MA5233 Computational Mathematics

Lecture 23: Eigenvalues

Simon Etter

2019/2020

Eigenvalues

Eigenvalue problem

Given $A \in \mathbb{R}^{n \times n}$, find $\lambda_k \in \mathbb{C}$ and $x_k \in \mathbb{C}^n$ such that $Ax_k = \lambda_k x_k$.

Application 1: resonance frequencies

- Consider an airplane wing which is slightly pushed out of its resting position and then let go.
- Denote by $d(t) \in \mathbb{R}^n$ the dislocation of some reference positions on the wing at time t.

  Dislocation means that resting position is given by $d(t) = 0$.
- Evolution of $d(t)$ is modeled by some ODE $\ddot{d}(t) = F\big(d(t)\big)$.
- We reduce this ODE to a first order ODE $\dot{y} = f(y)$ by introducing $y_1(t) = d(t)$, $y_2(t) = \dot{d}(t)$.
- Linearising around the steady state $y = 0$, we obtain $\dot{y} \approx \nabla f(0)\, y$.
- Switching to $w := V^{-1}y$ with Λ, V the eigendecomposition of $\nabla f(0) = V \Lambda V^{-1}$, we obtain decoupled ODEs $\dot{w}_k = \lambda_k w_k$.

Eigenvalues

Application 1: resonance frequencies (continued)

- Let us now attach an engine to the wing which oscillates at some fixed frequency.
- This engine introduces a driving term into our ODE: $\dot{w} = \lambda w + e^{\mu t}$.
  Note that $e^{\mu t}$ oscillates if μ is imaginary.
- We are interested in steady-state solution for which the amplitude $A \neq A(t)$ is constant in time.
- We therefore make the ansatz $w(t) = A\, e^{\mu t}$ which yields

$$\dot{w} = A\mu\, e^{\mu t} = A\lambda\, e^{\mu t} + e^{\mu t} = \lambda w + e^{\mu t}.$$

- This equation is satisfied for all t if

$$A\mu = A\lambda + 1 \quad \Longleftrightarrow \quad A = \frac{1}{\mu - \lambda}.$$

- We observe that $|A| \gg 1$ if $\mu \approx \lambda$.

Eigenvalues

Application 1: resonance frequencies (continued)

Conclusions:

- Eigenvalues correspond *resonance frequencies* of physical structures.
- Knowing these frequencies is important in engineering applications.
- Wing may break if it has a resonance frequency near frequency of the engine.
- Famous example of what happens if you ignore resonance: `https://youtu.be/XggxeuFDaDU`

Eigenvalues

Application 2: quantum mechanics

Classical mechanics:

- State of a particle is described by position x and momentum p.
- Energy of particle in state (x, p) is given by $E = \frac{p^2}{2m} + V(x)$.

Quantum mechanics:

- State of a particle is described by wave function $\psi : \mathbb{R}^3 \to \mathbb{R}$.
- The only states which have well-defined energies are the eigenfunctions of the Hamiltonian operator H given by

$$(H\psi)(x) := -\frac{\Delta\psi(x)}{2m} + V(x)\,\psi(x).$$

- For ψ such that $H\psi = E\psi$, the energy is given by the corresponding eigenvalue E.

Eigenvalues

Application 2: quantum mechanics (continued)

Many everyday phenomena are consequences of the quantum-mechanical nature of electrons.

Examples: material strength, ductility, conductivity, colour.

We can numerically predict these properties using the following workflow:

- Assemble Hamiltonian $H \in \mathbb{R}^{n \times n}$ based on location and type of atoms.
- Determine eigenpairs E_k, ψ_k of H.
- Compute quantity of interest based on $(E_k, \psi_k)_{k=1}^{m}$.

A few more physical details:

- Each state can hold only a single electron (Pauli exclusion principle).
- Electrons typically occupy lowest-energy states at room temperature.

Combined, these two points imply that if the system contains m electrons, then only the states $k \in \{1, \ldots, m\}$ contain electrons (assuming the states are sorted such that $E_1 \leq E_2 \leq \ldots$).

Eigenvalues

Application 2: quantum mechanics (continued)

An important quantity of interest is the total energy E_{tot} of an electronic system, which is given by

$$E_{\text{tot}} := \sum_{k=1}^{m} E_k = \operatorname{Tr}\big(f(H)\big) \quad \text{where} \quad f(E) := \begin{cases} 1 & \text{if } E \leq E_m, \\ 0 & \text{otherwise.} \end{cases}$$

We define $f(H) := V\, f(\Lambda)\, V^{-1}$ for a diagonalisable matrix $H = V\Lambda V^{-1}$, where $f(\Lambda)_{k\ell} := f(\Lambda_{kk})\, \delta_{k\ell}$. Second equality follows from $\operatorname{Tr}\big(V\, f(\Lambda)\, V^{-1}\big) = \operatorname{Tr}\big(f(\Lambda)\big)$.

Take-ways from Applications 1 & 2

- Eigenvalue problems fall into two categories:
  - Find few eigenvalues (e.g. single λ closest to some reference value μ).
  - Find many or all eigenvalues.
- In most applications, the matrix in question is symmetric.
- Full eigendecompositions are often used as a means to evaluate matrix functions. Another important example:

$$\dot{y} = Ay, \quad y(0) = y(0) \qquad \Longleftrightarrow \qquad y(t) = e^{At} y_0.$$

Eigenvalues

Limitations of eigenvalue algorithms

It is well-known that eigenvalues of $A \in \mathbb{R}^{n \times n}$ can be computed as the roots of the characteristic polynomial $p(\lambda) := \det(A - \lambda I)$.

The converse is also true: every polynomial $p(\lambda)$ is the characteristic polynomial of some matrix A.

It is known that there exists no general and finitely computable formula for the roots of polynomials of degree ≥ 5.

Conclusion: eigenvalue solvers for matrix sizes $n \geq 5$ must be iterative.

Consequence

All eigensolvers iteratively improve some trial eigenpairs until convergence. Properties of exact eigenpairs give us some guidance on how to choose trial eigenpairs, see next slide.

Eigenvalues

Properties of eigenvalues and -vectors

If $A \in \mathbb{R}^{n \times n}$ is arbitrary:

- Eigenvalues and -vectors may be real or complex.
- Eigenvectors for distinct eigenvalues are linearly independent.
- There may be less than n eigenvectors.

First of the above points means we must use complex arithmetic even for real matrices. This makes the algorithms more computationally demanding but could be handled.

Second and third points have more severe consequences:

- Linear independence is not well-defined if rounding errors are present. Example: $(1\ 0)^T$ and $(1\ \varepsilon)^T$ are linearly independent, but it could be that the two vectors are equal in exact arithmetic.
- We do not know how many trial eigenvectors to use since A may have less than n eigenvectors.

All of these issues disappear for symmetric matrices.

Eigenvalues

Properties of eigenvalues and -vectors

If $A \in \mathbb{R}^{n \times n}$ is symmetric:

- Eigenvalues and eigenvectors are real.
- Eigenvectors for distinct eigenvalues are orthogonal.
- There are exactly n mutually orthogonal eigenvectors

These nice properties of the symmetric eigenproblem may be recovered to some extent by replacing the nonsymmetric eigendecomposition with Schur decomposition.

Thm: Schur decomposition

For any matrix $A \in \mathbb{R}^{n \times n}$, there exists an orthogonal matrix $Q \in \mathbb{C}^{n \times n}$ and an upper-triangular matrix $R \in \mathbb{C}^{n \times n}$ such that $A = QRQ^H$.

Furthermore:

- Diagonal of R are the eigenvalues of A.
- Eigenvectors can be recovered by determining the kernels of $R - \lambda_k I$.
- Columns q_k of Q form nested invariant subspaces of A:

$$Aq_k = \sum_{\ell=1}^{k} q_\ell \, R_{\ell k} \in \text{span}\{q_1, \ldots, q_k\}.$$

Eigenvalues

Remarks

- All nonsymmetric eigensolvers first compute Schur decomposition and then compute eigenvectors in a later stage if needed.
- Matrix functions can be computed based on Schur rather than eigendecomposition.

  B. Parlett. *A recurrence among the elements of functions of triangular matrices.* Linear Algebra and its Applications (1976), doi:10.1016/0024-3795(76)90018-5

Outline for rest of lecture

- Algorithms for computing few eigenvalues.
- Algorithms for computing all eigenvalues.

Eigenvalues

Def: Rayleigh quotient

Given $A \in \mathbb{R}^{n\times n}$ and $x \in \mathbb{R}^n$, the Rayleigh quotient $r(x)$ is given by

$$r(x) := \frac{x^T A x}{x^T x}.$$

Rayleigh quotient theorem

Assume (λ_k, x_k) are the eigenpairs of A. Then,

- $r(x_k) = \lambda_k$ for all k.
- If A is symmetric, then $\nabla r(x) = 0 \iff x \in \{x_1, \ldots, x_n\}$.

Proof. First part: $r(x_k) = \frac{x_k^T A x_k}{x_k^T x_k} = \lambda_k \frac{x_k^T x_k}{x_k^T x_k} = \lambda_k$.

Second part follows from

$$\nabla r(x) = \tfrac{2}{x^T x}\big(\tfrac{A+A^T}{2} x - r(x)\, x\big).$$

Corollary

$$\big|\lambda_k - r(x)\big| = \begin{cases} \mathcal{O}\big(\|x - x_k\|^2\big) & \text{if } A \text{ is symmetric,} \\ \mathcal{O}\big(\|x - x_k\|\big) & \text{otherwise.} \end{cases}$$

Eigenvalues

Thm: Power iteration

Assume $A \in \mathbb{R}^{n\times n}$ is symmetric and its eigenpairs λ_k, x_k are such that

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \ldots \geq |\lambda_n|.$$

Assume $x \in \mathbb{R}^n$ satisfies $x_1^T x \neq 0$. Then,

$$\left\| \frac{A^k x}{\|A^k x\|} \pm x_1 \right\| \leq C \left(\frac{|\lambda_2|}{|\lambda_1|} \right)^k.$$

Proof. Expand x in terms of eigenvectors x_k, i.e. $x = \sum_{\ell=1}^n c_\ell x_\ell$ where $c_1 = x_1^T x \neq 0$. We then have

$$A^k x = \sum_{\ell=1}^n c_\ell \, \lambda_\ell^k \, x_\ell,$$

which shows that $A^k x \to c_1 \lambda_1^k x_1$.

Remark

Power iteration also works for nonsymmetric matrices. The above proof generalises to nonsymmetric case if we replace expansion in eigenvectors with expansion in Jordan vectors.

Eigenvalues

Simultaneous iteration

Assume $A \in \mathbb{R}^{n \times n}$ is symmetric and its eigenpairs λ_k, x_k are such that

$$|\lambda_1| > \ldots > |\lambda_m| \geq |\lambda_{m+1}| \geq \ldots \geq |\lambda_n|.$$

Let $X \in \mathbb{R}^{n \times m}$ be arbitrary. Denote by $Q^{(k)} \in \mathbb{R}^{n \times m}$ the thin Q-factor of $A^k X$, and denote its columns by $q_\ell^{(k)}$. Then, under some technical assumptions on X we have $\|q_\ell^{(k)} \pm x_\ell\| \leq C\, a^k$ for some $C > 0$, $a \in (0, 1)$ and all $\ell \in \{1, \ldots, m\}$.

Remark

Precise statement and proof of this result requires notion of angle between subspaces, which is beyond the scope of this module.

The gist is that simultaneous iteration works much like power iteration, but it finds the m eigenvectors of largest magnitude rather than just one.

Eigenvalues

Inverse iteration

- Power iteration finds the eigenpair (λ_1, x_1) whose eigenvalue λ_1 is largest in magnitude.
- Eigenpair (λ_k, x_k) closest to some given $\mu \in \mathbb{C}$ can be found by applying power iteration to the matrix $(A - \mu I)^{-1}$. This is known as inverse iteration.
- Inverse iteration can also be used to speed up convergence if we have a good guess for the eigenvalue λ that we are looking for.

Eigenvalues

Other algorithms for computing few eigenpairs

As with iterative solvers for linear systems, there are many algorithms for computing a selected subset of eigenvalues.

Power, simultaneous and inverse iteration are popular because they are (relatively) easy, but they are typically not the most efficient algorithms.

Well-known alternatives:

- Arnoldi / Lanczos algorithm: similar to Krylov subspace methods. Good for finding eigenvalues of smallest and largest magnitude.
  Terminology: Arnoldi / Lanczos *algorithms* are eigensolvers. Arnoldi / Lanczos *iteration* determines orthogonal bases for Krylov subspaces.
- Jacobi-Davidson: good for finding interior eigenvalues.
- LOBPCG: determine largest / smallest eigenpair by maximising / minimising Rayleigh quotient.
  Faster than Lanczos if a good preconditioner for A is available.

Eigenvalues

QR algorithm

- Terminology: *QR algorithm* is an eigensolver. *QR factorisation* is $QR = A$.
- One of the best algorithms for computing all eigenpairs.
  There are other algorithms for this task which may outperform QR under certain circumstances, but no known algorithms significantly beats QR.
- Based on simultaneous iteration with block size $m = n$, but includes several tricks to make the algorithm faster.
- Most important trick: before we start the simultaneous iteration, we determine an orthogonal matrix Q such that

$$Q^T A Q \text{ is } \begin{cases} \text{tridiagonal if } A \text{ is symmetric.} \\ \text{Hessenberg if } A \text{ is nonsymmetric.} \end{cases}$$

- Such a Q can be determined using a finite number of Householder reflections (recall Lecture 4).
- Determining Q takes $\mathcal{O}(n^3)$ FLOPs, which is as expensive or more expensive than running simultaneous iteration until convergence to machine precision, see next slide.

Eigenvalues

FLOPs required by QR algorithm

	nonsymmetric case		symmetric case	
	without Schurvectors	with	without eigenvectors	with
transformation to Hessenberg/tridiagonal form	$\frac{10}{3}n^3$	$\frac{14}{3}n^3$	$\frac{4}{3}n^3$	$\frac{8}{3}n^3$
real double step Hessenberg/tridiagonal QR algorithm (2 steps per eigenvalues assumed)	$\frac{20}{3}n^3$	$\frac{50}{3}n^3$	$24n^2$	$6n^3$
total	$10n^3$	$25n^3$	$\frac{4}{3}n^3$	$9n^3$

Table taken from Chapter 3 of lecture note on
"Numerical Methods for Solving Large Scale Eigenvalue Problems" by Prof. P. Arbenz
(`http://people.inf.ethz.ch/arbenz/ewp/lnotes.html`).

Main take-away: even though eigenproblems in principle require an infinite number of operations, computing $A = V\Lambda V^T$ can be done in $\mathcal{O}(n^3)$ FLOPs in practice.

Eigenvalues

References and further reading

- L. N. Trefethen and D. Bau. *Numerical Linear Algebra.* Society for Industrial and Applied Mathematics (1997),
- P. Arbenz. *Numerical Methods for Solving Large Scale Eigenvalue Problems*.
  `http://people.inf.ethz.ch/arbenz/ewp/lnotes.html`

Trefethen & Bau provide a good overview and intuition, Arbenz gives all the details.