Computational Mathematics: Handout 11

Curtis Bright

December 7, 2022

1 A Modular Euclidean Algorithm

In this handout we cover a modular version of the Euclidean algorithm. This provides a way to control the coefficient growth of the Euclidean algorithm of polynomials over coefficient fields like $\mathbb{Q}$. Note that applying the usual Euclidean algorithm on polynomials with coefficients in $\mathbb{Q}$ typically causes a great increase in the size of the numerators and denominators of the intermediate coefficients used in the algorithm (and in the coefficients of the $s, t \in \mathbb{Q}[x]$ provided by the extended Euclidean algorithm, which typically explode in size even when run on coprime $a, b \in \mathbb{Q}[x]$ with small integer coefficients).

[1]:
```
# An example demonstrating the coefficient growth that occurs in the Euclidean␣
↪algorithm in Q[x]
F.<x> = QQ[]
a = F(random_vector(ZZ, 10, 10).list())
b = F(random_vector(ZZ, 10, 10).list())
g, s, t = xgcd(a,b)
print(a)
print(b)
print(s)
```

```
3*x^9 + 7*x^8 + 4*x^7 + 8*x^6 + 2*x^5 + 3*x^4 + 4*x^3 + 6*x^2 + 9*x
5*x^9 + 7*x^8 + 6*x^7 + 3*x^6 + 4*x^5 + 7*x^4 + x^3 + 4*x^2
6443632968160/118131505340139*x^7 - 34866263779/6217447649481*x^6 -
958350531281/39377168446713*x^5 - 214215554689/39377168446713*x^4 +
5207481762998/118131505340139*x^3 + 794474335838/118131505340139*x^2 -
5285189195002/118131505340139*x + 1/9
```

Additionally, the modular approach also works in $\mathbb{Z}[x]$ (not just $\mathbb{Q}[x]$).

1.1 GCDs in $\mathbb{Z}[x]$

We've seen that the Euclidean algorithm does not work in $\mathbb{Z}[x]$ since $\mathbb{Z}$ is not a field. A priori it is not even clear if the concept of GCD makes sense in $\mathbb{Z}[x]$ as not every ring has unique factorization. An example of a ring that does not have unique factorization (and therefore does not have GCDs) is the polynomial ring $\mathbb{Z}[x]$ with arithmetic performed modulo $x^2 - 3$ (typically denoted $\mathbb{Z}[x]/\langle x^2 - 3\rangle = \{a + b\sqrt{3} : a, b \in \mathbb{Z}\}$).

Disregarding this, a theorem of Gauss implies that GCDs do in fact exist in $\mathbb{Z}[x]$ and we will develop an algorithm to compute them.

1.1.1 Irreducible polynomials

A polynomial f in $\mathbb{Z}[x]$ is called *irreducible* if it cannot be factored any further in $\mathbb{Z}[x]$, i.e., the decomposition $f = gh$ must be trivial (one of $g, h \in \mathbb{Z}[x]$ is invertible and thus ± 1).

For example, $x^2 - 1$ is not irreducible, since it factors as $(x-1)(x+1)$.

Note that the irreducibility of a polynomial can depend on its coefficient ring. For example, $x^2 - 2$ is irreducible over $\mathbb{Z}$ but not over $\mathbb{R}$. Conversely, $2x + 2$ is not irreducible over $\mathbb{Z}$ as it factors as $2 \cdot (x+1)$ which is nontrivial in $\mathbb{Z}$ (neither factor is invertible). However, $2x + 2$ is irreducible over $\mathbb{R}$, since the factorization $2(x+1)$ is trivial over $\mathbb{R}$ as 2 is invertible in $\mathbb{R}$.

1.1.2 Gauss' lemma

A polynomial is called *primitive* if the greatest common divisor of its coefficients is 1.

For example, $6x^2 + 2x + 3$ is primitive as $\gcd(6, 2, 3) = 1$ but $6x + 3$ is not primitive as $\gcd(6, 3) = 3$.

A property of integer polynomials proven by Gauss is that the product of two primitive polynomials is also a primitive polynomial.

Furthermore, a nonconstant polynomial f is irreducible (over $\mathbb{Z}$) if and only if f is primitive and f is irreducible (over $\mathbb{Q}$).

In other words, for nonconstant primitive polynomials irreducibility over $\mathbb{Z}$ and irreducibility over $\mathbb{Q}$ correspond exactly.

These properties are known as Gauss' lemmas and using them it follows that $\mathbb{Z}[x]$ has unique factorization because $\mathbb{Q}[x]$ has unique factorization. More generally, if R has unique factorization then $R[x]$ also has unique factorization.

1.1.3 Simplifying assumption

Say $f, g \in \mathbb{Z}[x]$ and we want to compute $\gcd(f, g)$ over $\mathbb{Z}$. It is not a restrictive assumption to assume that f and g are primitive, because if they were not it is easy to compute their "primitive parts" by dividing through by the greatest common divisor of their coefficients first.

Let $\operatorname{pp}(f)$ be defined to be $f / \gcd(f_0, f_1, \ldots, f_n)$. In order to compute the GCD of f and g it sufficies to compute the GCD of the "non-primitive" parts (i.e., $\gcd(f_0, f_1, \ldots, f_n, g_0, g_1, \ldots, g_m)$) and the GCD of the primitive parts $\operatorname{pp}(f)$ and $\operatorname{pp}(g)$. Thus, from now on we will assume that f and g are primitive. By Gauss' lemma this also implies their product is primitive and $\operatorname{pp}(fg) = \operatorname{pp}(f) \cdot \operatorname{pp}(g)$.

1.1.4 Computing GCDs in $\mathbb{Z}[x]$ via GCDs in $\mathbb{Q}[x]$

As stated above, we assume that $f, g \in \mathbb{Z}[x]$ are primitive and we want to compute their GCD over $\mathbb{Z}$. We already know how to compute their GCD over $\mathbb{Q}$ using the Euclidean algorithm.

Let $v := \gcd_{\mathbb{Q}[x]}(f, g)$ be the result of applying Euclid's algorithm. As we previously saw, by construction v will be *monic*, i.e., have a leading coefficient of 1. However, its other coefficients will very likely be over $\mathbb{Q}$ and not over $\mathbb{Z}$; thus it is not acceptable as a GCD over $\mathbb{Z}$.

Corollary 6.10 in Modern Computer Algebra states that if h is the GCD of f and g over $\mathbb{Z}$ then h is primitive and

$$h/\operatorname{lc}(h) = v \qquad \text{where } \operatorname{lc}(h) \text{ is the leading coefficient of } h.$$

Thus, we need to multiply v by $\operatorname{lc}(h)$ in order to compute h. Of course, we don't know $\operatorname{lc}(h)$ since we don't know h. However, we can find a multiple of $\operatorname{lc}(h)$. Because h divides f and g (by definition it is the largest divisor) it also follows that $\operatorname{lc}(h)$ divides $\operatorname{lc}(f) = f_n$ and $\operatorname{lc}(g) = g_m$ and thus also $\gcd(f_n, g_m)$.

It follows $\gcd(f_n, g_m) \cdot v$ is an integer polynomial which is a constant multiple of h. It may be a nontrivial multiple (introducing a nonprimitive part) but in such a case we can just take its primitive part as h must be primitive.

In summary, when f and g are primitive we have

$$\gcd_{\mathbb{Z}[x]}(f, g) = \operatorname{pp}\bigl(\gcd(f_n, g_m) \cdot \gcd_{\mathbb{Q}[x]}(f, g)\bigr).$$

1.1.5 Example

Suppose $\tilde{f} := 30x^3 - 10x^2 + 30x - 10$ and $\tilde{g} := 6x^2 - 14x + 4$.

Since $\gcd(30, -10, 30, -10) = 10$ and $\gcd(6, -14, 4) = 2$ we can divide $\tilde{f}$ by 10 and $\tilde{g}$ by 2 to obtain their primitive parts and take $\gcd_{\mathbb{Z}[x]}(\tilde{f}, \tilde{g}) = 2\gcd_{\mathbb{Z}[x]}(\tilde{f}/10, \tilde{g}/2)$.

Now suppose $f := \tilde{f}/10 = 3x^3 - x^2 + 3x - 1$ and $g := \tilde{g}/2 = 3x^2 - 7x + 2$. We can compute $\gcd(f, g)$ over $\mathbb{Q}$ as $x - 1/3$:

[2]:
```
R.<x> = QQ[]
f = 3*x^3-x^2+3*x-1
g = 3*x^2-7*x+2
gcd(f,g)
```

[2]:
```
x - 1/3
```

Furthermore, the leading coefficients of f and g is $f_3 = g_2 = 3$, so $\gcd(f_3, g_2) = 3$.

It follows that $\gcd_{\mathbb{Z}[x]}(f, g) = \operatorname{pp}(3 \cdot (x - 1/3)) = \operatorname{pp}(3x - 1) = 3x - 1$ and $\gcd_{\mathbb{Z}[x]}(\tilde{f}, \tilde{g}) = 2(3x - 1) = 6x - 2$.

[3]:
```
R.<x> = ZZ[]
ftilde = 30*x^3-10*x^2+30*x-10
gtilde = 6*x^2-14*x+4
gcd(ftilde,gtilde)
```

[3]:
```
6*x - 2
```

1.2 Reducing modulo p

The idea behind the modular GCD algorithm is that will reduce the coefficients of f and g modulo a prime p, perform Euclid's algorithm on f, g (as elements of $\mathbb{F}_p[x]$), and recover $h := \gcd_{\mathbb{Z}[x]}(f, g)$ from $\gcd_{\mathbb{F}_p[x]}(f, g)$. In order for the recovery to work correctly p must be large enough so that all of the true (non-reduced) coefficients of h lie in the range $\{-\frac{p-1}{2}, \ldots, \frac{p-1}{2}\}$. This is the "symmetric" representation of $\mathbb{F}_p$ and it is used instead of the standard representation (that is, $\{0, \ldots, p-1\}$) because h may have negative coefficients.

However, some primes p cause problems with this approach. For example, consider $p = 3, 5, 7$ and computing $\gcd_{\mathbb{F}_p[x]}(f, g)$ for the above primitive polynomials $f := 3x^3 - x^2 + 3x - 1 = (x^2 + 1)(3x - 1)$ and $g := 3x^2 - 7x + 2 = (x - 2)(3x - 1)$.

[4]:
```
for p in [3, 5, 7]:
    F.<x> = GF(p)[]
    f = 3*x^3-x^2+3*x-1
    g = 3*x^2-7*x+2
    print("gcd_F{}[x](f, g) = {}".format(p, gcd(f,g)))
```

```
gcd_F3[x](f, g) = 1
gcd_F5[x](f, g) = x^2 + x + 4
gcd_F7[x](f, g) = x + 2
```

We have the following:

$$\begin{aligned} \gcd_{\mathbb{F}_3[x]}(f, g) &= 1 \\ \gcd_{\mathbb{F}_5[x]}(f, g) &= x^2 + x - 1 \\ \gcd_{\mathbb{F}_7[x]}(f, g) &= x + 2 \end{aligned}$$

Note that in the last case ($p = 7$) the algorithm works correctly: $\gcd(\mathrm{lc}(f), \mathrm{lc}(g)) \cdot \gcd_{\mathbb{F}_7[x]}(f, g) \equiv 3(x + 2) \equiv 3x - 1 \pmod 7$ is the true GCD of f and g over $\mathbb{Z}$.

However, in the first two cases ($p = 3, 5$) the algorithm does not work correctly, as the degree of $\gcd_{\mathbb{F}_p[x]}(f, g)$ is not correct (too small when $p = 3$ and too large when $p = 5$). What is going on here?

1.2.1 A criteria for nontrivial GCDs

Suppose F is a field and $f, g \in F[x]$ and $\gcd(f, g) = h$ over F. Recall that Euclid's algorithm allows us to find $s, t \in F[x]$ with $sf + tg = h$.

If $h \neq 1$ then there is a nontrivial solution to the equation

$$sf + tg = 0 \text{ with } \deg(s) < \deg(g) \text{ and } \deg(t) < \deg(f). \qquad (*)$$

Namely, one can take $s := g/h$ and $t := -f/h$. In fact, the existence of such a (s, t) provide a *certificate* that $\gcd(f, g)$ is nontrivial (see lemma 6.13 in Modern Computer Algebra).

Thus, equation $(*)$ can be used to determine if $\gcd(f, g)$ is trivial or nontrivial; if $(*)$ has a solution then $\gcd(f, g) \neq 1$ and if $(*)$ has no solution then $\gcd(f, g) = 1$.

Note that (∗) can equivalently be written as the following matrix-vector product equation:

$$\begin{bmatrix} f_n & & & & g_m & & & & \\ f_{n-1} & f_n & & & \vdots & g_m & & & \\ \vdots & f_{n-1} & \ddots & & g_0 & \vdots & g_m & & \\ f_1 & \vdots & & f_n & & g_0 & \vdots & g_m & \\ f_0 & f_1 & & f_{n-1} & & & g_0 & \vdots & \ddots & \\ & f_0 & & \vdots & & & & g_0 & & g_m \\ & & \ddots & f_1 & & & & & \ddots & \vdots \\ & & & f_0 & & & & & & g_0 \end{bmatrix} \begin{bmatrix} s_{m-1} \\ s_{m-2} \\ \vdots \\ s_0 \\ t_{n-1} \\ t_{n-2} \\ \vdots \\ t_0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \in F^{n+m}$$

Note $\deg(f) = n$ and $\deg(g) = m$ and the ith row of this expression corresponds to the coefficient of the x^{n+m-i} and hence why the right-hand side contains all zeros, as there are no terms x^{n+m-i} on the right-hand side of (∗). The matrix in this expression is known as the *Sylvester* matrix of f and g.

Linear algebra then tells us that $\gcd(f, g) \neq 1$ if and only if the Sylvester matrix of f and g is singular (i.e., there is a nontrivial solution of this matrix-vector equation).

The determinant of the Sylvester matrix of f and g is known as the *resultant* $\operatorname{res}(f, g)$. A matrix is singular if and only if its determinant is 0, so we can equivalently state this as

$$\gcd(f, g) \neq 1 \iff \operatorname{res}(f, g) = 0.$$

The above takes place over a field F but due to Gauss' theorem it can also be modified to work over $\mathbb{Z}$:

$$\gcd_{\mathbb{Z}[x]}(f, g) \text{ is nonconstant} \iff \operatorname{res}(f, g) = 0.$$

1.2.2 A criteria for choosing primes

The reason we introduced the resultant is because it allows an easy specification of the primes p for which the GCD over $\mathbb{F}_p$ can be used to find the GCD over $\mathbb{Z}$.

Theorem (6.26, Modern Computer Algebra) Let $f, g \in \mathbb{Z}[x]$ be nonzero and of degrees n and m, let $h = \gcd(f, g)$ over $\mathbb{Z}$, and let p be a prime that does not divide $\gcd(f_n, g_m)$.

Then $\deg \gcd_{\mathbb{F}_p[x]}(f, g) \geq \deg h$ (i.e., polynomials might split "deeper" modulo p, causing $\gcd(f, g)$ over $\mathbb{F}_p$ to be larger than the $\gcd(f, g)$ over $\mathbb{Z}$.)

Moreover, the degree of $\gcd(f, g)$ over $\mathbb{F}_p$ will be **equal** to the degree of $\gcd(f, g)$ over $\mathbb{Z}$ if and only if p does not divide $\operatorname{res}(f/h, g/h)$.

Furthermore, p does not divide $\operatorname{res}(f/h, g/h)$ exactly when $\gcd_{\mathbb{F}_p[x]}(f, g) \equiv h/\operatorname{lc}(h) \pmod{p}$. (The inverse of $\operatorname{lc}(h)$ mod p exists since $\operatorname{lc}(h)$ divides $\gcd(f_n, g_m)$ which does not have p as a divisor.)

1.3 The modular algorithm for GCDs

So the prime p must satisfy the following:

1. p does not divide $\gcd(f_n, g_m)$
2. p does not divide $\operatorname{res}(f/h, g/h)$
3. The coefficients of $\gcd(f_n, g_m) \cdot h/\operatorname{lc}(h)$ have absolute value at most $(p-1)/2$ so they fit in the symmetric range

If p satisfies all three conditions then we can compute h, the $\gcd(f, g)$ over $\mathbb{Z}$, by:

- Using the Euclidean algorithm to compute $\gcd(f, g)$ over $\mathbb{F}_p$
- Multiplying this computed GCD by $\gcd(f_n, g_m)$ and reduce the coefficients to be in the symmetric range modulo p
- Return the primitive part of the above polynomial

1.3.1 Example

Let's compute the integer GCD of $f := 3x^3 - x^2 + 3x - 1 = (x^2+1)(3x-1)$ and $g := 3x^2 - 7x + 2 = (x-2)(3x-1)$ using this approach.

First, p must not divide $\gcd(f_3, g_2) = \gcd(3, 3) = 3$. Thus $p \neq 3$

Recall that $f/h = x^2 + 1$ and $g/h = x - 2$, and the Sylvester matrix of these two polynomials is

$$\begin{bmatrix} 1 & 1 & 0 \\ 0 & -2 & 1 \\ 1 & 0 & -2 \end{bmatrix}$$

which has determinant $(-2)^2 + 1 = 5$. Thus $p \neq 5$.

The coefficients of $\gcd(f_3, g_2) \cdot (3x-1)/3 = 3x - 1$ have absolute value at most 3, so we must have $(p-1)/2 \geq 3$, i.e., $p \geq 7$.

Thus, the simplest selection is $p = 7$.

As we saw above, Euclid's algorithm computes $\gcd_{\mathbb{F}_7[x]}(f, g) = x + 2$. We multiply this by $\gcd(f_3, g_2) = 3$ to obtain $3x + 6$ which when reduced to the symmetric range is $3x - 1$ which is already primitive.

1.3.2 Caveats

One unrealistic part of this example: the conditions on p involved h so in order to properly select p we are required to know $h = \gcd(f, g)$ over $\mathbb{Z}$. But *that's the very thing we are trying to compute!*

How can we get around this?

We could derive an upper bound on $\operatorname{res}(f/h, g/h)$ and then select p to be larger than this. However, this is very wasteful in practice and tends to use a prime p much larger than necessary. So we will ignore the resultant condition for now.

What about the sizes of the coefficients of h? It can be shown that the maximum coefficient of h has absolute value at most $\sqrt{n+1} \cdot 2^n A$ where A is an upper bound on the coefficients of f and g.

Thus if we choose a prime larger than $B := 2\gcd(f_n, g_m)\sqrt{n+1} \cdot 2^n A$ then we can guarantee that all coefficients of h will be bounded in absolute value by $(p-1)/2$.

It can also be shown that if you choose a random prime between B and $2B$ then p will not divide $\gcd(f/h, g/h)$ with probability at least 1/2. In other words, it shouldn't be hard to find a prime that works.

How can you tell if a prime works? The simplest approach is simply to verify that the purported GCD is actually a divisor of both f and g. If so, it follows that p does not divide $\operatorname{res}(f/h, g/h)$. Why? Because if p did divide $\operatorname{res}(f/h, g/h)$ then by Thm 6.26 the degree of $\gcd_{\mathbb{F}_p[x]}(f, g)$ will be strictly larger than the true GCD h. In such a case $\gcd_{\mathbb{F}_p[x]}(f, g)$ cannot possibly divide both f and g (over $\mathbb{Z}$) because then it would also have to divide their GCD h which is nonsensical given that $\gcd_{\mathbb{F}_p[x]}(f, g)$ has a larger degree than h.