of b and $(a \bmod b)$ must be a common divisor of a and b.

Suppose we wish to compute $\gcd(a, b)$. Before describing the Euclidean Algorithm, we will make a few remarks on a few basic properties of the gcd which follows from the definition. Those properties are the following: (i) $\gcd(a, b) = \gcd(b, a)$, (ii) $\gcd(a, -b) = \gcd(a, b)$, and (iii) if $a > 0$ then $\gcd(a, 0) = a$. Those three properties together allows us to reduce the computation to the case where $a \geq 0$ and $b \geq 0$ with one of them nonzero. We will assume these conditions henceforth.

Now, we describe the Euclidean algorithm. First, if $b = 0$, then we know that $a > 0$ and $\gcd(a, b) = a$. If $b \neq 0$, then we apply the above lemma, and thus it follows that $\gcd(a, b) = \gcd(b, (a \bmod b))$. Notice that $b > (a \bmod b) \geq 0$, since $(a \bmod b)$ is indeed the remainder in the division theorem. So, if we apply the lemma repeatedly, the second arguments will strictly decrease each time in the repetition. Of course, the second argument eventually reaches zero. Then, the last $\gcd(\tilde{r}, 0)$ would be equal to the first argument $\tilde{r}$. Tracing back the series of equality, this number is indeed $\gcd(a, b)$.

This procedure eventually terminates because the repetition will stops when the strictly decreasing sequence of the nonnegative second arguments reaches zero. Also, each step is precisely defined computation. So it is, of course, an algorithm. Here are some examples.

$$\gcd(8,5) = \gcd(5,3) = \gcd(3,2) = \gcd(2,1) = \gcd(1,0) = 1$$
$$\gcd(70,28) = \gcd(28,14) = \gcd(14,0) = 14$$
$$\gcd(12,42) = \gcd(42,12) = \gcd(12,6) = \gcd(6,0) = 6$$
$$\gcd(0,91) = \gcd(91,0) = 91$$

The repeated application of the lemma in this algorithm makes it suitable for a recursive definition. Namely, we can view the Euclidean Algorithm as calling itself recursively in a specific manner until the second argument is zero. Here is one coded example of the definition of Euclidean algorithm. In this syntax, `%` is the modulus as usual, and `$_[`i`]` is the $(i+1)$-st element of the input array.

```
sub gcd {
if ($_[1]) { return gcd($_[1], $_[0] % $_[1]) }
else { return $_[0] }
}
```

3 Lamé's Theorem

In this section, we will investigate the worst case complexity of the Euclidean Algorithm. Theorem by Gabiel Lamé gives a way to relate the complexity of the Euclidean Algorithm to the Fibonacci numbers as we shall see presently. As we will see later in the corollary to the theorem, the complexity can indeed be calculated in terms of a logarithm in the Big-O notation.

Before stating the theorem, let us remark the following. First, the Euclidean Algorithm can be applied to any non-negative integers a and b such that not both zero. Obviously, we can further enforce the condition $a \geq b$ by choosing a to be the maximum of two numbers. Second, there are a few cases where the number of recursive calls in Euclidean Algorithm to find $\gcd(a,b)$ is obvious. If $b = 0$, then there is no recursive calls. If $a = b$, then $\gcd(a,a) = \gcd(a,0) = a$, so the Euclidean Algorithm requires exactly one call for any a. So, we may look at slightly more restrictive conditions $a \geq b > 0$ or $a > b \geq 0$. Under those conditions, it is redundant to state a and b are not both zero, since such case is impossible anyway.

In many books, the condition used is $a \geq b > 0$ for the Euclidean Algorithm. This is the case in Rosen, too. I shall state Lamé's theorem in accordance with this condition. Note that our version of Lamé's theorem is somewhat different from the one found in Rosen. The statement in Rosen indeed follows from our version of the theorem, as we will see in the corollary. For the simplicity, I will

use the notation $E(a, b)$ to mean the number of recursive calls required in the Euclidean algorithm to find $\gcd(a, b)$.

Theorem 7 (Lamé) *Suppose a, b be integers such that $a \geq b > 0$. For any natural number n, we have $E(a, b) < n$ whenever $b < f_{n+1}$ or $a < f_{n+2}$*

Proof: First, we remark that it suffices to prove the contrapositive statement. Namely, it is enough to prove that, for any natural number n, we have $b \geq f_{n+1}$ and $a \geq f_{n+2}$ whenever $E(a, b) \geq n$. We will do this by the mathematical induction.

First we consider the base case with $n = 0$. Suppose $E(a, b) \geq n = 0$. Since $b > 0$ from our premise, we actually know that $b \geq 1$ anyway. Also as a premise, we supposed $a \geq b$, so $a \geq 1$. Note that $1 = f_1 = f_2$. So, we indeed have $b \geq 1 = f_{0+1}$ and $a \geq 1 = f_{0+2}$. Namely, the statement holds for $n = 0$.

Now, as an induction hypothesis, suppose that our statement is true for $n = k$. Namely, suppose that $E(a.b) \geq k$ implies $b \geq f_{k+1}$ and $a \geq f_{k+2}$ for natural number k. We wish to show that the statement is true for $n = k + 1$. Namely, we wish to show $E(a, b) \geq k + 1$ implies $b \geq f_{k+2}$ and $a \geq f_{k+3}$.

Suppose $E(a, b) \geq k+1$. Note that this implies $E(a, b)$ is positive. Thus, we can indeed look at the result of the first recursive call. One recursive call gives $\gcd(a, b) = \gcd(a, a \bmod b)$. Let $\tilde{a} = b$ and $\tilde{b} = a \bmod b$. Note that $\gcd(\tilde{a}, \tilde{b})$ requires one less recursive calls than $\gcd(a, b)$. So, $\gcd(\tilde{a}, \tilde{b})$ requires $E(a, b) - 1$ recursive calls. Since $E(a, b) \geq k+1$ by our supposition, we have $E(a, b)-1 \geq k$. Then, by the induction hypothesis, it follows that $f_{k+1} \leq \tilde{b} = a \bmod b$ and $f_{k+2} \leq \tilde{a} = b$.

It remains to show that $a \geq f_{k+3}$. The division algorithm gives $a = bq + (a \bmod b)$. From our premise, we know $a \geq b$. It follows that $q \geq 1$ in the above equation of the division theorem. So, we have

$a = bq + (a \bmod b) \geq b + (a \bmod b) \geq f_{k+2} + f_{k+1} = f_{k+3}$.

This completes the proof.

Now, the next corollaries to the theorem allows us to express the worst case complexity of Euclidean Algorithm in terms of the Big-O notations and in terms of number of decimal digits. To state the result rigorously using the Big-O notation, there are a few things we need to be careful.

In the definition of the Big-O, we find that we can use the phrase $f(x)$ is $O(g(x))$ if f and g are functions from $\mathbb{R}$ or $\mathbb{Z}$ to $\mathbb{R}$. Since the Big-O essentially concerns with the relationship between the behaviours of $f(x)$ and $g(x)$ for arbitrarily large x, we can indeed extend the definition of the Big-O for functions f and g which are from $\{x \in \mathbb{R} \mid x \geq c\}$ or $\{x \in \mathbb{Z} \mid x \geq c\}$ to $\mathbb{R}$ where c is some real constant. We can also use $>$ in place of $\geq$ in the definition of the restricted domains.

Now, the number of recursive calls, $E(a, b)$, can be regarded as a function of a and b, where a and b are positive integers such that $a \geq b$. In order to make a Big-O statement with a as a variable, we need to somehow regard $E(a, b)$ as a function of a only. To do this, we can fix $b = b_0$. Then, for each $b_0 > 0$, $E(a, b_0)$

can be regarded as a function of a, where the domain is $\{a \in \mathbb{Z} \mid a \geq b_0\}$. By the above remark, then it make sense to discuss the Big-O of $E(a, b_0)$.

The corresponding consideration of $E(a, b)$ with respect to the variable b requires a bit more care, if we wish to talk about the Big-O notion with respect to b. If we fix $a = a_0$, then the domain of this function $E(a_0, b)$ of b would be $\{b \in \mathbb{Z} \mid a_0 \geq b > 0\}$. The problem is that b is not allowed to take values greater than this fixed a_0. This prevents us from making sense out of the Big-O notion. So, alternatively, we consider a sequence $\{a_b\}_{b=1}^{\infty}$ which satisfies $a_b \geq b$ for all b. For any such sequence, $E(a_b, b)$ is defined and can be regarded as a function of b, where the domain is $\{b \in \mathbb{Z} \mid b > 0\}$. Then, it makes sense to discuss the Big-O of $E(a_b, b)$.

Corollary 8 *The worst case complexity of $E(a, b)$ can be stated as follows:*

1) For any fixed integer $b_0 > 0$, $E(a, b_0)$ as a function of a is $O(\log a)$.

2) If $\{a_b\}_{b=1}^{\infty}$ is a sequence such that $a_b \geq b$ for each b, then $E(a_b, b)$ as a function of b is $O(\log b)$.

Proof: For the first statement, let $n = E(a, b_0)$ where a is some positive integer such that $a \geq b_0$. Then, by the contrapositive of the Lamé's theorem, we obtain $a \geq f_{n+2}$. Note that $f_{n+2} > \varphi^n$ from the previous lemma, so it follows that $a > \varphi^n$. Taking $\log_\varphi$ of both sides, we have

$\log_\varphi a > \log_\varphi \varphi^n = n = E(a, b_0)$

which indicates that $E(a, b_0)$ is indeed $O(\log a)$.

Now for the second statement, let $\{a_b\}_{b=1}^{\infty}$ be a sequence such that $a_b \geq b$ for each b. Now let $n = E(a_b, b)$ where b is some positive integer. Then, again by the contrapositive of the Lamé's theorem, we obtain $b \geq f_{n+1}$. Note that we have $f_{n+1} > \varphi^{n-1}$ from the previous lemma, so it follows that $b > \varphi^{n-1}$. Taking the $\log_\varphi$ of both sides, we have

$\log_\varphi b > \log_\varphi \varphi^{n-1} = (n-1)$,

thus,

$\log_\varphi b + 1 > n$.

Since the constant term 1 has no growth, this indicates that $E(a_b, b)$ is indeed $O(\log b)$.

Corollary 9 *The upper limit of $E(a, b)$ can be given as follows:*

1) $5d_a > E(a, b)$ where d_a is the number of decimal digits in a,

2) $5d_b \geq E(a, b)$ where d_b is the number of decimal digits in b.

Proof: Let a and b be integers such that $a \geq b > 0$. In the same manner as in the proof of the last corollary, we have

$a > \varphi^n$

$b > \varphi^{n-1}$.

Now, taking $\log_{10}$ of both sides, and using the fact that $\log_{10} \varphi > \frac{1}{5}$, we have

$\log_{10} a > \log_{10} \varphi^n = n \log_{10} \varphi > \frac{1}{5} n$

$\log_{10} b > \log_{10} \varphi^{n-1} = (n-1) \log_{10} \varphi > \frac{1}{5}(n-1)$.

Thus, by simple manipulation, we obtain

$5 \log_{10} a > n = E(a, b)$

$5 \log_{10} b + 1 > n = E(a, b)$.

The number of decimal digits in a and b are given by $d_a = \lfloor \log_{10} a \rfloor + 1$ and $d_b = \lfloor \log_{10} b \rfloor + 1$ respectively. We also know

$\log_{10} a = \lfloor \log_{10} a \rfloor + \varepsilon_a$ where $0 \leq \varepsilon_a < 1$

$\log_{10} b = \lfloor \log_{10} b \rfloor + \varepsilon_b$ where $0 \leq \varepsilon_b < 1$.

Thus, $d_a > \log_{10} a$ and $d_b > \log_{10} b$. Thus, combined with the above result, we have

$5d_a > 5 \log_{10} a > E(a, b)$

$5d_b + 1 > 5 \log_{10} b > E(a, b)$.

Note that $E(a, b)$ is an integer, so the second inequality indeed implies that $5d_b \geq E(a, b)$.

4 More on the Fibonacci Numbers

In the previous sections, we saw the number φ repeatedly. In particular, we had an explicit formula $f_n = \frac{1}{\sqrt{5}}(\varphi^n - (1 - \varphi)^n)$ for the Fibonacci numbers. It is natural to wonder how and from where this number φ arose. In this section, we will investigate the relationship between the Fibonacci numbers and the number φ. The material is essentially covered in §5.2 of Rosen. We will use linear algebra a little more explicitly.

Lemma 10 *Let A be a 2 by 2 matrix and $\mathbf{u}_0$ be a 2 by 1 matrix, given by*

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \quad \textit{and} \quad \mathbf{u}_0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

Suppose $\mathbf{u}_k$ is defined recursively by $\mathbf{u}_k = A\mathbf{u}_{k-1}$. Then, for all $n \in \mathbb{N}$, we have

$$\mathbf{u}_n = A^n \mathbf{u}_0 = \begin{bmatrix} f_{n+1} \\ f_n \end{bmatrix}$$

where f_n is the n-th Fibonacci number.

Proof: It is easy to verify the first equality $\mathbf{u}_n = A^n \mathbf{u}_0$. With the convention $A^0 = I$, the equality holds for the base case $n = 0$. As an induction hypothesis, suppose the equality holds for $n = k - 1$, namely $\mathbf{u}_{k-1} = A^{k-1}\mathbf{u}_0$; then, by the definition of $\mathbf{u}_k$ and the induction hypothesis, we have $\mathbf{u}_k = A(A^{k-1}\mathbf{u}_0) = A^k \mathbf{u}_0$. Thus, by the mathematical induction, the equality holds for all $n \in \mathbb{N}$.

We will show the remaining also by the mathematical induction. Again, as a base case, we consider the case when $n = 0$. This is trivial, as we see

$$\mathbf{u}_0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} f_1 \\ f_0 \end{bmatrix} = \begin{bmatrix} f_{0+1} \\ f_0 \end{bmatrix}.$$

Now, as an induction hypothesis, suppose the statement is true for $n = k - 1$. Namely, suppose

$$\mathbf{u}_{k-1} = \begin{bmatrix} f_k \\ f_{k-1} \end{bmatrix}.$$

Then, by the definition of $\mathbf{u}_k$ and the definition of the Fibonacci numbers, we have

$$\mathbf{u}_k = A\mathbf{u}_{k-1} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} f_k \\ f_{k-1} \end{bmatrix} = \begin{bmatrix} f_k + f_{k-1} \\ f_k \end{bmatrix} = \begin{bmatrix} f_{k+1} \\ f_k \end{bmatrix},$$

so the statement is true for the case $n = k$. Thus by the principle of mathematical induction, we conclude that the statement holds for all natural number n.

In the view of the equation above lemma, all the Fibonacci numbers can be derived from the powers of the matrix A. So, it would be of our interest to find a mean to compute A^k. This is where linear algebra comes into play. We will conclude this section by giving an alternative proof to the theorem from the previous section. In this proof, the number φ arises naturally as an "eigenvalue" of the matrix A.

Proof of Theorem 3: As observed in the last lemma, the second component of $\mathbf{u}_n = A^n\mathbf{u}_0$ is f_n. Using techniques from linear algebra, we can indeed compute A^n as follows.

First, we take the determinant of $A - \lambda I$. This yields a polynomial called *characteristic polynomial*. The actual computation gives

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

The associated *characteristic equation* is obtained by equating the characteristic polynomial with 0. So, in our case, it is $\lambda^2 - \lambda - 1 = 0$. The *eigenvalues* for the matrix A are indeed the roots of this equation. By the quadratic formula, we find $\lambda = \frac{1}{2}(1 \pm \sqrt{5})$. Namely, the eigenvalues of A are precisely φ and $1 - \varphi$. Then, we can find the *eigenvectors* corresponding each of the eigenvalues. We obtain

$$\begin{bmatrix} 1-\varphi & 1 \\ 1 & -\varphi \end{bmatrix} \begin{bmatrix} x_\varphi \\ y_\varphi \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \text{when} \quad \begin{bmatrix} x_\varphi \\ y_\varphi \end{bmatrix} = \begin{bmatrix} \varphi \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 1-(1-\varphi) & 1 \\ 1 & -(1-\varphi) \end{bmatrix} \begin{bmatrix} x_{1-\varphi} \\ y_{1-\varphi} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \text{when} \quad \begin{bmatrix} x_{1-\varphi} \\ y_{1-\varphi} \end{bmatrix} = \begin{bmatrix} 1-\varphi \\ 1 \end{bmatrix}$$

We would like to express $\mathbf{u}_0$ as a linear combination of those eigenvectors $\mathbf{v}_\varphi$ and $\mathbf{v}_{1-\varphi}$, so that the multiplication by any power of A would split nicely. Therefore, we need to solve a vector equation $\mathbf{u}_0 = \alpha\mathbf{v}_\varphi + \beta\mathbf{v}_{1-\varphi}$. In a matrix form, we have

$$\begin{bmatrix} \varphi & 1-\varphi \\ 1 & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

The 2×2 matrix that appear in the above equation is nonsingular since $\varphi \neq 1-\varphi$. So, we can solve this equation for α and β, and we find that $\alpha = \frac{1}{\sqrt{5}}$ and $\beta = -\frac{1}{\sqrt{5}}$.

We can now compute $\mathbf{u}_n$ very efficiently, using the definition of the eigenvalues and the corresponding eigenvectors, together with the linearity of the matrix multiplication. Computation yields

$\mathbf{u}_n = A^n \mathbf{u}_0$
$= A^n(\alpha \mathbf{v}_\varphi + \beta \mathbf{v}_{1-\varphi})$
$= A^n(\alpha \mathbf{v}_\varphi) + A^n(\beta \mathbf{v}_{1-\varphi})$
$= \alpha A^n \mathbf{v}_\varphi + \beta A^n \mathbf{v}_{1-\varphi}$
$= \alpha \varphi^n \mathbf{v}_\varphi + \beta(1-\varphi)^n \mathbf{v}_{1-\varphi}$.

The second component of this vector $\mathbf{u}_n$ is indeed f_n. Since the second components of $\mathbf{v}_\varphi$ and $\mathbf{v}_{1-\varphi}$ are both just 1, we compute

$f_n = \alpha \varphi^n y_\varphi + \beta(1-\varphi)^n y_{1-\varphi}$
$= \alpha \varphi^n + \beta(1-\varphi)^n$
$= \frac{1}{\sqrt{5}}(\varphi^n - (1-\varphi)^n)$

which is the formula we desired.

5 Towards the Generalization

The methods we used in the last section to find the explicit formula for the Fibonacci numbers can be applied to the more general class of sequences which contains the Fibonacci sequence. Such class of interest consists of sequences which are solutions to some *linear recurrence relations*, namely the sequences defined by some linear recurrence relation together with some initial values. Indeed, we could investigate the various properties of linear recurrence relations and its solutions using linear algebra.

The generalization shall allow us to tie many of our previous results on the Fibonacci numbers together in the larger context, and to develop better understanding of the Fibonacci numbers. As we saw the flavour of such theory already in this set of notes, linear algebra would be the essential tool for this approach.

The concrete results from the studies in this direction were introduced in §5.2 of Rosen. However, since the book does not assume readers' knowledge in linear algebra, many of the results were stated without proofs and the exposition does not provide the overview of the underlying theory in depth. If one does understand some of the elementary linear algebra, it may be advisable to seek some further exposure to the subject.