Special topics course Numerical Polynomial Algebra

Anand Deopurkar and Markus Hegland

ANU winter semester 2018

Chapter 1: Numerical introduction

introductory section – numerical aspects

- here we consider polynomial systems of equations for the real and complex fields

- polynomial systems of equations are a generalisation of linear systems of equations which include powers x_i^k and products of powers of the unknowns
- applications of polynomial systems of equations:
  - computing Gaussian quadrature points
  - computing coefficients of Runge-Kutta methods
  - eigenvalue problems
  - rank-k matrix approximation
  - tensor approximations and decompositions
  - polynomial optimisation
  - semidefinite programming
  - machine learning
  - robotics
  - cryptography
  - …
- for some cases there are stable and well-studied methods using specialised algorithms
- for some cases very large systems can be solved using general (Buchberger-based) algorithms
- in some cases even small system are practically unsolvable
- algorithms in the rational field and finite fields are often effective
- algorithms for the real and complex fields based on floating point numbers are often unstable
- simple example is Euclid's algorithm which is related to Gaussian elimination (without pivoting)
  - problems to solve are Toeplitz systems
  - some numerical problems can be illustrated there
  - algorithms based on stable (QR) algorithms have been investigated

1.1 Linear spaces of polynomials

This section is based on section 1.1 in Stetter's book. Here we define the linear space of multivariate polynomials. We first define the monomial basis. A univariate monomial is a power x^j for $x \in \mathbb{R}$ and $j \in \mathbb{N}_0$ where $\mathbb{N}_0$ is the set of non-negative integers. A multivariate monomial is a product of the powers of the variables, i.e.,

$$x^j = x_1^{j_1} \cdots x_s^{j_s}$$

where $x \in \mathbb{R}^s$ and $j \in \mathbb{N}_0^s$. The *set of (s-variate) monomials* $\mathcal{T}^s$ is

$$\mathcal{T}^s = \{x^j \mid x \in \mathbb{R}^s,\ j \in \mathbb{N}_0^s\}.$$

The set of monomials is *closed under multiplication*, i.e., if $x^j, x^k \in \mathcal{T}^s$ then $x^j * x^k = x^{j+k} \in \mathcal{T}^s$. As the multiplication is associative we have

Proposition: The set $(\mathcal{T}^s, *)$ of monomials is a symmetric semigroup with unit 1.

proof: use map $x^j \to j$

Definition: A *semigroup* $(S, *)$ is a set S with a binary operation $*$ satisfying the associative law

$$a * (b * c) = (a * b) * c$$

Examples:

- non-negative integers $(\mathbb{N}_0, +)$

$$\mathbb{N}_0 = \{0, 1, 2, \ldots\}$$

  - is symmetric and contains unit 0
  - does not contain negative integers and thus is not a group
- vectors of non-negative integers $(\mathbb{N}_0^s, +)$

Proposition: The semigroup $(\mathcal{T}^s, *)$ is isomorph to $(\mathbb{N}_0^s, +)$.

Definition: The set of s-variate polynomials is

$$\mathcal{P}^s = \{p(x) = \sum_{j \in J} c_j x^j \mid J \subset \mathbb{N}_0^s, |J| < \infty, c_j \in \mathbb{R}\}.$$

Here $|J|$ denotes the cardinality (size) of the set J. A shorthand notation for this is

$$\mathcal{P}^s = \mathbb{R}(\mathcal{T}^s).$$

From the definition it follows that $\mathcal{P}^s$ is a *commutative algebra*, i.e.,

- a real vector space with component-wise addition and multiplication by reals
- a ring with the multiplication defined by

$$p(x)q(x) = \sum_{j,k} c_j d_k x^{j+k}$$

  where $p(x) = \sum_j c_j x^j$ and $q(x) = \sum_k d_k x^k$.

In practice, the polynomials occurring are sparse, i.e., the size $|J|$ in the sums are small.

In computations, the degree of the polynomials will be limited. One uses, for example

Definition: (monomials with finite degree)

$$\mathcal{T}_d^s = \{x^j \in \mathcal{T}^s \mid |j| < d\}$$

where degree of x^j is $|j| = j_1 + \cdots + j_s$

- $\mathcal{T}_d^s$ is not a semigroup

Generating a vector space from this set gives polynomials of degree d

$$\mathcal{P}_d^s = \mathbb{R}(\mathcal{T}_d{}^s).$$

The size $\mathcal{T}_d{}^s = \binom{d+s}{s}$ is equal to the dimension of the vector space $\mathcal{P}_d^s$.

Examples:

- $\mathcal{T}_d{}^1 = \{1, x, x^2, \ldots, x^d\}$ which generates

  $$\mathcal{P}_d^1 = \mathbb{R}(\mathcal{T}_d{}^1),$$

  the univariate polynomials of degree up to d and
- $\mathcal{T}_1{}^s = \{1, x_1, x_2, \ldots, x_s\}$ which generates

  $$\mathcal{P}_1^s = \mathbb{R}(\mathcal{T}_1{}^s),$$

  the multivariate polynomials of degree up to one.

In some applications one uses the tensor product of univariate polynomials of a fixed degree and has

$$\mathcal{T}_d{}^{\otimes s} = \{x^j \in \mathcal{T}^s \mid j_i = 0, \ldots, d,\ i = 1, 2, \ldots, s\}$$

which generates

$$\mathcal{P}_d^{\otimes s} = \mathbb{R}(\mathcal{T}_d{}^{\otimes s}) = \mathcal{P}_d^1 \otimes \cdots \otimes \mathcal{P}_d^1.$$

For a given finite subset of $\mathcal{T}^s$ one defines the vector of basis functions for the corresponding polynomial vector space as array $\underline{\mathbf{x}} = (x^j)_{j \in J}$. A polynomial in the space $\mathcal{P}_J$ generated by x^j for $j \in J$ then takes the form

$$p(x) = c^T \underline{\mathbf{x}}$$

where $c \in \mathbb{R}^{|J|}$.

For example, if $J = \{0, \ldots, d\}$ then

$$\underline{\mathbf{x}} = (x^d, x^{d-1}, \ldots, 1)^T$$

(we will order highest degree first). Then $c^T \underline{\mathbf{x}}$ generates all the elements of $\mathcal{P}_d^1$, i.e.

$$\mathcal{P}_d^1 = \{c^T \underline{\mathbf{x}} \mid c \in \mathbb{R}^{d+1}\}.$$

If $J = \{e_1, e_2, \ldots, e_s, 0\}$ where e_i is the i-th standard basis vector and 0 the zero vector one has

$$\underline{\mathbf{x}} = (x, 1)^T = (x_1, x_2, \ldots, x_s, 1)^T$$

which generates the elements of $\mathcal{P}_1^s$, i.e.,

$$\mathcal{P}_1^s = \{c^T \underline{\mathbf{x}} \mid c \in \mathbb{R}^{s+1}\}$$

in this case.

Definition: A polynomial system $P(x)$ is an array of polynomials $p_i(x) \in \mathcal{P}^s$ such that

$$P(x) = \begin{bmatrix} p_1(x) \\ \vdots \\ p_n(x) \end{bmatrix}.$$

A polynomial system (or system of polynomials) can be expressed as the matrix vector product

$$P(x) = C\underline{\mathbf{x}}$$

for some matrix $C \in \mathbb{R}^{n \times |J|}$.

1.2 Solutions of polynomial systems of equations

This section includes basic material plus topics from Section 1.2 of Stetter's book.

Definition: A real (complex) solution or zero of a polynomial system is a vector $x \in \mathbb{R}^s$ ($x \in \mathbb{C}^s$) which satisfies

$$P(x) = 0.$$

From a mathematical perspective, we are interested in two questions:

- Does a particular polynomial system have a solution?
- What is the nature of the set of all solutions of a polynomial system? (Eg, is it a smooth curve or manifold, a point or a line or vector space?)

Solutions of univariate polynomial systems

We remember first that zero degree polynomials have no zeros (unless the are the zero polynomial which has value identical zero). In the following we implicitly assume that any polynomial we are talking about is not the zero polynomial.

Then we also know that any polynomial of degree one has exactly one zero. This is valid for any field, including $\mathbb{R}$ and $\mathbb{C}$.

The following is an application of a theorem for continuous functions as polynomials are continuous.

Proposition: Any real, univariate polynomial which has both positive and negative function values has at least one real zero.

A direct consequence is

Corollary: Any real univariate polynomial of odd degree has at least one real zero.

However, not all real univariate polynomials have real zeros, a simple example is $p(x) = x^2 + 1$. A fundamental result of algebra, however, is the following.

Theorem: Any complex polynomial of degree larger than zero has at least one (complex) zero.

Note, however, that a system with $n > 1$ univariate polynomial equations may not have a solution. For example, the system

$$x^2 - 1 = 0$$
$$x^3 - 1 = 0$$

has the solution $x = 1$ while the system

$$x^2 - 4 = 0$$
$$x^3 - 1 = 0$$

does not have a solution.

The nature of the set of solutions is provided now:

Proposition: The set of all solutions of a system of real or complex univariate polynomials is finite and may have at most d elements where d is the degree of the system.

Summarising, for the case $s = 1$ one has a good understanding of the set of solutions using results from algebra and calculus.

Solutions of linear systems of equations

In this case of first degree polynomials in $\mathcal{P}_1^s$ and the polynomial system is given by

$$P(x) = \begin{bmatrix} A & -b \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix} = Ax - b.$$

The matrix is $A \in \mathbb{R}^{n,s}$. Now let

$$\text{range}(A) = \{Ax \mid x \in \mathbb{R}^s\}$$

be the *range* of the matrix A. By definition one has

A linear system $Ax - b = 0$ *has a solution* x *if*

$$b \in \text{range}(A).$$

A general $b \in \mathbb{R}^n$ can be decomposed as

$$b = b_1 + b_2$$

where $b_1 \in \mathrm{range}(A)$ and $b_2 \in \mathrm{range}(A)^\perp$, i.e., b_2 is orthogonal to the range. It follows that we have a solution only if $b_2 = 0$.

Now for any $y \in \mathrm{range}(A)^\perp$ one has

$$(Ax)^T y = 0$$

for all $x \in \mathbb{R}^s$ and thus y has to satisfy the homogeneous system of equations

$$A^T y = 0.$$

The set of all such y is called the *null-space* of A

$$\mathrm{null}(A^T) = \{y \mid A^T y = 0\}$$

and thus

$$\mathrm{range}(A)^\perp = \mathrm{null}(A^T).$$

If the nullspace of A is nontrivial (not equal $\{0\}$) then the determinant of A is zero

$$\det(A) = 0.$$

Note that the determinant is a polynomial in the matrix elements and thus the nullspace is a solution of a particular polynomial equation.

If $b \in \mathrm{range}(A)$ it has to be orthogonal to $\mathrm{range}(A)^\perp$ and one gets the major result of linear algebra:

Proposition: The linear system $Ax = b$ *has a solution* if b is orthogonal to any vector $y \in \mathrm{null}(A^T)$.

The null space $\mathrm{null}(A)$ can be shown to be a linear space with dimension $s - r$ where r is the rank of the matrix A (which basically is defined by the dimension of the null space). One can see that if x is a solution of $Ax = b$ and $y \in \mathrm{null}(A)$ then $x + y$ is also a solution as $A(x + y) = b$. One then has a characterisation of all the solutions of $Ax = b$:

Proposition: The set of solutions of $Ax = b$ is an *affine set* given by

$$\{x + y \mid Ay = 0\}$$

for any solution x of $Ax = b$.

The original characterisation $Ax + b = 0$ of the solutions and the proposition above give an *implicit* description of the set of solutions. As the null space of A is a linear or vector space, there exists a

matrix $Y \in \mathbb{R}^{s,m}$ such that $AY = 0$ and where the columns of Y are just the basis of the null space of A. It then follows that the set of solutions of $Ax - b = 0$ can be represented as

$$\{x + Yt \mid t \in \mathbb{R}^m\}.$$

Note that m is the dimension of the null space of A. We call this representation *explicit*.

As in the case of univariate polynomials one also has a good idea of what the solutions are for linear systems of equations. The situation is not as simple for higher degree multivariate polynomials.

Inverse function theorem

The next theorem reduces the theory of nonlinear problems to linear problems for the case $s = n$. Here we consider continuously differentiable functions F. We will use the *Jacobi matrix* which we denote by

$$J_F(x) = F'(x) = \begin{bmatrix} \partial F_1(x)/\partial x_1 & \cdots & \partial F_1(x)/\partial x_s \\ \vdots & & \vdots \\ \partial F_n(x)/\partial x_1 & \cdots & \partial F_n(x)/\partial x_s \end{bmatrix}.$$

Inverse Function Theorem: *Let $F : \mathbb{R}^s \to \mathbb{R}^s$ be C^1 in a neighborhood of some point $x_0 \in \mathbb{R}^s$. If the null space*

$$\operatorname{null}(F'(x_0)) = 0$$

then F is invertible in a neighborhood of x_0.

This means that the equation

$$F(x) = F(x_0) + y$$

has a unique solution x for all y in a neigborhood of $F(x_0)$.

Solutions of polynomial systems of equations

This is related to the section 1.2 of Stetter's book.

We will denote the Jacobian of a polynomial by $P'(x) = J_P(x)$.

More generally, we introduce differential operators related to polynomials. Let the gradient be

$$\partial = \nabla = (\partial_1, \ldots, \partial_s).$$

(A row vector.) Then let

$$\partial_j = \nabla^j = \frac{1}{j!} \frac{\partial^{|j|}}{\partial x^j}$$

where

- $j! = j_1! \cdots j_s!$
- $|j| = j_1 + \cdots + j_s$
- $\partial x^j = \partial x_1^{j_1} \cdots \partial x_s^{j_s}$

Furthermore, let $q(x) = \sum_j b_j x_j$ then we define

$$q(\partial) = \sum_j b_j \partial_j .$$

Then one has

Proposition: (Leibniz rule)

$$\partial_j (p \cdot q) = \sum_{k \leq j} \partial_{j-k} p \, \partial_k q$$

Proposition: (derivative of polynomials)

If $p \in \mathcal{P}_d^s$ then

- $\partial_j p \in \mathcal{P}_{d-|j|}^s$ *for* $|j| \leq d$
- $\partial_j p = 0$ *for* $|j| > d$

Proposition: (Taylor)

$$p(x+h) = \sum_{k=0}^{d} \sum_{|j|=k} \partial_j p(x) \, h^j =: p(h; x)$$

Proposition: (zero polynomial) *A polynomial $p(x)$ defines the zero mapping if and only if it is the zero polynomial (i.e. has only zero coefficients).*

proof: induction over s:

- show for $s = 1$
- $p(x) = p(x_1, \ldots, x_{s-1}, x_s)$ is univariate in x_s ...

Proposition: (existence in $\mathbb{C}^s$) *Any polynomial $p \in \mathcal{P}^s$ has at least one zero in $\mathbb{C}^s$ if $s > 0$.*

proof: induction as before

Definition: A polynomial system $P(x)$ is *essentially linear* if $\det(P'(x))$ is constant.

Proposition: *If $n = s$, a polynomial system $P(x)$ has a non-invertible Jacobian at least in one point $x \in \mathbb{C}^s$ unless $P(x)$ is essentially linear.*

proof: induction as before and note that determinant of Jacobian is a polynomial

1.3 Floating point numbers and computations

This section is related to chapter 4 of the book by Stetter.

When solving polynomial systems in the real and complex fields one requires approximations. There are basically two possibilities which can be illustrated

1. Approximation in the rational field $\mathbb{Q}$
2. Approximation in the set $\mathcal{F}(D)$ of floating point numbers

The first approach has the advantage that all the computations can be done in $\mathbb{Q}$. However, the denominators may become very large when the computations are extensive and one thus requires to monitor the size of the denominators and approximate when required. Here we will choose computations with floating point numbers which is the default choice for real computations in computational science. A disadvantage is that essentially all computations performed in floating point arithmetic are done approximately. However, this is done automatically and only the effects are visible to the user.

We will use base 2 for our floating point numbers and consider a slightly idealised infinite set for our theoretical discussions, basically neglecting the effects of over- and under-flow. The floating point numbers depend on the number digits D used to compute which has a big effect on the accuracy. The **set of floating point numbers** is

$$\mathcal{F}(D) = \{\pm m\, 2^E \mid m = n\, 2^{-D},\ n = 2^{D-1}, 2^{D-1}+1, \ldots, 2^D - 1,\ E \in \mathbb{Z}\} \cup \{0\}.$$

The rational number m is called the mantissa and the integer E the exponent. In practice the range of E is also bounded (which gives rise to over- and under-flow errors). The representation with two integers n and E characterises each nonzero floating point number uniquely. Note that one does require to represent the number zero separately.

The set of floating point numbers $\mathcal{F}(D) \subset \mathbb{Q}$ contains binary fractions which are distributed in a *semi-logarithmic* fashion:

Proposition: Every interval $[2^{E-1}, 2^E]$ contains 2^{D-1} floating point numbers which are spaced equally in that subinterval.

Algebraically, set of floating point numbers $\mathcal{F}(D)$ is

- not a field, group or ring with respect to $+$ and $*$, but
- is *self-similar* as

$$2\mathcal{F} = \mathcal{F}.$$

The main approximation tool is the rounding function.

Definition: *A rounding function* $\phi : \mathbb{R} \to \mathcal{F}(D)$ *satisfies*

- $\phi(-x) = -\phi(x)$ for $x \in \mathbb{R}$ (odd function)
- $x_1 < x_2 \Rightarrow \phi(x_1) \leq \phi(x_2)$ (monotone function)

Definition: *A rounding function ϕ is optimal if*

$$|\phi(x) - x| \leq |y - x|, \quad \text{for all } y \in \mathcal{F}(D).$$

Note that there multiple optimal rounding functions, however, for use the particular choice of an optimal rounding function makes no difference.

For all optimal rounding functions one has the following error bound:

Proposition: *Let $\phi : \mathbb{R} \to \mathcal{F}(D)$ be an optimal rounding function. Then the rounding error satisfies*

$$\phi(x) - x = \delta\,|x|$$

for some δ satisfying

$$|\delta| \leq \frac{1}{2^{D-1} - 1}.$$

proof:

As $\phi(x) \in \mathcal{F}(D)$ there exists an integer $n \in [2^{D-1}, 2^D)$ and an integer E such that

$$\phi(x) = \pm n\, 2^{-D} \cdot 2^E.$$

One can see that

$$|x| \in (n-1, n+1)\, 2^{E-D}$$

and has the same sign as $\phi(x)$. Consequently,

$$|x| \geq (2^{D-1} - 1)\, 2^{E-D} = \left(\frac{1}{2} - \frac{1}{2^D}\right) 2^E.$$

Furthermore, one derives

$$|\phi(x) - x| \leq 2^{E-D}$$

and thus

$$|\phi(x) - x| \leq \frac{2^{E-D}}{(1/2 - 1/2^D)2^E} = \frac{1}{2^{D-1} - 1}.$$

■

The rounding error is thus bounded relatively, i.e., one can guarantee that a fixed number of digits of the rounded number are accurate.

In the following we will only consider functions on $\mathcal{F}(D)$ which are defined as real functions by

$$f_D(x) = \phi(f(x)).$$

We call these functions floating point functions and also consider the case where x and $f(x)$ are arrays of floating point numbers by rounding component-wise.

A special class of floating point functions define the floating point arithmetic operations defined, e.g., as

$$x_1 +_D x_2 = \phi(x_1 + x_2).$$

The thus defined *floating point arithmetic* does not satisfy the usual laws of arithmetic, instead, one has:

Proposition:

- floating point addition and multiplication are commutative
- $x_1 -_D x_2 = x_1 +_D (-x_2)$
- floating point operations are neither associative not distributive
- in general $(x_1 -_D x_2) +_D x_2 \neq x_1$
- in general $\phi(\sqrt{x *_D x}) \neq x$

We will in the following often just use $x_1 + x_2$ for floating point additions as well if no confusion is possible.

Floating point computations

Our computations will take one polynomial system and compute another one using mostly arithmetic operations. As each arithmetic operation on floating point numbers introduces an error in general we will end up with an error in the resulting polynomial system and thus in the resulting solutions. In the next section we consider an example of the computations illustrated for linear systems of equations and in the section after that we will discuss how the error in the polynomial system may affect the solution.

1.4 Gaussian elimination

The topic of this course is the direct (numerical) solution of polynomial systems of equations using floating point arithmetic. The algorithms are generalisations of Gaussian elimination algorithms. The key idea of Gaussian elimination is the transformation of a general linear system of equations to a triangular system of equations which has the same solutions. The motivation for this transformation is that triangular systems can be solved more conveniently.

Definition: *A triangular polynomial system of equations*

$$P(x) = \begin{bmatrix} p_1(x) \\ \vdots \\ p_n(x) \end{bmatrix}$$

has components $p_k(x)$ *which only depend on the variables* $x_k \ \ldots, x_n$*:*

$$p_k(x) = p_k(x_k, \ldots, x_n), \quad k = 1, \ldots, n.$$

Such a triangular system computes the components x_k^* of the solution by *back substitution:*

- for $k = n, \ldots, 1$ solve the univariate polynomial equations

  $$p_k(x_k, x_{k+1}^*, \ldots, x_n^*) = 0$$

  to get $x_k^* = x_k$.

Gaussian elimination achieves the transformation purely by linear operations and uses the observation

Proposition: *Let* $M \in \mathbb{R}^{n \times n}$ *be an invertible matrix. Then the two polynomial systems of equations* $P(x) = 0$ *and* $MP(x) = 0$ *have the same solutions.*

This proposition is valid for polynomial systems, however, in order to solve these systems one requires a bit more. For linear systems of equations, however, this result is sufficient to formulate the major direct solvers. Examples of matrices M used in the construction of the algorithms include

Examples:

- M is a *permutation matrix*
- $M = I - ue_k^T$ is an *elementary matrix* where u is such that $e_k^T u = 0$
- $M = I - 2uu^T$ is a *Householder transform (or reflection)* where $||u|| = 1$, M is symmetric and $M^{-1} = M$

For example, let

$$P^{(0)}(x_1, x_2) = \begin{bmatrix} 2x_1 + x_2 - 1 \\ 4x_1 - x_2 + 4 \end{bmatrix} = C^{(0)} \begin{bmatrix} x \\ 1 \end{bmatrix} = \begin{bmatrix} 2 & 1 & -1 \\ 4 & -1 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}$$

then choosing the elementary matrix

$$M^{(1)} = \begin{bmatrix} 1 & 0 \\ -2 & 1 \end{bmatrix}$$

provides the matrix

$$C^{(1)} = M^{(1)}C^{(0)} = \begin{bmatrix} 2 & 1 & -1 \\ 0 & -3 & 6 \end{bmatrix}$$

and the triangular system

$$P^{(1)}(x) = C^{(1)} \begin{bmatrix} x \\ 1 \end{bmatrix} = \begin{bmatrix} 2x_1 + x_2 - 1 \\ -3x_2 + 6 \end{bmatrix}.$$

More generally, for a given $P^{(0)}(x) \in (\mathcal{P}_1^s)^n$ with $s = n$ the following algorithm reduces $P^{(0)}$ to an equivalent (same solutions) triangular system of equations. Note that in this case

$$P^{(0)}(x) = C^{(0)}\underline{\mathbf{x}} = [\,A, b\,] \begin{bmatrix} x \\ 1 \end{bmatrix}$$

and the algorithm computes the coefficient matrices $C^{(k)}$ of the polynomial systems $P^{(k)}(x) = C^{(k)}\underline{\mathbf{x}}$:

Algorithm: (Gaussian elimination)

- $C^{(0)} = [\,A, b\,]$
- $C^{(k)} = (I - l^{(k)}e_k^T)\,C^{(k-1)}, \quad k = 1, \ldots, n-1$ where
  - $l_k = (0, \ldots, 0, l_{k+1}^{(k)}, \ldots, l_n^{(k)})$
  - $l_j^{(k)} = C_{jk}^{(k-1)} / C_{kk}^{(k-1)},\ j = k+1, \ldots, n.$
  - $e_k = (0, \ldots, 0, 1, 0, \ldots, 0)^T$

Note that Gaussian elimination *breaks down* if $C_{kk}^{(k-1)} = 0$ and $C_{kj}^{(k-1)} \neq 0$ for $j > 0$. Furthermore, *numerical instability* occurs if $C_{kk}^{(k-1)}$ is very small compared to the other elements in the same column. To avoid these effects the rows of $C^{(k-1)}$ are permuted before the elimination step guaranteeing that

$$|C_{kk}^{(k-1)}| \geq |C_{jk}^{(k-1)}|, \quad \text{for all } j > k.$$

The resulting algorithm is then called *Gaussian elimination with partial pivoting.* In some cases, the results may have poor accuracy despite the application of pivoting. This may be due to ill-conditioning of the original system. This will be further discussed in the next section. Approaches to deal with this make use of other knowledge available about the solution including sparsity or smoothness.

1.5 Solving systems with uncertainty

The is connected to chapter 3 of the book by Stetter.

We would like to answer questions regarding polynomial systems of equations of the form

$$P(x) = 0$$

like

- Is x (or set of x) a solution?
- Does the polynomial system $Q(x) = 0$ have the same solutions?

We will say that $P(x)$ is *well posed* if

- There exists a set V of solutions, i.e., $V = \{x \mid P(x) = 0\} \neq \emptyset$.
- The set V depends continuously on P.

Now of course one does require to define a topology on the polynomial systems $P(x)$ and the sets V – this shall be done later. At this stage we remark that polynomial systems are computationally challenging as small perturbations of the coefficients C (recall that $P(x) = C\underline{\mathbf{x}}$) may lead to empty solution sets V if we use real arithmetic. A very simple example is the quadratic equation $x^2 = 0$ which has one solution $x = 0$. Perturbing this a little to get $x^2 + \epsilon = 0$ for some small $\epsilon > 0$ gives us a problem with no real solution no matter how small ϵ is. Thus the equation $x^2 + c = 0$ is ill-posed for $c = 0$.

We now introduce a formalism to model what is going on. We will slightly deviate from Stetter by using probablity distributions. Assume that $P(x)$ is only approximately known. We will consider two cases:

- We know a polynomial system $P_\delta(x)$ which satisfies $\|P_\delta - P\| \leq \delta$ for some (typically unknown) $\delta \geq 0$ or
- we know a sample $P(x, \omega)$ taken from some (typically unknown) distribution of polynomial systems.

Solving $P_\delta(x) = 0$ or $P(x, \omega) = 0$ directly may give solutions with very large errors or no solutions at all.

We will now use two different classes of distances between two polynomial systems $P_1(x) = C_1\underline{\mathbf{x}}$ and $P_2(x) = C_2\underline{\mathbf{x}}$. The first class is just the l_q distance between the functions defined on some interval $[a, b]$, i.e.,

$$d(P_1, P_2) = \|P_1 - P_2\|_q,$$

and

$$\|P_1 - P_2\|_q = \left(\int_a^b (P_1(x) - P_2(x))^q \, dx \right)^{1/q}$$

for $q > 1$ and

$$\|P_1 - P_2\|_\infty = \max_{x \in [a,b]} |P_1(x) - P_2(x)|.$$

The second class of distances is defined as some norm of the distance of the coefficient matrices, i.e.,

$$d(P_1, P_2) = \|C_1 - C_2\|.$$

Assume that we have an implementation (an algorithm) of a function $F : \mathcal{A} \to \mathcal{Z}$ where $\mathcal{A}$ and $\mathcal{Z}$ are normed vector spaces. Then the Lipschitz constant L_F of F is such that

$$\|F(b) - F(a)\| \leq L_F \|b - a\|.$$

If b is an approximation (or perturbation) of a then the Lipschitz constant tells us how much a perturbation of a affects the function value $F(a)$. For example, if $F(a)$ is a matrix vector product $F(a) = Ma$ we have

$$\|M(b - a)\| \leq \|M\| \, \|b - a\|$$

and $L_F = \|M\|$. If we consider relative errors we have

$$\frac{\|F(b) - F(a)\|}{\|F(a)\|} \leq \kappa_F \frac{\|b - a\|}{\|a\|}.$$

The relative error is mostly considered for linear problems and from the definition it follows that

$$\kappa_F = \|M\| \, \|M^{-1}\|$$

if $F(a) = Ma$. Note that for linear problems the condition number for multiplying with a matrix and for multiplication with the inverse (i.e. solving a linear system of equations) is the same. The condition number κ_M is always larger than one and is equal to the ratio of the largest to the smallest singular value.

We call a problem well-conditioned, if L_F or κ_F (the condition number) is small and ill-conditioned otherwise.

For the solution of ill-posed problems one often uses regularisation. A particular example uses l_1 based minimisation as this may improve sparsity. A possible algorithm would be to interleave orthogonal elimination steps (using Householder matrices) with with steps which set the values of the coefficient matrix C which are very small to zero. Orthogonal elimination maintains the norm of $\|P(x)\|_2$ while setting small sized elements C_j to zero would reduce the norm

$$\|C\|_\infty = \sup_z \frac{\|Cz\|_\infty}{\|z\|_\infty}.$$

Finally, one may consider backward error analysis, where the effects of the errors occurring during the floating point computations are modelled as data errors which occur before the computations. One can then use the Lipschitz constant or condition number to get bounds on the errors in the result.

A totally different approach would consider Bayesian methods. Here one starts with a prior for the coefficients of C, i.e., a probablity density $p(C)$. Then one considers the likelihood of any polynomial system $C\underline{\mathbf{x}}$ given a system with errors $P(x,\omega)$. This likelihood is a conditional probility of $p(C(\omega) \mid C)$. For example, errors could be normally distributed with density $\gamma \exp(-\|C(\omega) - C\|_2^2/(2\sigma^2))$ where γ is a normalisation constant. The posterior density of C is then proportional to

$$p(C)\ \exp(-\|C(\omega) - C\|_2^2/(2\sigma^2)).$$

The so-called MAP (maximum aposteriori method) would then estimate C as the maximum of the aposteriori density and a thorough Bayesian analysis would do a comprehensive study of the posterior.