Computational Physics

Fully discrete problems

Where do discrete problems come from?

Only few problems in physics are truly discrete, e.g.

- quantum-mechanical spin

Most discrete problems are in fact abstractions:

- planets of the solar system, the bulb of a pendulum or the atoms of a Helium gas are approximated as point masses
- Ohmic resistance, capacitances and inductances in a circuit are lumped into point-like elements (resistors, capacitors, inductors) connected by featureless signal lines;
  similar in technical optics (objectives, microscopes)
- molecular orbitals are approximated as hybridizations of atomic orbitals

Very often: "Discrete" physics from weak interaction of "objects" that are held together by strong internal interactions.

A type of approximate/perturbative approach.

Types of discrete problems: Response problems

A response problem (inverse problem) is of the form

$$f(x) = s,$$

with given function $f(x)$, given vector s and unknown vector x.

These problems are notoriously hard if $f(x)$ is nonlinear, especially if $\dim\{x\} > \dim\{s\}$.

For linear functions $f(x) = Ax + c$ with square matrix A, this problem reduces to a standard linear problem

$$Ax = b.$$

Typical examples:

- small signal response of electronic circuits
- deformation under load in "discrete" mechanical structures
- equilibrium states in nonlinear systems (e.g. DC operating point in electronic circuits)

Types of discrete problems: Stationary problems

Consider an explicit ODE in time

$$\mathrm{i}\partial_t x(t) = f[x(t)] \qquad \text{with some function } f(x).$$

Stationary states are states that are constant in time except for a phase factor $x(t) = x \exp(-\mathrm{i}\omega t)$

Stationary states usually only exist for linear ODEs: $f(x) = Ax$ with square matrix A. This reduces the problem the matrix eigenvalue problem

$$Ax = \omega x$$

Typical examples:

- energy eigenstates in discrete quantum systems
- resonance frequencies in mass-spring systems or RCL-circuits

Solving linear eigenvalue problems

How to solve matrix problems of the type

$$Ax = \lambda x.$$

Typical methods to solve these problems:

- Dense, full, "exact" diagonalization, e.g. Jacobi method or QR-decomposition (found in numerical libraries)
- iterative methods, e.g. fix point iteration to find *some* eigenpairs
- Krylov-subspace versions of the iterative methods (e.g. ARPACK)

Basic fix point iteration

What is the effect of applying A to an arbitary vector y?

Expand y in eigenbasis:

$$y = \sum_n \alpha_n x_n \qquad \Rightarrow \qquad Ay = \sum_n \alpha_n A x_n = \sum_n \alpha_n \lambda_n x_n.$$

Applying a matrix to an arbitrary vector "amplifies" the eigenvectors with large eigenvalues that contribute to the vector.

Fix point iteration to find approximate eigenpairs:

1. Pick an arbitrary normalized vector y, $||y|| = 1$.
2. Calculate $x = Ay$ and normalize $x \leftarrow x/||x||$
3. If the change $||x - y||$ is sufficiently small, then x is an approximate eigenvector and terminate (other termination conditions possible). Otherwise set $y = x$ and go back to "2".
4. From the eigenvector find the corresponding eigenvalue via the Rayleigh quotient $\lambda = \langle y, Ay \rangle / \langle y, y \rangle$.

Fix point iteration will find the eigenvalue with the *greatest* norm that was contained in the original starting vector.

Variations of the fix point iteration

There are two operations that change the eigen*values* but not the eigen*vectors* of a matrix (so-called spectral transformations):

- matrix inversion:
  $Ax = \lambda x \qquad \Rightarrow \qquad A^{-1}x = \lambda^{-1}x.$
- adding a constant μ to the diagonal:
  $Ax = \lambda x \qquad \Rightarrow \qquad (A + \mu\mathbb{I})x = Ax + \mu x = \lambda x + \mu x = (\lambda + \mu)x.$

This leads to the two main variants of the fix point iteration that are very common in practice:

- Applying the fix point iteration to $\tilde{A} = A^{-1}$ *("inverse iteration")*.
  The largest eigenvalue $\tilde{\lambda}$ of $\tilde{A}$ is always the *smallest* eigenvalue $\lambda = \tilde{\lambda}^{-1}$ of A.
- Applying the fix point iteration to $\tilde{A} = (A - \mu\mathbb{I})^{-1}$ *("shift-and-invert")*.
  The largest eigenvalue $\tilde{\lambda}$ of $\tilde{A}$ is the eigenvalue $\lambda = \mu + \tilde{\lambda}^{-1}$ of A that is *closest* to μ.

In-class example

Consider the matrix with approximate eigenvalues

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 5 & 6 & 7 \\ 1 & 1 & 8 & 9 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

$$\lambda_1 \approx 11.8789$$
$$\lambda_2 \approx 2.8673$$
$$\lambda_3 \approx -1.4668$$
$$\lambda_4 \approx 0.7206$$

Implement fix point iteration, inverse iteration and shift-and-invert to find the largest, smallest and the negative eigenvalue.

Using one of these methods find an eigenvalue of the matrix

$$B = \frac{1}{5} \begin{pmatrix} 3 & 4 & 0 \\ -4 & 3 & 0 \\ 0 & 0 & 5 \end{pmatrix}$$

Solving linear response problems

Linear response problems can be formulated as a matrix problem

$$Ax = b$$

and solved in the usual ways, e.g. by inverting the matrix via Gauss-Jordan.

In practice, there might be reasons why this is not a good idea:

- matrix inversion scales in time with $\mathcal{O}(N^3)$
- a matrix requires $\mathcal{O}(N^2)$ amount of memory
- the matrix A might not be given explicitly, but just as a prescription of how to find the result of a matrix-vector product
- usually, one does not need the inverse, just the solutions for a few rhs.

Often, discrete problems are small (because of their idealised nature) and direct inversion is perfectly fine.

Do not implement it yourself, use numeric libraries!

Approximate linear solvers: Krylov methods

If full inversion is not viable, there are alternatives to find *approximate* solutions to linear matrix problems:

- Iterative solvers, especially Krylov-methods[1]
- Incomplete direct solvers (UMFPACK, PARDISO, Super-LU)

Here, we only mention Krylov-methods

Basic idea: The inverse of an operator can be written as a series

$$(\mathbb{I} + M)^{-1} = \mathbb{I} - M + M^2 - M^3 + M^4 + \dots.$$

This is guaranteed to converge if M is "small".

So it seems plausible that we can find the product $(\mathbb{I} + M)^{-1} b$ in the vector space spanned by $\{b, Mb, M^2 b, M^3 b, M^4 b, \dots\}$
(so-called *Krylov space*).

[1] Literature: Y. Saad, "Iterative Methods for Sparse Linear Systems", SIAM 2003

Approximate linear solvers: Krylov methods (cont'd)

The general approach to $Ax = b$ is always something like this:

1. Start with the simplest possible Krylov space $\mathcal{K}^{(1)} = \{b\}$
2. With some Krylov space $\mathcal{K}^{(n)} = \{k^{(1)}, k^{(2)}, \ldots k^{(n)}\}$ given, find the action of A restricted to $\mathcal{K}^{(n)}$, i.e. construct the "reduced matrix" $H_{ij}^{(n)} = \langle k^{(i)}, Ak^{(j)} \rangle$
3. Solve the reduced problem $H^{(n)}\tilde{x}^{(n)} = (1, 0, 0, \ldots)^T$.
4. Check how well the back-projected solution $x^{(n)} = \sum_i \tilde{x}_i^{(n)} k_i^{(i)}$ solves the actual full problem.
5. Terminate if the error is small enough, otherwise extend the Krylov space with additional vector $k^{(n+1)} = Ak^{(n)}$ and go back to "2".

Note how you never need the matrix A itself, but only its action on a vector.

There are methods that do this in slightly different ways, e.g.

- conjugate gradients (CG) and its variants (BiCG, BiCGstab)
- GMRES and its variants

Solving nonlinear problems

Nonlinear problems are all vector-valued problems of the form $f(x) = 0$, where the function is *not* of the linear form $f(x) = Ax - b$.

While linear problems have their accuracy issues, the number of solutions is known and there are methods that are guaranteed to work at least in princple.

Neither is true for nonlinear problems.

Because of the unknown number of solutions, nonlinear problems are always solved iteratively, where the starting point is critical.

Some methods are:

- Relaxation method (basically fix point iteration for weak nonlinearities)
- Newton's method
- Nonlinear Krylov-methods (nonlinear conjugate gradients, restarted GMRES)

In-class example

Consider the nonlinear matrix problem

$$\left[\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} + ||x|| \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}\right] x = \begin{pmatrix} 0 \\ 1/2 \end{pmatrix}.$$

Construct a sequence of approximate solutions x_n by solving the linear problems

$$\left[\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} + ||x_{n-1}|| \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}\right] x_n = \begin{pmatrix} 0 \\ 1/2 \end{pmatrix}, \quad x_0 = \begin{pmatrix} 0 \\ 0 \end{pmatrix},$$

What happens if the rhs is $(0, 1)^T$ instead of $(0, 1/2)^T$?

Numerical errors in discrete problems

Sources of numerical error in discrete problems:

- round-off (fundamental)
- convergence of iterative solvers (can be selected)
- applicability of the idealizations that led to the model (basically "discretization errors")

We will restrict ourselves to uniquely solvable linear problems of the form:

$$Ax = b; \qquad A \text{ invertible.}$$

We introduce some round-off, e.g. in the representation of b:

$$Ax' = b + \Delta; \qquad ||\Delta|| \text{ small.}$$

What can we say about the error $||x - x'||$ of the solution?

What are round-off errors and where do they come from?

Round-off appears because we approximate real numbers with a finite number of digits.

In the common floating-point arithmetic, this means that numbers have a relative accuracy of about 10^{-15}.

Every mathematical operation involves rounding, which in most cases does not affect the relative accuracy too much.

There is one operation that can ruin any *relative* accuracy: subtraction!

Consider two numbers a and b with relative accuracy 10^{-15} and $|a - b| \approx 10^{-7}$.

Their sum still has relative accuracy 10^{-15}, but their difference has only 10^{-8}, because the first 7 digits cancel each other.

Really bad round-off errors typically come from *subtracting* nearly equal numbers!

Round-off errors and matrices: Condition number

- Consider the linear problem $Ax = b$ with (round-off) perturbation Δ:

$$Ax' = b + \Delta; \qquad \Rightarrow \qquad x' = A^{-1}(b + \Delta); \qquad \frac{||\Delta||}{||b||} \ll 1.$$

- We expand b and Δ into the eigenbasis of A:
  $b = \sum_n \alpha_n x_n; \qquad \Delta = \sum_n \beta_n x_n$
  $A^{-1}b = \sum_n \frac{\alpha_n}{\lambda_n} x_n; \qquad A^{-1}\Delta = \sum_n \frac{\beta_n}{\lambda_n} x_n$
- The largest relative error occurs if b and Δ are eigenvectors to the largest and smallest eigenvalues, respectively.
- Therefore, we find for the relative error of the solution:

$$\frac{||A^{-1}\Delta||}{||A^{-1}b||} \leq C \frac{||\Delta||}{||b||} \qquad \text{with } C = \frac{\lambda_{\text{max}}}{\lambda_{\text{min}}}.$$

- C is called the *condition number* of A.
- How small errors (e.g. round-off) impact a matrix problem is determined by the eigenvalue spectrum of A.

In-class example

Implement a program that evaluates the exponential function via the Taylor series

$$\exp(x) \approx f^{(N)}(x) = \sum_{n=0}^{N} \frac{1}{n!} x^n.$$

1. Pick a few positive values of x and plot how the relative error $|f^{(N)}(x) - \exp(x)|/\exp(x)$ evolves as you increase N.
2. Do the same with some negative values for x.
3. Why is the expansion less accurate for some values of x?
4. Find an way to get good accuracy for all a.

Homework

Implement and test the transfer matrix method for multiple scattering of waves in a stack of materials with different indices (see problem sheet).