Problem

Given $x^5 = 1$ and $x \neq 1$, find

$$a = \left(\frac{1}{x^2 + x + 1} + \frac{1}{x^2 - x + 1}\right)^5 \tag{1}$$

Solution

Restating the problem in a more general form, given $x^n = 1$ and $x \neq 1$, find

$$a = \left(\frac{1}{f_k(x)} + \frac{1}{f_k(-x)}\right)^n \tag{2}$$

where

$$f_k(x) = 1 + x + x^2 + \ldots + x^k \tag{3}$$

The question implies (assumes) that a has the same value for all non-trivial n^{th} roots of unity for which the expression is well-defined. In particular, this requires the denominators to be non-zero.

Let the set of possible values of x be the set $X_{k,n}$ defined

$$X_{k,n} = \{x \in \mathbb{C} \cdot x^n = 1 \wedge x \neq 1 \wedge f_k(x) \neq 0 \wedge f_k(-x) \neq 0\} \tag{4}$$

Since $f_k(x)$ is a geometric progression and $x \neq 1$,

$$f_k(x) = \frac{1 - x^{k+1}}{1 - x} \tag{5}$$

Note $f_k(-x)$ will depend upon whether k is even or odd. Assuming $x \neq -1$,

$$f_k(-x) = \frac{1 - x^{k+1}}{1 + x}, \qquad \text{if } k \text{ is odd} \tag{6}$$

$$f_k(-x) = \frac{1 + x^{k+1}}{1 + x}, \qquad \text{if } k \text{ is even} \tag{7}$$

In both cases we will cross-multiply to make a common denominator, as per

$$\frac{1}{f_k(x)} + \frac{1}{f_k(-x)} = \frac{f_k(x) + f_k(-x)}{f_k(x) \times f_k(-x)} \tag{8}$$

Case: odd k

$$\frac{1}{f_k(x)} + \frac{1}{f_k(-x)} = \frac{\frac{1-x^{k+1}}{1-x} + \frac{1-x^{k+1}}{1+x}}{\frac{1-x^{k+1}}{1-x} \times \frac{1-x^{k+1}}{1+x}} \tag{9}$$

$$= \frac{1}{1 - x^{k+1}} \times \frac{\frac{1}{1-x} + \frac{1}{1+x}}{\frac{1}{1-x} \times \frac{1}{1+x}} \tag{10}$$

$$= \frac{2}{1 - x^{k+1}} \tag{11}$$

Case: even k

$$\frac{1}{f_k(x)} + \frac{1}{f_k(-x)} = \frac{\frac{1-x^{k+1}}{1-x} + \frac{1+x^{k+1}}{1+x}}{\frac{1-x^{k+1}}{1-x} \times \frac{1+x^{k+1}}{1+x}} \tag{12}$$

$$= \frac{(1+x)(1-x^{k+1}) + (1-x)(1+x^{k+1})}{1-x^{2k+2}} \tag{13}$$

$$= \frac{2-2x^{k+2}}{1-x^{2k+2}} \tag{14}$$

$$= 2 \times \frac{1-x^{k+2}}{1-x^{2k+2}} \tag{15}$$

In the original question, $n = 5$ and $k = 2$, and a factor of $(1-x)$ cancels out:

$$\frac{1}{f_2(x)} + \frac{1}{f_2(-x)} = 2 \times \frac{1-x^4}{1-x^6} \tag{16}$$

$$= 2 \times \frac{x(1-x^4)}{x(1-x)} \tag{17}$$

$$= 2 \times \frac{x-1}{x(1-x)} \tag{18}$$

$$= -\frac{2}{x} \tag{19}$$

Hence

$$a = \left(-\frac{2}{x}\right)^5 = -32 \tag{20}$$

Discussion

Are there solutions for other n, k? By solution, we mean the expression for a is the same regardless of non-trivial unity n-roots for which it is well defined.

It is not immediately obvious how to proceed. Perhaps Cyclotomic polynomials will help? In the meantime, Table 1 contains computer-generated (see Listing 1) combinations for different values of n and k. Shown are the combinations of n and k which result in a "valid answer" as explained above.

Some combinations don't allow for all the roots, such as $n = 6, k = 2$, which only has a single allowable root.

Possibly of more interest are the cases for which the number of valid roots equals $n - 1$ (i.e. all of them). These have been highlighted.

n	k	#valid roots
5	**2**	**4**
6	2	1
6	3	4
8	3	4
10	4	1
11	**6**	**10**
12	3	8
12	5	6
12	7	8
14	6	1
16	7	8
17	**10**	**16**
18	5	12
18	8	1
18	11	12
20	9	10
22	10	1
23	**14**	**22**
24	7	16
24	11	12
24	15	16
26	12	1
28	13	14
29	**18**	**28**
30	9	20
30	14	1
30	19	20
32	15	16
34	16	1
35	**22**	**34**
36	11	24
36	17	18
36	23	24
38	18	1

Table 1: Combinations of n, k which have a valid solution between $n = 3$ and $n = 40$.

Those highlighted have the maximum number of valid n-roots.

```
import cmath
import math

def non_trivial_roots_of_unity(n):
    # Excludes 1.
    return [
        complex(math.cos(2 * math.pi * i / n), math.sin(2 * math.pi * i / n))
        for i in range(1, n)
    ]


def f(k, x):
    # 1+x+x^2+...x^k (i.e. includes x^k)
    return sum(x**i for i in range(k + 1))


def answers(k, n, tolerance=1e-10):
    for x in non_trivial_roots_of_unity(n):
        f1 = f(k, x)
        f2 = f(k, -x)
        if cmath.isclose(f1, 0, abs_tol=tolerance):
            continue
        if cmath.isclose(f2, 0, abs_tol=tolerance):
            continue
        yield (1 / f1 + 1 / f2) ** n


def n_unique_answers(k, n, tolerance=1e-10):
    unique = []
    for x in answers(k, n):
        if not any(cmath.isclose(x, u, abs_tol=tolerance) for u in unique):
            unique.append(x)
    return len(unique)


def find_uniques(N):
    return [
        (k, n)
        for n in range(3, N)
        for k in range(2, n)
        if n_unique_answers(k, n) == 1
    ]
```

Listing 1: Code to find interesting combinations of n, k