Euclidean algorithm

These notes give an alternative, recursive presentation of the Euclidean algorithm for calculating the GCD of two non-negative integers (Algorithms 2.3.4 and 2.3.7 in the course notes). The recursive versions are simpler to describe and prove correct. In practice, that is, if you were to write computer programs for these algorithms, the iterative versions in the course notes would run faster.

Recall the following property of GCDs:

Proposition 1 *For any integers a, b, q, r with $a = qb + r$, $\gcd(a, b) = \gcd(b, r)$.*

The Euclidean algorithm divides the larger number, say a by the smaller one, say b, to get $a = qb+r$, where r is the remainder ($0 \leq r < b$), and then applies the above principle. This is repeated until the GCD of the two numbers is obvious.

Example 1 *Compute $\gcd(2247, 973)$, using the Euclidean algorithm.*

Solution : We collect our calculations in the following table.

Step	a	b	division			r
1	2247	973	2247	$=$	$2 \times 973 + 301$	301
2	973	301	973	$=$	$3 \times 301 + 70$	70
3	301	70	301	$=$	$4 \times 70 + 21$	21
4	70	21	70	$=$	$3 \times 21 + 7$	7
5	21	7	21	$=$	$3 \times 7 + 0$	0
6	7	0				

Finally, the algorithm returns $\gcd(2247, 973) = \gcd(7, 0) = 7$. ■

We can summarise this method with the following recursive description.

Algorithm 1: EUCLID(a, b), a recursive algorithm to compute GCDs

input : Non-negative integers a, b such that $b \leq a$
output: $\gcd(a, b)$

if $b = 0$ **then**
  return a;
else
  Divide a by b to get $a = qb + r$, where r is the remainder;
  return EUCLID(b, r);

Due to Proposition 1, it is intuitively clear that the algorithm is correct. We may prove this rigorously by strong induction.

Note: It is not clear at the outset on which parameter we should perform the induction, or that we should use induction in the first place. The use of recursion indicates that induction would be a natural proof method—it is geared precisely towards proving properties of (positive) integers given that the property holds for smaller integers. Moreover, observe that in every recursive call of the algorithm, the second input r is necessarily smaller than the original second input b. The first input may have the same value; this happens when $b = a$. So we choose to conduct the induction based on the second input to the algorithm.

Theorem 2 *The algorithm* EUCLID(a, b) *returns* $\gcd(a, b)$*, for any non-negative integers a, b such that $b \leq a$.*

Proof : The proof is by induction on $b = \min\{a, b\}$.

Base case: $b = 0$. In this case, the algorithm returns the correct value, as $\gcd(a, 0) = a$ for any non-negative integer a.

Induction hypothesis: assume that $\text{EUCLID}(c, d) = \gcd(c, d)$ for any non-negative integers c, d such that $0 \leq d \leq n$.

Inductive step: Consider a, b, such that $\min\{a, b\} = b = n + 1$. $\text{EUCLID}(a, b)$ returns $\text{EUCLID}(b, r)$. By the Division Algorithm, the remainder r satisfies $0 \leq r < b = n + 1$, so $0 \leq r \leq n$. By the induction hypothesis, $\text{EUCLID}(b, r) = \gcd(b, r)$, which is equal to $\gcd(a, b)$ by Proposition 1. ■

The GCD of two numbers satisfies an important property that it may be written as an integer combination of the numbers.

Theorem 3 (Bézout Lemma) *Let a, b be two non-negative integers, and let $d = \gcd(a, b)$. Then there are integers x, y such that $d = ax + by$.*

This property follows directly from the Euclidean algorithm, and the numbers x, y as above may be computed using our calculations for the GCD. Suppose the number returned in the else branch of $\text{EUCLID}(a, b)$ is $d = \gcd(b, r)$, and that we have found integers u, v such that $d = ub + vr$. Since $r = a - qb$, we can express d as a combination of a, b. Indeed, $d = ub + v(a - qb) = va + (u - vq)b$, so we may take $x = v$ and $y = u - vq$. We may modify the algorithm to keep track of these integers.

Algorithm 2: EXT-EUCLID(a, b), the extended Euclidean algorithm

input : Non-negative integers a, b such that $b \leq a$
output: (d, x, y) such that $d = \gcd(a, b)$ and $ax + by = d$

if $b = 0$ **then**
  return (a,1,0);
else
  Divide a by b to get $a = qb + r$, where r is the remainder;
  Let $(d, u, v) =$ EXT-EUCLID(b, r);
  Let $x = v$, and $y = u - vq$;
  return (d, x, y);

Let us see how this works for the numbers in Example 1.

Example 2 *Compute numbers x, y such that $2247x + 973y = \gcd(2247, 973) = 7$.*

Solution : Starting with the table in Example 1, we work up from the last row, writing $7 = ax + by = 7(1) + 0(0)$. The pair x, y in row i form the pair u, v for the *previous* row $(i - 1)$. We now compute the pair x, y for row $(i - 1)$, and continue this way until we reach row 1.

Step	a	b	division: $a = bq + r$	r	u	v	$x = v$	$y = u - vq$
1	2247	973	$2247 = 2 \times 973 + 301$	301	13	-42	-42	97
2	973	301	$973 = 3 \times 301 + 70$	70	-3	13	13	-42
3	301	70	$301 = 4 \times 70 + 21$	21	1	-3	-3	13
4	70	21	$70 = 3 \times 21 + 7$	7	0	1	1	-3
5	21	7	$21 = 3 \times 7 + 0$	0	1	0	0	1
6	7	0					1	0

The algorithm returns $(7, -42, 97)$, and we may verify that $7 = (-42)2247 + (97)973$. ■

We may prove by strong induction that the algorithmEXT-EUCLID(a, b) correctly returns a triple (d, x, y) such that $d = \gcd(a, b)$ and $ax + by = d$, whenever $b \leq a$. The argument is similar to that for Theorem 2, and is left as an exercise. The Bézout Lemma follows directly from the correctness of the algorithm.