Elimination for polynomial systems of equations and applications to high dimensional approximation, quadrature and other numerical problems

Markus Hegland, ANU

Mathematical Sciences Institute, ANU

1. polynomial systems of equations
2. elimination for polynomial systems
3. examples of polynomial equations
4. computational challenges

1. polynomial systems of equations

system of equations

- aim: compute $x \in \mathbb{R}^d$ which solve

$$p(x) = 0$$

- $p : \mathbb{R}^d \to \mathbb{R}^n$ polynomial of degree k
- iterative methods
- there are elimination based direct solvers

example: polynomial systems of degree one

- polynomial of degree $k = 1$

$$p(x) = Ax - b$$

  - solution theory:
    - exactly one solution ($n = m$, A regular)
    - no solution
    - solution set is an affine space

solving linear systems

- direct methods: Gaussian elimination

$$PA = LU$$

  - reduction to triangular system
  - faster methods for special matrices (e.g. FFT)
- iterative methods:
  - fixed point (stationary) methods
  - optimisation based
- we can solve large systems of equations stably

example: polynomial systems with one variable

- dimension $d = 1$ i.e., $x \in \mathbb{R}$

$$p(x) = Cm(x)$$

  - $C \in \mathbb{R}^{n,k+1}$ coefficients
  - $m(x) = [x^k, x^{k-1}, \ldots, 1]^T$ monomial basis
- solution theory ($p \neq 0$):
  - solution is zero of gcd of $p_1, \ldots, p_n$
  - $s \leq k_g$ solutions if k_g is degree of gcd

example

$$p(x) = \begin{bmatrix} x^3 - 2x^2 + x - 2 \\ x^2 + 2x - 8 \end{bmatrix}$$

- monomial basis $m(x) = \begin{bmatrix} x^3 & x^2 & x & 1 \end{bmatrix}^T$
- matrix of the polynomial system

$$C = \begin{bmatrix} 1 & -2 & 1 & -2 \\ 0 & 1 & 2 & -8 \end{bmatrix}$$

- represent $p(x)$ using monomial basis

$$p(x) = \begin{bmatrix} 1 & -2 & 1 & -2 \\ 0 & 1 & 2 & -8 \end{bmatrix} m(x)$$

Euclid's algorithm

- Step 1: augmenting system with (redundant) equation

$$C_1 = \begin{bmatrix} 1 & -2 & 1 & -2 \\ 0 & 1 & 2 & -8 \\ 1 & 2 & -8 & 0 \end{bmatrix}$$

- Step 2: elimination

$$C_2 = \begin{bmatrix} 1 & -2 & 1 & -2 \\ 0 & 1 & 2 & -8 \\ 0 & 4 & -9 & 2 \end{bmatrix}$$

- Step 3: elimination

$$C_3 = \begin{bmatrix} 1 & -2 & 1 & -2 \\ 0 & 1 & 2 & -8 \\ 0 & 0 & -17 & 34 \end{bmatrix}$$

solution

- final system of equations

$$C_3 z = \begin{bmatrix} 1 & -2 & 1 & -2 \\ 0 & 1 & 2 & -8 \\ 0 & 0 & -17 & 34 \end{bmatrix} \begin{bmatrix} x^3 \\ x^2 \\ x \\ 1 \end{bmatrix} = 0$$

  - the last equation is $-17x + 34 = 0$
  - unique solution is $x = 2$
  - the other two equations for x^2 and x^3 redundant

general case

$$p(x) = Cm(x)$$

- array of monomials $m(x) = [x^\alpha]^T \in \mathbb{R}^N$
- coefficient matrix $C \in \mathbb{R}^{n,N}$
  - if $\alpha_i \le k_i$ for some integers k_i, then C is (matricized) tensor
- theory
  - uses *ideal* I_p generated by p_i

$$I_p = \left\{ \sum_{i=1}^{n} q_i(x) p_i(x) \mid q_i(x) \in \mathcal{P}_d \right\}$$

    where $\mathcal{P}_d$ is the set of polynomials of d variables
  - zeros of p are zeros of all elements of the ideal

2. elimination for polynomial systems

illustrative examples

- for linear systems (Gauss)

$$p(x,y) = \begin{bmatrix} 2x+2y-6 \\ x+3y+1 \end{bmatrix} \Rightarrow \begin{bmatrix} 2x+2y-6 \\ 2y+4 \end{bmatrix}$$

- for polynomials of one variable (Euclid)

$$p(x) = \begin{bmatrix} x^2-4 \\ x^3-8 \end{bmatrix} \Rightarrow \begin{bmatrix} x^2-2 \\ 4x-8 \end{bmatrix} \Rightarrow [x-2]$$

- for multivariate polynomials

$$p(x,y) = \begin{bmatrix} xy+3x \\ x^2-2 \end{bmatrix} \Rightarrow \begin{bmatrix} xy+3x \\ x^2-2 \\ 3x^2+2y \end{bmatrix} \Rightarrow \begin{bmatrix} xy+3x \\ x^2-2 \\ 3x^2+2y \\ 2y+6 \end{bmatrix}$$

S polynomial – the elimination step

- let

$$p(x) = ax^{\alpha} + r(x), \quad q(x) = bx^{\beta} + s(x)$$

  and

$$\gamma = \alpha \vee \beta, \quad \text{(elementwise minimum)}$$

- then

$$S(p,q) = bx^{\beta-\gamma}p - ax^{\alpha-\gamma}q = bx^{\beta-\gamma}r - ax^{\alpha-\gamma}s$$

  eliminates the first terms in q using p (symmetric)
- special case (reduction): $\gamma = \beta$

$$S(p,q) = bp - ax^{\alpha-\beta}q = br - ax^{\alpha-\beta}s$$

Groebner basis

- based on ordering of the (multiexponents of the) monomials (here lexicographic)
- A Groebner basis of an ideal I is a generating set of polynomials $p_1, \ldots, p_n$ of I such that for any

$$x^\alpha + s(x) \in I$$

there exists a p_i with

$$p_i(x) = ax^\alpha + r(x)$$

where $a \neq 0$

- Note that the leading terms x^α and ax^α depend on the ordering of the monomials
- the gcd of polynomials in one variable is a Groebner basis of the ideal generated by these polynomials

Buchberger algorithm

- Input: generating set $p_1, \ldots, p_n$ for ideal I
- Output: a Groebner basis for I
- Algorithm:
- initialise Groebner basis with input
- repeat
  - let $q = S(p_i, p_j)$ for some i, j
  - reduce q using the Groebner basis generated so far
  - if resulting polynomial is not zero, add to Groebner basis as p_{-1}
- until the reduced S-polynomials for all pairs p_i, p_j are zero

Theorem:

Let G be a Groebner basis for ideal I wrt lex order. Then $G_l = G \cap \mathcal{P}[x_{l+1}, \ldots, x_n]$ is a Groebner basis for the ideal $I_l = I \cap \mathcal{P}[x_{l+1}, \ldots, x_n]$

- here field is $\mathbb{R}$
- lex order is lexicographic order
- I_l is the elimination ideal

triangular polynomial systems

$$p(x) = \begin{cases} p_1(x_1, \dots, x_d) \\ p_2(x_2, \dots, x_d) \\ \vdots \\ p_d(x_d) \end{cases}$$

where

- $p_j : \mathbb{R}^{d-j+1} \to \mathbb{R}^{m_j}$
- components of p_j form a Groebner basis

why Groebner basis?

- example of a triangular system which is not Groebner basis

$$p(x, y) = \begin{bmatrix} y^4 + x - 1 \\ y \\ x^4 - 1 \end{bmatrix}$$

  - only solution: $x = 1, y = 0$

solving triangular polynomial system

1. use Euclid to possibly reduce p_d such that $m_d = 1$
2. solve $p_d(x_d) = 0$, possibly multiple solutions
3. substitute x_d in p_j by solution x_d^*
4. if $d > 0$ replace d by $d - 1$ and go back to start

3. examples of polynomial equations

general application areas

- fitting lower dimensional structures (varieties) to data points
- manifold learning
- low rank approximations of tensors

least squares

- minimise

$$\phi(x) = \|p(x)\|^2$$

- normal equations are polynomial system of equations

$$D(x)\,p(x) = 0$$

  - $D(x) = \frac{\partial p}{\partial x}(x)$ is the Jacobi matrix of $p(x)$

polynomial optimisation with polynomial constraints

- minimise polynomial objective

$$x = \text{argmin}_x \, \phi(x)$$

- polynomial constraints

$$p(x) = 0$$

- Lagrangian

$$L(x, \lambda) = \phi(x) + \lambda^T p(x)$$

- stationary points satisfy polynomial system in (x, λ)

$$\frac{\partial \phi}{\partial x}(x) + \lambda^T \frac{\partial p}{\partial x}(x) = 0$$

and

$$p(x) = 0$$

low rank interpolation

$$p(X, Y) = \left(XY^T - A\right) \circ M$$

- $X \in \mathbb{R}^{n \times k}$, $Y \in \mathbb{R}^{m \times k}$
- Hadamard product with mask matrix $M \in \{0, 1\}^{n \times m}$
- interpolatory factorisation of A if $Y^T = [I, Z]$
- most interpolation point choices seem not to work
- **question:** lattice points or similar which work?
- (penalised) least squares problems
- also for tensors, ALS

numerical (Gaussian) quadrature

$$p(x, w) = \left[\sum_{i=1}^{N} w_i f_j(x_i) - \int_\Omega f_j(x)\, dx\right]$$

- $f_1, \ldots, f_n$ polynomial basis functions for domain $\Omega \subset \mathbb{R}^s$
- domain Ω is cube, simplex or ball
- $d = 2N$
- todo: piecewise polynomials, quasi Monte Carlo methods
- also: Runge Kutta methods

4. computational challenges

challenges

- floating point algorithms based on Buchberger's algorithm are both slow and unstable compared to methods to solve linear system of equations
- numerical improvements by H. Stetter (Linz) and collaboraters in late 90s

zeros of quadratic polynomial – what can go wrong

- solving $x^2 + px + q = 0$
- solution

$$x_{1,2} = \frac{-p \pm \sqrt{p^2 - 4q}}{2}$$

  - if q small then the two solutions are approximately

$$x_1 \approx -p, \quad x_2 \approx -q/p$$

  - if q is extremely small applying the original formula gives

$$x_1 \approx -p, \quad x_2 \approx 0$$

  - solving the quadratic equation numerically challenging

a system of equations for solution pair – using Vieta's formulas

- some help from algebra: solve system of equations for both solutions
  - turns problem with multiple solutions into one with only one solution
  - use factorisation $x^2 + px + q = (x - x_1)(x - x_2)$ to get

$$x_1 + x_2 + p = 0$$
$$x_1 x_2 - q = 0$$

solving system of equations

- elimination of x_1 in the system gives $x_2^2 + px_2 + q = 0$
- an algebraic approach might produce reduced system

$$x_1 + x_2 + p = 0$$
$$x_2^2 + px_2 + q = 0$$

  - solve by back substitution – choosing the most accurate solution

$$x_2 = -\frac{p + \operatorname{sign} p \sqrt{p^2 - 4q}}{2}$$
$$x_1 = -x_2 - p$$

  - for our extreme problem one gets approximately $x_2 = -p, \quad x_1 = 0$

a more stable approach

- a numerically more stable approach choses reduced system to be

$$x_1 x_2 - q = 0$$
$$x_2^2 + p x_2 + q = 0$$

  - solve by back substitution – choosing the most accurate solution for x_2:

$$x_2 = -\frac{p + \operatorname{sign}(p)\sqrt{p^2 - 4q}}{2}$$
$$x_1 = q/x_2$$

  - for our extreme problem one gets approximately $x_2 = -p, \quad x_1 = -q/p$

Vieta's formulas for higher degree polynomials or the system of polynomial equations for all zeros of one polynomial

- we want to compute *all* zeros $x_1, \ldots, x_n$ of the polynomial

$$p(x) = x^n + a_1 x^{n-1} + \cdots + a_{n-1} x + a_n$$

- now let $\sigma_k = \sigma_k(x_1, \ldots x_n)$ be defined by

$$\prod_{k=1}^{n} (x - x_k) = x^n + \sigma_1 x^{n-1} + \cdots \sigma_{n-1} x + \sigma_n$$

  - σ_k is a symmetric, homogenous polynomial of degree k

- then the x_k satisfy the system of polynomial equations

$$\sigma_k(x_1, \ldots, x_n) = a_k, \quad \text{for } k = 1, \ldots, n$$

- we compute the zeros of the polynomial by solving this system of polynomial equations (we need some algebraic geometry ...)

Wilkinson's (perfidious) polynomial

(from Wikipedia: 'Wilkinson's polynomial, original: Wilkinson, 1959, in Numerische Mathematik)

$$p(x) = \prod_{k=0}^{20} (x - k)$$

- Wilkinson showed that 30 bit arithmetic was not sufficient to determine the roots of this polynomial from the coefficients wrt to the monomial basis
- also, some coefficients are extremely large
- need to use different representation of polynomial
  - orthogonal polynomials, Lagrangian basis, ...
  - recursions

polynomials defined by a recursion

$$p_n(x) = (x - \alpha_n)p_{n-1}(x) - \beta_n^2 p_{n-2}(x)$$

- happens when polynomial generated by orthogonalisation (Gram-Schmidt), common in numerical analysis, connections to Lanczos, conjugate gradient method
- algebraic approach:
    - interprete polynomial as characteristic polynomial of a matrix
    - the zeros correspond to rank deficient matrices

LU factorisation of a tridiagonal matrix

$$\begin{bmatrix} a_1 & b_1 & & \\ b_1 & a_2 & \ddots & \\ & \ddots & \ddots & b_{n-1} \\ & & b_{n-1} & a_n \end{bmatrix} = \begin{bmatrix} 1 & & & \\ l_1 & 1 & & \\ & \ddots & \ddots & \\ & & l_{n-1} & 1 \end{bmatrix} \begin{bmatrix} g_1 & b_1 & & \\ & g_2 & \ddots & \\ & & \ddots & b_{n-1} \\ & & & g_n \end{bmatrix}$$

- where
  - here $l_k = b_k / g_k$ are elimination coefficients
  - and $g_k = a_k - l_{k-1} b_{k-1}$ are Schur complements
- and determinant p_n of this matrix is

$$p_n = g_1 g_2 \cdots g_n$$

Recursion

- recursion for determinant

$$p_n = g_n t_{n-1} = (a_n - b_{n-1}^2/g_{n-1})p_{n-1} = a_n t_{n-1} - b_{n-1}^2 p_{n-2}$$

- substitute $a_k = x - \alpha_k$ and $b_k = \beta_k$ for all k then

$$p_n = (x - \alpha_n)p_{n-1} - \beta_{k-1}^2 p_{n-2}$$

- thus p_n is a determinant

$$p_n(x) = (-1)^n \det \begin{bmatrix} \alpha_1 - x & -\beta_1 & & \\ -\beta_1 & \alpha_2 - x & \ddots & \\ & \ddots & \ddots & -\beta_{n-1} \\ & & -\beta_{n-1} & \alpha_n - x \end{bmatrix}$$

- use (stable) numerical eigenvalue solver to compute zero(s) of polynomial (Golub and Walsh, 1968) ...

Conclusions

- algebraic sets (defined by polynomials) occur frequently in applications
- there are some direct solvers for polynomial systems of equations which use elimination
- however, at this stage numerical solvers have difficulties to solve systems with more than a few unknowns as current methods are
  - computationally expensive
  - numerically unstable