Resultant theory. A brief overview.

December 26, 2017

Recently while I working on my research a came across a problem. I was faced with a number of multivariate systems and there are two things I need to know for these systems,

- do the polynomials of each system have common roots?
- and if yes could we identify them;

In order to address these questions and move on with my research it was decided that I was going down the line of the resultant theory! The resultant initially help us identify whether the system has a common root and furthermore we could use resultant theory to find those roots as well. Here I will only talk about the first usefulness of the resultants.

If p and q are two polynomials the can be factored into linear factors,

$$\begin{aligned} p(x) = &\quad a_0(x - r_1)(x - r_2)\cdots(x - r_m) \\ q(x) = &\quad b_0(x - s_1)(x - s_2)\cdots(x - s_n) \end{aligned}$$

then the **resultant** R of p and q is defined as,

$$R = a_0^n b_0^m \prod_{i=1}^{m}\prod_{j=1}^{n}(r_i - s_j)$$

From the definition, it is clear that the resultant will equal zero if and only if p and q have at least one common root. Thus if we want to know if common roots exist we only have to calculate the resultant and test whether it is equal to zero.

Calculating the resultant in this way can be expensive. An explicit formula for the resultant as a determinant was given by Sylvesterin 1840 `http://www.tandfonline.com/doi/abs/10.1080/14786444008649995`.

Let us consider that p and q are,

$$\begin{aligned} p(x) = &\quad \sum_{i=0}^{m} a_i x^i \\ q(x) = &\quad \sum_{i=0}^{n} b_i x^i \end{aligned}$$

where $deg(p) = m$and $deg(q) = n$. The Sylvester matrix of p and q is the $(m + n) \times (m + n)$ matrix,

$$\begin{vmatrix}
a_0 & a_1 & a_2 & \dots & a_m & 0 & \dots & 0 \\
0 & a_0 & a_1 & \dots & a_{m-1} & a_m & \dots & 0 \\
 & & \ddots & . & . & . & . & . \\
0 & 0 & . & . & . & . & . & a_m \\
b_0 & b_1 & b_2 & \dots & b_n & 0 & \dots & 0 \\
0 & b_0 & b_1 & \dots & b_{n-1} & b_n & \dots & 0 \\
 & & \ddots & . & . & . & . & . \\
0 & 0 & . & . & . & . & . & b_n
\end{vmatrix}$$

in which there are n rows of p coefficients, m rows of q coefficients, and all elements not shown are zero. The resultant is the determinant of the Sylvester matrix.

The Sylvester's resultant is implemented in the python library Sympy. Thus here I will give you a numerical example. Note that I have also implemented the Sylvester's resultant and my code can be found here: but for the example I will be using the implemented version of Sympy.

```
>>> import sympy as sym
>>> from sympy.polys import subresultants_qq_zz

>>> x = sym.symbols('x')

>>> f = x ** 2 - 5 * x + 6
>>> g = x ** 2 - 3 * x + 2

>>> matrix = subresultants_qq_zz.sylvester(f, g, x)
>>> matrix
Matrix([
[1, -5,  6, 0],
[0,  1, -5, 6],
[1, -3,  2, 0],
[0,  1, -3, 2]])
>>> matrix.det()
0
```

Note that the Sylvester's resultant is very useful but can only handle up to 2 variables. An alternative matrix formulation was given by Bezout during the eighteenth century. The Bezout formulation was then reformulated by Caley in 1865 and this is the second formulation we will discuss here. I will calling it the Bezout Caley formulation.

Let us consider again the univariate polynomials p and q. Let the $d_{\max} = max(degree(p, x), degree(q, x))$. Consider the polynomial,

$$\Delta(x, a) = \begin{vmatrix} p(x) & q(x) \\ p(a) & q(a) \end{vmatrix}$$

where α is a new variable and $p(a)$ stands for uniformly replacing x by α in p. Making $x = \alpha$ would make $\Delta = O$ which means that $x\alpha$ divides Δ. Thus the Bezout Cayley polynomial of degree $d_{\max} - 1$is defined as,

$$\delta(x, a) = \frac{\Delta(x, a)}{x - a}$$

The polynomial is symmetric in x and α. Every common zero of $p(x)$ and $q(x)$ is a zero of $\delta(x, \alpha$ no matter what value α has; thus at a common zero of p and q, the coefficient of every power product of α in $\delta(x, \alpha$ must be O.

Thus we have a $d_{max}-1$ equations by equating the coefficients of the power products of α to zero. Treating $x^0, x^1, \ldots, x^{d_{max}-1}$ as unknowns we retrieve $d_{max}-1$ equations in $d_{max}-1$ unknowns. They will have a common root if and only if the the determinant of the coefficient matrix is equal to 0, this $d_{max} \times d_{max}$ matrix is the Bezout Cayley matrix.

A numerical example:

```
>>> f = sym.lambdify(x, x ** 2 - 5 * x + 6)
>>> g = sym.lambdify(x, x ** 2 - 3 * x + 2)

>>> matrix = subresultants_qq_zz.bezout(f(x), g(x), x)
>>> matrix
Matrix([
[ 8, -4],
[-4,  2]])
>>> matrix.det()
0
```