Type & Complexity of algs.

Direct:
in predictable fixed # operations gives exact ans.
of ops done exactly.
 spared: \(f_1, f_2, f_3, \ldots, f_n, \sin, \cos, \text{etc.}
\)

Eq.
\[
\begin{bmatrix}
1 & 0 \\
0 & 1
\end{bmatrix}^x = 1
\]

solve \(Ax = b\) by Gaussian elim.
add inúb each of rows.
to each other.
to set diag \(= 0\) below diagonal.
How long take? \# its below diagonal.
\[
\frac{\text{cost}}{\text{b}} \approx n^3
\]
to add a multi.g row to another.
\(= n^3\) add., \(= n\) multi. = \(2n\) ops.
Total \(\approx n^3\) ops.
Can we do better? \(O(\log n)\)?

Iterative:
gets closer to answer but never quite
even if ops exact.

\(O(n^3)\) polynomial time.
vs. \(O(\log n)\).

Any eqs? sum a fcn. on \(\{0,1,\ldots, 2^{\text{work}}\}\).
Complicated.

Ex.
Krylov space methods for \(Ax = b\).

Start with \(x_0 \in \mathbb{R}^n\) eg a random guess for \(x\).
Alg.
gives \(x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow \ldots\)
Each step needs a vector by \(A\).

\[
\text{Residuals } r_k = Ax_k - b \quad \text{got smaller.}
\]

\[
\|r_k\| \rightarrow 0
\]
as \(k \rightarrow \infty\).

You run alg. until \(\|r_k\| < \varepsilon\) where \(\varepsilon\) is
desired error level.

\(Q:\) does this control error? \(\|r_k - x\|\) ? \(A\): see later.

Cost: \(\text{dep. on } n \& \varepsilon\).

Why better than, direct?

Each step \(O(?) \text{ ask: } O(\varepsilon)\)
\& if \(A\) well-behaved, \(\text{resid.}\) always fast.

\(O^n\) check:

say \(\|r_k\| \leq \frac{1}{2} \|r_1\|
\]
then \(\|r_k\| \leq C2^{-k}\).

say \(\|r_k\| \leq 10^{-b} \text{ error, } k \approx 30\).

\[
\text{Cost: } O(n^2 k) = O(n^2 \log \frac{1}{\varepsilon}) = O(\log \frac{1}{\varepsilon}^2)
\]

\(A\) often sparse \((\text{off-diag. zero})\) so can apply

to vector faster, eg \(O(n)\) per step.

Much faster than direct.

High acc. \& we always need.
Roots of polynomial order p:
- eg. p=2 quadratic formula works
- p=3 cubic
- p=4 quartic
- p=5...

Ax = Ax, eigenvalues Aj

n=2: \[
\begin{pmatrix}
\alpha_1 - \lambda & \alpha_2 \\
\alpha_3 & \alpha_4 - \lambda
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2
\end{pmatrix} =
\begin{pmatrix}
0 \\
0
\end{pmatrix}
\]
singular \( \Rightarrow \) det = 0
\( (\alpha_1 - \lambda) (\alpha_4 - \lambda) - \alpha_3 \alpha_2 = 0 \)
2nd order poly for 2 roots.

General n: 1st order poly
- if can solve poly roots can solve EVP
- (actually terrible way to do it).

Could there be direct alg for n \( \geq 5 \) EVP?

(That didn't go poly root) general.

- Am example of direct vs iter alg?
  
  Direct vs Iter.

Direct not always slower: conf. on exact direct soln June 23-29.

\textit{Ratio of convergence:}

A) Eg: say want
    calculate: \( f_1 + f_2 + \frac{1}{2} + \frac{1}{3} + \cdots = \sum k^{-2} \) series.
    In real life you must stop, eq. truncate @ \( n \) term: 
    \( f_k = \sum_{n=1}^{k} n^{-2} \)

slow converge: \[
\frac{f_k}{10^6} = \frac{1}{1.614936}
\]

- Problem: \( f_k \) exactly slow (increases). Tightly converge up to where frozen.

- Ask: \( k=10^3 \rightarrow \) digit = ? 3.

error \( \varepsilon_k = f_k - f \approx n^{-1} \)

- \( \varepsilon_k \) can, order 1. - extremely slow eg. \( 10^{-15} \) need \( 10^{15} \) ops.

Can prove this: \( |\varepsilon_k| = \sum_{n=k}^{\infty} n^{-2} \leq \int_{k}^{\infty} n^{-2} dn = \frac{1}{k} \)

\textbf{Lemma:} \( |\varepsilon_k| \leq 1/k \) - can prove observed order.

\textbf{Asymptotic prefactor, this lecture:}
Usually enough to prove \( \varepsilon_k = O(k^p) \) for: then plot \( k, \varepsilon_k \) to best extract! 

(Inf) \( - \text{slope} = -p \) 

\( k \approx (\log(k_0)/s)^{1/p} \)

B) Basic iteration methods for EVP:

\( A \in \mathbb{R}^{n \times n} \), \( A^T = A \) \( \Rightarrow \) diagonalizable: \( A = V \Lambda V^T \), \( V \) s col. are eigenvectors: \( V_1, \ldots, V_n \), \(-\text{unique}\).

\( \text{diag} \in \lambda_j \exists i = 1 \) \( |\lambda_i| > |\lambda_2| > \ldots \) up to \( k \).

\( \forall x \in \mathbb{R}^n \), \( x = \sum_{i=1}^n \lambda_j v_j \) 

then \( Ax = \sum_{i=1}^n \lambda_j A v_j = \sum_{i=1}^n \lambda_j x_j v_j \), idea: effect of largest \( \lambda_j \)'s grow much

\( \Rightarrow \) \( V_1 \) dominates.

Power iteration: pick \( x^{(0)} \) randomly, 

for \( k = 1, 2, \ldots \)

\( x^{(k)} = A x^{(k-1)} \)

\( x^{(k)} \leftarrow x^{(k)} / \|x^{(k)}\| \) normalize.

Each iter \( O(n^2) \) if \( A \) dense.

Note: \( x^{(k)} = \sum_{i=1}^n \lambda_i^k x_1 v_1 + \sum_{i=1}^n \lambda_i^k x_2 v_2 + \ldots \)

\( k \rightarrow 0 \Rightarrow \lambda_1 > |\lambda_2| > \ldots \)

Thus \( \|x^{(k)} - (\pm v_1)\|_2 = O(|\lambda_1|^{-k}) \)

Error of \( x^{(k)} \) from an eigenv.

\( \text{Proof: } \lambda_1 = V_1^T x^{(0)} \neq 0. \)

Type of Convergence: \( \varepsilon_k = O(r^k) \) for some \( 0 < r < 1 \) called geometric, exponential, linear.

Better or worse than algebraic?

\( r^k = O(k^p) \)

Try: \( p = 1, 2, \ldots \)

Take ratio: \( \lim_{k \to \infty} \frac{r^k}{k^p} = \lim_{k \to \infty} \frac{k}{r^k} = \frac{\text{d}}{\text{d}k} \frac{e^{\pm k} \cdot k^{-1}}{e^{\pm k} \cdot (k)^{-1}} = 0 \)

\( \Rightarrow \text{yes } r^k = O(k^{-1}). \)

Also: \( r^k = O(k^{-p}) \)

\( k \rightarrow \text{a little } - \) \( k^p \rightarrow 0 \)

\( \frac{\text{d}}{\text{d}x} \frac{e^{x} \cdot x^{-1}}{e^{x} \cdot (x)^{-1}} = 0 \)

For each \( p = 1, 2, \ldots \), \( r^k = O(k^{-p}) \).