Why we cannot do [arr = arr + 1;] in C++ ?

In C++, the name of an array is actually a constant pointer to the first element of the array. This means tat you cannot modify the value of the array pointer itself (i.e., the address of the first element of the array) by using pointer arithmetic.

So, if you try to do something like `arr = arr + 1`, you will get a compilation error because you are trying to modify a constant pointer. This is because pointer `arr` is pointing to the memory location of the first element of the array, and you cannot change this address.

However, you can use pointer arithmetic to access other elements of the array. For example, you can use `arr + 1` to get a pointer to the second element of the array, like this:

```
int arr[5] = { 1, 2, 3, 4, 5 };

int *ptr = arr; // pointer to the first element of the array

ptr++; // pointer now points to the second element of the array

cout << *ptr << endl; // prints 2
```

In this example, `ptr` is initially pointing to the first element of the array `arr`. We then use the pointer arithmetic `ptr++` to increment the pointer to point to the second element of the array. Finally, we dereference the pointer using `*ptr` to get the value of the second element of the array.

So, in summary, you cannot modify the value of the array pointer itself using pointer arithmetic, but you can use pointer arithmetic to access other elements of the array.