Numerics, and polynomial equations

Markus Hegland, ANU Anand Deopurkar, ANU Edda, Feng and Apphia, ANU summer scholars

1. Introduction

Overview

- aim: develop new numerical approaches of solving systems of polynomial equations
- there is more to solving nonlinear systems of equations than Newton-Raphson
- we consider two particular examples:
  - design of Gaussian quadrature methods
    - quadrature using in finite element methods for PDEs
  - computing low rank tensor factorisations from data
    - used in solving PDEs, reduced basis methods, fast (online) computations
- questions: complexity (speed and storage space) and accuracy

zeros of quadratic polynomial

- 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:

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

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

Example of system of polynomial equations

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 ...)

2. Solving polynomial systems of equations in numerical analysis

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) ...

Example 1: Gaussian quadrature

Quadrature methods

numerical approximation of integral

$$I(f) = \int_{-1}^{1} \omega(x) f(x)\, dx$$

by a quadrature method

$$Q(f) = \sum_{i=1}^{n} w_i f(x_i)$$

- often: $\omega(x) = 1$
- Gauss' approach: choose w_i and x_i such that $I(f) = Q(f)$ for all f polynomials of degree $2n - 1$
- Newton-Cotes: choose $x_i = -1 + (i-1)h$ for $i = 1, \ldots, n$ and $h = 2/(n-1)$ and w_i such that $I(f) = Q(f)$ for all polynomials f of degree $n - 1$
- quasi Monte Carlo: choose all $w_i = \frac{1}{n} \int_{-1}^{1} \omega(x)\, dx$ and choose x_i such they are well spaced out (each ball of fixed size contains same number of points)

an example of a Gauss quadrature which is quasi Monte Carlo

- Chebyshev-Gauss quadrature: approximate

$$I(f) = \int_{-1}^{1} \frac{f(x)}{\sqrt{1-x^2}}\, dx$$

by

$$Q(f) = \frac{\pi}{n} \sum_{i=1}^{n} f(x_i)$$

where $x_i = \cos\left(\frac{2i-1}{2n}\pi\right)$

- orthogonal polynomials: Chebyshev polynomials
- quadrature points explicitely known

system of polynomial equations

- for the $2n$ unknowns x_i and w_i one solves the $2n$ equations obtained from

$$Q(x^k) = I(x^k), \quad k = 0, \ldots, 2n-1$$

  - i.e. compute points and weights from moments
- explicitly

$$w_1 x_1^k + \cdots + w_n x_n^k - a_k = 0, \quad k = 0, 1, \ldots 2n-1$$

  - where $a_{2k+1} = 0$ and $a_{2k} = 2/(2k+1)$
- we consider the case $\omega(x) = 1$

Gaussian quadrature polynomial equations

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ x_1^2 & x_2^2 & \cdots & x_n^2 \\ \vdots & \vdots & & \vdots \\ x_1^{2n-1} & x_2^{2n-1} & \cdots & x_n^{2n-1} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{2n-1} \end{bmatrix}$$

- here where $a_{2k+1} = 0$ and $a_{2k} = 2/(2k+1)$

Gaussian quadrature in block form and equivalent form

- block structure of augmented matrix

$$\begin{bmatrix} V & -a \\ W & -b \end{bmatrix} \begin{bmatrix} w \\ 1 \end{bmatrix} = 0$$

- V Vandermonde matrix, is invertible if all x_i are distinct
- elminating w / Schur complement

$$\begin{bmatrix} V & -a \\ 0 & -b - WV^{-1}a \end{bmatrix} \begin{bmatrix} w \\ 1 \end{bmatrix} = 0$$

- one can show that $V^{-1}W$ is a polynomial (in the x_i) matrix

Quasi Monte Carlo

- polynomial system written with Vandermonde matrix

$$Ve - a = 0$$

  - here $e = (1, \ldots, 1)^T$ and a is not the same vector as for the Gauss method

- use polynomial elimination to solve the system
  - idea: something $V = LU$ then U leads to a system where the k-th row only contains $x_{k+1}, \ldots, x_{n-1}$
- note that the the Quasi Monte Carlo scheme does not have the same accuracy except for the case of $\omega(x) = 1/\sqrt{1-x^2}$ which is a Gauss scheme with polynomials being the Chebyshev polynomials
- the polynomial system has $n+1$ unknowns where the weight $w \in \mathbb{R}$ is computed explicitely from the first equation as $w = a_0/n$

Gauss-Kronrod

- in this scheme, one is given some of the quadrature points $t_1, \ldots, t_k$ and looks for the others $x_1, \ldots, x_{n-k}$ and the corresponding weights $w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$ (where w_i are vectors)
- the equations are similar to the ones for Monte Carlo and Gauss, i.e., of the form

$$V(t_1, \ldots, t_k)w_1 + V(x_1, \ldots, x_{n-k})w_2 - a = 0$$

  - where $V(\cdots)$ are rectangular Vandermond matrices for the given points
- a special case is the Patterson scheme where two points at the boundary are fixed but the others are chosen, i.e., $k = 2$
- note that these schemes do have reduced accuracy compared with the Gauss scheme
- the polynomial system has $2n - k$ unknowns

Legendre polynomials

- $p_0, p_1, \ldots$ orthogonal polynomials wrt scalar product $(p, q) = I(pq)$ and degree p_k is k
- satisfy a 3-term recursion

Theorem Gauss quadrature points x_i are zeros of p_n

Proof.

- let $p_n(x_i) = 0$ for $i = 1, \ldots, n$
- choose w_i such that $Q(p) = I(p)$ for p with degree less than n
- for every polynomial p wit degree less than $2n$ there exist two polynomials r, q with degree less than n such that

$$p = qp_n + r$$

- as q is orthogonal to p_n one has

$$I(p) = I(qp_n) + I(r) = I(r)$$

- as p_n is zero on x_i one has

$$Q(p) = Q(qp_n) + Q(r) = Q(r)$$

Example 2: Low rank approximation of tensors

low rank approximation of matrix A

- singular value decomposition

$$A = U\Sigma V^T$$

  - best low rank approximation (in Frobenius/L2) of rank $k \leq k_A$:

$$A_k = U_k \Sigma_k V_k^T$$

- rank revealing QR factorisation

$$A = QR\Pi$$

  - rank k approximation

$$A_k = Q_k R_k \Pi$$

    projection onto space spanned by first k columns of $A\Pi$

- interpolative factorisation based on LU decomposition

$$A = \Pi L U \Pi'$$

  - rank k approximation

$$A_k \approx \Pi L_k U_k \Pi'$$

3. Algebraic methods of solving polynomial systems – elimination theory

Algebra and numerical analysis

- given polynomials $p_i(x) = 0, i = 1, \ldots, s$ find $x \in \mathbb{R}^d$
- simple numerical approach: solve system with Newton-Raphson method often challenging
- observation: $p(x) = 0$ for all $p \in I = \text{ideal}(p_1, \ldots, p_s)$
- method: choose $p_i \in I$, $i = 1, \ldots, n$ for $n > s$
- linear space of polynomials $\text{span}(p_i) \subset V$ with monomial basis x^α and $z_\alpha = x^\alpha$ then matrix C with $p_i(x) = \sum_\alpha C_{i,\alpha} x^\alpha$ and

$$Cz = 0$$

- solution – combination of algebra and numerical analysis
  - determine linear space V (algebra) and basis x^α in lexicographic ordering
  - simplify $Cz = 0$ to triangular form (elimination)
  - solve $x^\alpha = z_\alpha$ for $z \in \ker(C)$ (back substitution and nonlinear solvers)

Example: one variable

Solving a system of equations with one variable

$$x^3 - 2x^2 + x - 2 = 0$$
$$x^2 + 2x - 8 = 0$$

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

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

- system of equations

$$Cz = \begin{bmatrix} 1 & -2 & 1 & -2 \\ 0 & 1 & 2 & -8 \end{bmatrix} \begin{bmatrix} x^3 \\ x^2 \\ x \\ 1 \end{bmatrix} = 0$$

augmenting and elimination (Euclid)

- 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

Example: 2 variables

Solving a system of equations with 2 variables

$$x_1 x_2 + 8 = 0$$
$$x_1 + x_2 + 2 = 0$$

- monomial basis $z = \begin{bmatrix} x_1 x_2 & x_1 & x_2 & 1 \end{bmatrix}$
- matrix of polynomial system

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

- polynomial system in matrix form

$$Cz = \begin{bmatrix} 1 & 0 & 0 & 8 \\ 0 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 x_2 \\ x_1 \\ x_2 \\ 1 \end{bmatrix} = 0$$

augmenting and elimination

- Step 1: augmenting, new basis: $z_1^T = \begin{bmatrix} x_1 x_2 & x_1 & x_2^2 & x_2 & 1 \end{bmatrix}$

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

- Step 2: elimination (last row with first row)

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

reduced system of equations and solution

$$C_2 z_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 8 \\ 0 & 1 & 0 & 1 & 2 \\ 0 & 0 & 1 & 2 & -8 \end{bmatrix} \begin{bmatrix} x_1 x_2 \\ x_1 \\ x_2^2 \\ x_2 \\ 1 \end{bmatrix} = 0$$

- Step 3: solve last equation to get (one) solution $x_2 = 2$
- Step 4: use factorisation of z_1 and substitute x_2

$$\begin{bmatrix} x_2 & 0 \\ 1 & 0 \\ 0 & x_2^2 \\ 0 & x_2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 1 & 0 \\ 0 & 4 \\ 0 & 2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ 1 \end{bmatrix}$$

substitution and reduction

- Step 5: new variable $z_2^T = \begin{bmatrix} x_1 & 1 \end{bmatrix}$ then satisfies

$$C_3 z_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 8 \\ 0 & 1 & 0 & 1 & 2 \\ 0 & 0 & 1 & 2 & -8 \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 1 & 0 \\ 0 & 4 \\ 0 & 2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ 1 \end{bmatrix} = 0$$

and from this one gets

$$C_3 z_2 = \begin{bmatrix} 2 & 8 \\ 1 & 4 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ 1 \end{bmatrix} = 0$$

- it follows that $x_1 = -4$ (equations redundant but not in terms of rounding error)

Example: 3 variables (rank 1 interpolation)

rank 1 matrix interpolation

$$
\begin{aligned}
xy_1 - 1 &= 0 \\
xy_2 - 5 &= 0 \\
y_1 - 1 &= 0
\end{aligned}
$$

- interpolate the values a_{11}, a_{21} and a_{22} to get a_{12} where

$$A = \begin{bmatrix} 1 \\ x_1 \end{bmatrix} \begin{bmatrix} y_1 & y_2 \end{bmatrix}$$

- monomial basis $z^T = \begin{bmatrix} xy_1 & xy_2 & x & y_1 & y_2 & 1 \end{bmatrix}$
- system matrix

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 & 0 & -5 \\ 0 & 0 & 0 & 1 & 0 & -1 \end{bmatrix}$$

- system of equations $Cz = 0$

augmented form

- product $z = \begin{bmatrix} y_1 \\ 1 \end{bmatrix} \otimes \begin{bmatrix} x \\ 1 \end{bmatrix} \otimes \begin{bmatrix} y_2 \\ 1 \end{bmatrix}$

$$z = \begin{bmatrix} y_1 x y_2 & y_1 x & y_1 y_2 & y_1 & x y_2 & x & y_2 & 1 \end{bmatrix}^T$$

- matrix C

$$C = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & -5 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 \end{bmatrix}$$

factorisation of z and substitution solution

- factorisation of z

$$z = \begin{bmatrix} y_1 I_4 \\ I_4 \end{bmatrix} \begin{bmatrix} x I_2 \\ I_2 \end{bmatrix} \begin{bmatrix} y_2 \\ 1 \end{bmatrix}$$

- polynomial system of equations

$$C \begin{bmatrix} y_1 I_4 \\ I_4 \end{bmatrix} \begin{bmatrix} x I_2 \\ I_2 \end{bmatrix} \begin{bmatrix} y_2 \\ 1 \end{bmatrix} = 0$$

- check C_1 to see that $y_1 = 1$:

$$C_1 = C \begin{bmatrix} y_1 I_4 \\ I_4 \end{bmatrix} = \begin{bmatrix} 0 & y_1 & 0 & -1 \\ 1 & 0 & 0 & -5 \\ 0 & 0 & 0 & y_1 - 1 \end{bmatrix}, \quad C_1 z_1 = C_1 \begin{bmatrix} x I_2 \\ I_2 \end{bmatrix} \begin{bmatrix} y_2 \\ 1 \end{bmatrix} = 0$$

continued . . .

- we then get

$$C_1 = \begin{bmatrix} 0 & 1 & 0 & -1 \\ 1 & 0 & 0 & -5 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

- check C_2 below to see that $x = 1$

$$C_2 = C_1 \begin{bmatrix} xI_2 \\ I_2 \end{bmatrix} = \begin{bmatrix} 0 & x-1 \\ 1 & -5x \\ 0 & 0 \end{bmatrix}, \quad C_2 z_2 = C_2 \begin{bmatrix} y_2 \\ 1 \end{bmatrix} = 0$$

- from which one has

$$C_2 = \begin{bmatrix} 0 & 0 \\ 1 & -5 \\ 0 & 0 \end{bmatrix}$$

and it follows from the equation above that $y_2 = -5$

generalisation

- this method can be used generally to solve a system of polynomial equations for which one of the polynomials has degree one
- need this to hold for all C_k
- question for which data does this structure occur?
  - only for ACA type structure
  - more generally??
- conjecture: if data points can be connected with (rectangular) spanning tree, this solution method works for rank 1 interpolation
- challenge: growth of matrix exponentially with number of unknowns but the matrix is very sparse

Example: 4 variables (Gaussian quadrature points)

the system of nonlinear equations

- compute weights w_i and quadrature points x_i such that

$$
\begin{aligned}
w_1 + w_2 &= 2 \\
w_1 x_1 + w_2 x_2 &= 0 \\
w_1 x_1^2 + w_2 x_2^2 &= 2/3 \\
w_1 x_1^3 + w_2 x_2^3 &= 0
\end{aligned}
$$

the equations in matrix form

- equations $Cz = CB(x) \begin{bmatrix} w \\ 1 \end{bmatrix} = 0$

$$\begin{bmatrix} 1 & & & 1 & & & \\ & 1 & & & 1 & & 2/3 \\ & & 1 & & & 1 & \\ & & & 1 & & & 1 \quad 2 \end{bmatrix} \begin{bmatrix} x_1^3 & & \\ \vdots & & \\ 1 & & \\ & x_2^3 & \\ & \vdots & \\ & 1 & \\ & & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ 1 \end{bmatrix}$$

- usually, equations written as $V(x) \begin{bmatrix} w^T & 1 \end{bmatrix}^T = 0$ where

$$V(x) = CB(x) = \begin{bmatrix} x_1^3 & x_2^3 & 0 \\ x_1^2 & x_2^2 & 2/3 \\ x_1 & x_2 & 0 \\ 1 & 1 & 2 \end{bmatrix}$$

at the end of the elimination algorithm

- during the elimination procedure, the matrix C is brought to (block-) triangular shape but the equations still have the form

$$C_e B_e(x) \begin{bmatrix} w \\ 1 \end{bmatrix} = 0$$

  - the matrix $B_e(x)$ is extended and may have many new columns but still has a block structure relating to the w_i

- finally, the values of x_i and w_i can then be determined using a substitution process similar to the one from the previous section which includes updating the matrix C_e as components emerge

tools from algebraic geometry

division algorithm and Groebner basis

- division algorithm
  - given $p_1, \ldots, p_n$ determine remainder of any p wrt $p_1, \ldots, p_n$
- Groebner bases and Buchberger algorithm
  - set of polynomials p_i such that for any $p \in \text{ideal}(p_1, \ldots)$ one has remainder wrt $p_1, \ldots, p_n = \text{zero}$
  - key property: the Groebner basis contains for any polynomial p in the ideal an element p_k such that the leading (monomial) term of p_k divides the leading term of p
  - Buchberger algorithm uses elimination approach to compute Groebner basis from generating system of an ideal

elimination theorem

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

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

extension theorem

- a zero $(x_{l+1}, \ldots, x_1)$ of I_l is called a *partial solution*

Theorem.
If the maximal order coefficients $c_q(x_{l+1}, \ldots, x_n)$ are nonzero for some of the Groebner basis

$$f = c_q(x_{l+1}, \ldots, x_n)x_l^q + \cdots + c_0(x_{l+1}, \ldots, x_n)$$

then partial solution extends to $(x_l, \ldots, x_n)$.