Lecture X: Iterative methods for sparse systems (class)

Sparse linear systems

$$\underbrace{A}_{n\times n}\,\vec{x} = \vec{b}$$

Think $n \lesssim 10^8$, # of • $\approx c \cdot n$

$\Rightarrow$ few GB (gigabytes) to store A

~~giga~~ ~~tera~~ petabytes to store A^{-1}

~years to compute A^{-1} or LU (Gaussian elim.)

$O(n^3)$ complexity

Today: Iterative methods for approximately solving sparse linear systems

Examples:

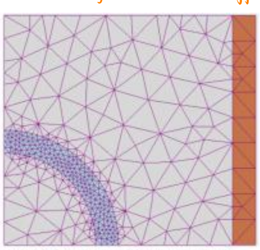
Graph analysis

Facebook has ~10^9 users,
Google indexes ~10^{14} webpages } with sparse connections

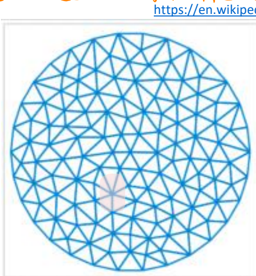
Differential equation discretization

finite difference or finite element methods

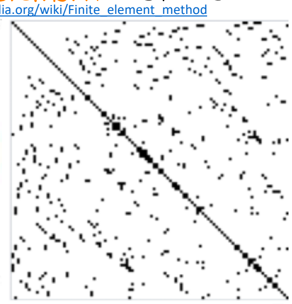
https://en.wikipedia.org/wiki/Finite_element_method



FEM mesh created by an analyst prior to finding a solution to a magnetic problem using FEM software. Colours indicate that the analyst has set material properties for each zone, in this case a conducting wire coil in orange; a ferromagnetic component (perhaps iron) in light blue; and air in grey. Although the geometry may seem simple, it would be very challenging to calculate the magnetic field for this setup without FEM software, using equations alone.



Solving the two-dimensional problem $u_{xx} + u_{yy} = -$ in the disk centered at the origin and radius 1, with zero boundary conditions.
(a) The triangulation.



(b) The sparse matrix L of the discretized linear system

$$a_{ij} = \begin{cases} \text{nonzero} & \text{if there is an edge } (i,j) \text{ (or } (i,i)) \\ 0 & \text{otherwise} \end{cases}$$

$\Rightarrow$ max # entries in a row = 1 + max. vertex degree

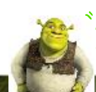
Example: Foster & Fedkiw '01  "Shrek"

Figure 7: A fully articulated animated character interacts with viscous mud. The environment surrounding the character is 150x200x150 cells. That resolution is sufficient to accurately model the character filling his mouth with mud. A 3D control curve is used to eject (spit) the mouthful of mud later in the sequence. This example runs at three minutes per frame.

$= 10^{6.7}$

Remark: $1000 \times 1000 \times 1000 = 10^9$ (3D grid)

Example: ECMWF 10-day weather forecasts

Remark: $1000 \times 1000 \times 1000 = 10^9$ (3D grid)

Example: ECMWF 10-day weather forecasts

9km horizontal,
137 vertical levels
$\Downarrow$
10^9 grid points
(~100? vars at each point)

https://www.ecmwf.int/en/about/media-centre/news/2016/new-forecast-model-cycle-brings-highest-ever-resolution

GRADIENT DESCENT

Define a cost function

$$C(\vec{x}) = \|A\vec{x} - \vec{b}\|^2 \qquad \vec{b} = Ax^*$$

$$= (x-x^*)^T A^T A (x-x^*)$$

or

$$C(\vec{x}) = (x-x^*)^T A (x-x^*)$$

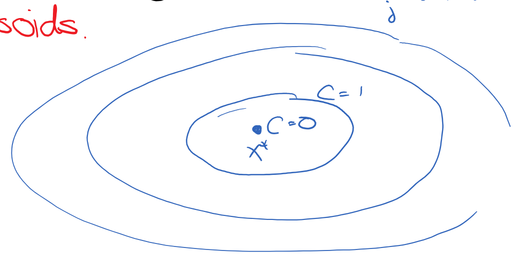
if A is already pos. def.

$$A = \sum_j \lambda_j \vec{v}_j \vec{v}_j^T \qquad \lambda_j \geq 0, \quad \{v_j\} \text{ o.n.}$$

Finding solution $\vec{x}^* \iff$ Minimizing $C(\vec{x})$

$$c = (x-\vec{x}^*)^T A (x-\vec{x}^*) = \sum_j \lambda_j \cdot |\vec{v}_j \cdot (x-x^*)|^2$$

Observe: C's level sets are ellipsoids.



Examples:

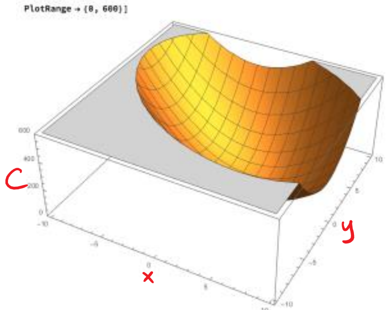
$A = \begin{pmatrix} 1 & -2 \\ 2 & 2 \end{pmatrix}$;

b = A.{2, 2}

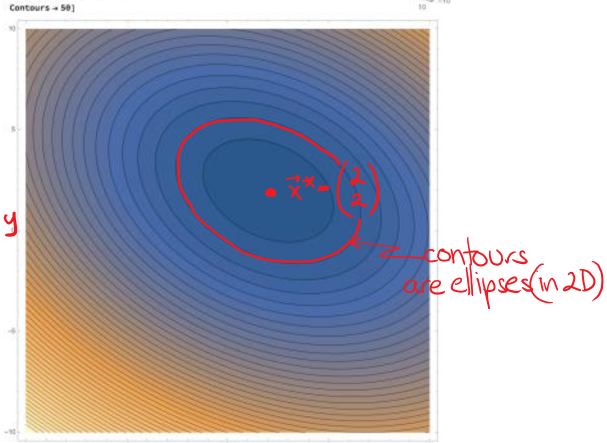
{-2, 8}

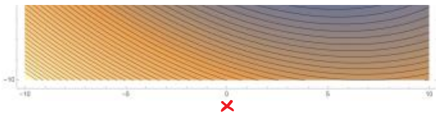
```
Plot3D[
 Norm[A.{x, y} - b]^2,
 {x, -10, 10}, {y, -10, 10},
 PlotRange -> {0, 600}]
```

cost function $C(x,y)$




```
ContourPlot[
 Norm[A.{x, y} - b]^2,
 {x, -10, 10}, {y, -10, 10},
 Contours -> 50]
```



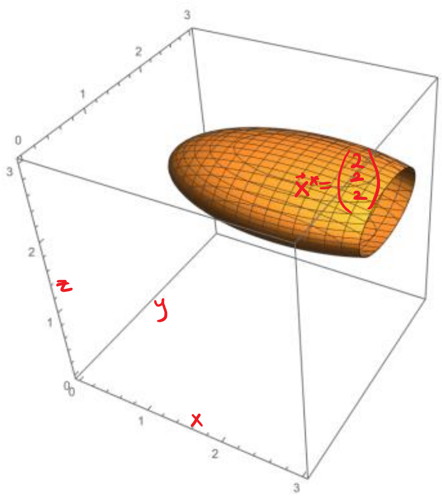


Ellipsoid in higher dimensions:

```
A = DiagonalMatrix[{1, 2, 4}];
b = A.{2, 2, 2};
ContourPlot3D[
 Norm[A.{x, y, z} - b] ^2 == 3,
 {x, 0, 3}, {y, 0, 3}, {z, 0, 3}
]
```

Goal: Find center, the solution $\vec{x}^*$

$A\vec{x}^* = \vec{b}$, $C(\vec{x}^*) = 0$.



"Gradient descent" = move downhill in the steepest direction, $-\nabla C$

For $C(x) = (x - x^*)^T A (x - x^*)$,

$$= \sum_{ij} A_{ij} (x - x^*)_i (x - x^*)_j$$

$$\frac{\partial C}{\partial x_k} = A_{kk} \cdot 2 (x - x^*)_k + \sum_{i \neq k} A_{ik} (x - x^*)_i + \sum_{j \neq k} A_{kj} (x - x^*)_j$$

$$\frac{\partial}{\partial x_k} (x - x^*)_j = \delta_{jk}$$

$$= 2 \sum_i A_{ki} (x - x^*)_i$$

$$\nabla C = \begin{pmatrix} \partial_{x_1} C \\ \partial_{x_2} C \\ \vdots \end{pmatrix} = 2 A (x - x^*) = 2 (Ax - b)$$

$$x^{t+1} = x^t - \beta \nabla C(x^t)$$

$$= x^t - \beta \cdot 2 (Ax^t - b)$$

What is the optimal choice for β?

$$C(x - \beta \underbrace{(Ax - b)}_{\substack{||| \\ \vec{r} \\ \text{residual}}}) = C(\vec{x} - \beta \vec{r})$$

$$= (x - \beta r - x^*)^T A (x - \beta r - x^*)$$

$$= \beta^2 (r^T A r) - \beta (r^T \underbrace{A(x - x^*)}_{= Ax - b = r} + (x - x^*)^T A r) + (x - x^*) A (x - x^*)$$

residual

$$= \beta^2(r^TAr) - \beta(r^TA(x-x^*) + (x-x^* \dots)$$

$$+ \underbrace{(x-x^*)A(x-x^*)}_{C(x)} \qquad = Ax-b=r$$

$$= \beta^2(r^TAr) - 2\beta\|r\|^2 + C(x)$$

$$0 = \frac{d}{d\beta}C(x-\beta r)$$

$$= 2\beta(r^TAr) - 2\|r\|^2$$

$$\Rightarrow \beta = \frac{\|r\|^2}{r^TAr}$$

$$\Rightarrow \boxed{x^{t+1} = x^t - \frac{\|\vec{r}\|^2}{\vec{r}^TA\vec{r}} \cdot \vec{r}} \quad \text{where } \vec{r} = Ax^t - b$$

Gradient descent algorithm:

Input: A, b, x_0, t_{max}, tol

$\vec{r}_0 = A\vec{x}_0 - \vec{b}$

FOR $t = 0, 1, \dots, t_{max}$:

IF $\|\vec{r}_t\| < (tol)\cdot\|b\|$, BREAK

$$\boxed{\begin{array}{l} \vec{z} = A\vec{r}_t \qquad \beta = \dfrac{\|\vec{r}_t\|^2}{\vec{r}_t \cdot \vec{z}} \\ \vec{x}_{t+1} = \vec{x}_t - \beta\vec{r}_t \\ \vec{r}_{t+1} = \vec{r}_t - \beta\vec{z} \end{array}}$$

RETURN $\vec{x}_t$

Mathematica:

```
GradientDescent[A_, b_, x0_, tmax_, tol_] := Module[{list, x, r, rnorm, bnorm, z, β, t},
  bnorm = Norm[b];
  x = x0;
  list = {x};
  r = A.x - b;
  For[t = 0, t < tmax, t++,
   rnorm = Norm[r];
   If[rnorm < tol bnorm, Break[]];];
   z = A.r;
   β = rnorm^2 / r.z;
   x = x - β r;
   r = r - β z;
   AppendTo[list, x];
  ];
  "This returns not just the (approximate) solution, but the whole list of values
   x0,...,xT tried along the way.";
  list
 ];
```

Example:

```
A = ( 2    -1.5
     -1.5   4  );
"A is positive definite:"
Eigenvalues[A]

"The exact solution:"
xstar = {3, 3}

b = A.{3, 3};
```

A is positive definite:

{4.80278, 1.19722}

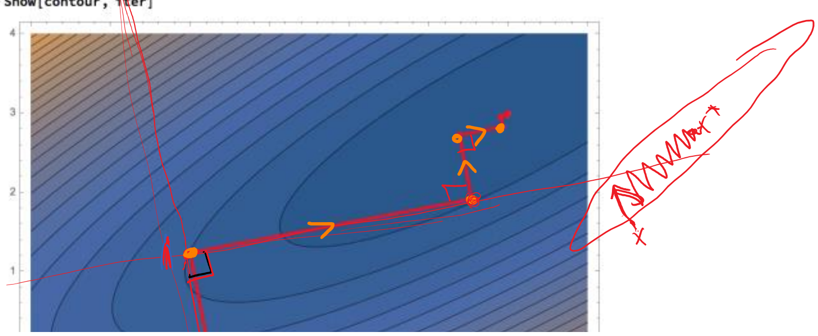
The exact solution:

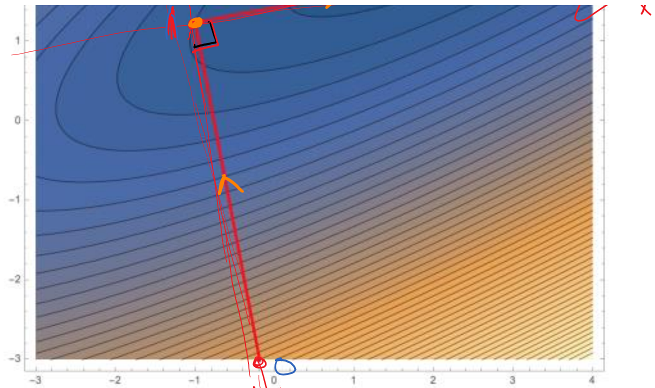
{3, 3}

```
x0 = {0, -4};
list = GradientDescent[A, b, x0, 1000, 10^-4]
```

{{0, -4}, {-1.0037, 1.24155}, {2.53695, 1.91955}, {2.38203, 2.72858}, {2.92853, 2.83323}, {2.90462, 2.95811}, {2.98897, 2.97426}, {2.98528, 2.99353}, {2.9983, 2.99603}, {2.99773, 2.999}, {2.99974, 2.99939}, {2.99965, 2.99985}}

```
"Display the results:";
iter = ListPlot[list, PlotRange → 3 {{-1, 1}, {-1, 1}}, Joined → True,
   PlotStyle → {Red, Thickness[.005]}, PlotMarkers → Automatic];
contour = ContourPlot[Norm[A.{x, y} - b]^2, {x, -3, 4}, {y, -3, 4}, Contours → 50];
Show[contour, iter]
```





Observe: Consecutive steps are orthogonal, $r_{t+1} \cdot r_t = 0$ (Exercise)

Observe: This finds an approximate solution.

Smaller condition number $\Rightarrow$ Rounder ellipse
$\Rightarrow$ Faster convergence

Remark: If A is not positive definite, the descent rule is
$\vec{x}_{t+1} = \vec{x}_t - \beta A^T \vec{r}_t$

Remark: Although not generally the fastest way of solving a set of linear equations, gradient descent is very robust, and **many** variants are used in **many** applications.

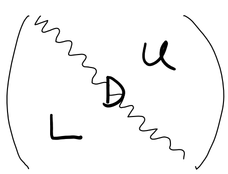
Convergence: $C_t \le C_0 \cdot \left(1 - \frac{2}{\kappa+1}\right)^{2t} \approx C_0 \left(1 - \frac{1}{\frac{\kappa+1}{2}}\right)^{\frac{\kappa+1}{2} \frac{4t}{\kappa+1}} \le C_0 e^{-\frac{4t}{\kappa+1}}$

$\Rightarrow$ to get $\frac{C_t}{C_0} \le 2^{-p}$, set $t = \frac{\log 2}{4}(\kappa+1)p$

Remark: A not pos. def. $\Rightarrow O(\kappa^2)$ convergence

Conjugate gradient algorithm has $O(\sqrt{\kappa})$ convergence

Jacobi and Gauss-Seidel iterative methods




```
A = (  2    -1.5
     -1.5    4  );
L = LowerTriangularize[A, -1]
U = UpperTriangularize[A, 1]
Diag = DiagonalMatrix@Diagonal[A]

{{0., 0.}, {-1.5, 0.}}

{{0., -1.5}, {0., 0.}}

{{2, 0}, {0, 4}}
```

Jacobi iteration

$A\vec{x} = \vec{b}$

$\Rightarrow D\vec{x} = -(L+U)\vec{x} + \vec{b}$

$$\boxed{\vec{x}_{t+1} = -D^{-1}(L+U)\vec{x}_t + D^{-1}\vec{b}}$$

$\vec{x}_{t+1} = E\vec{x}_t + \vec{q}$

```
"Jacobi iteration";
e = -Inverse[Diag].(L + U);
q = Inverse[Diag].b;

x = {0, -4};
listJacobi = {x};
For[k = 1, k ≤ 1000, k++,
  x = e.x + q;
  AppendTo[listJacobi, x];
 ];
```

Gauss-Seidel iteration

$\Rightarrow (D+L)\vec{x} = -U\vec{x} + \vec{b}$

$$\boxed{\vec{x}_{t+1} = -\underbrace{(D+L)^{-1}U\vec{x}_t}_{\text{compute using back-subst. (don't compute } (D+L)^{-1})} + (D+L)^{-1}\vec{b}}$$

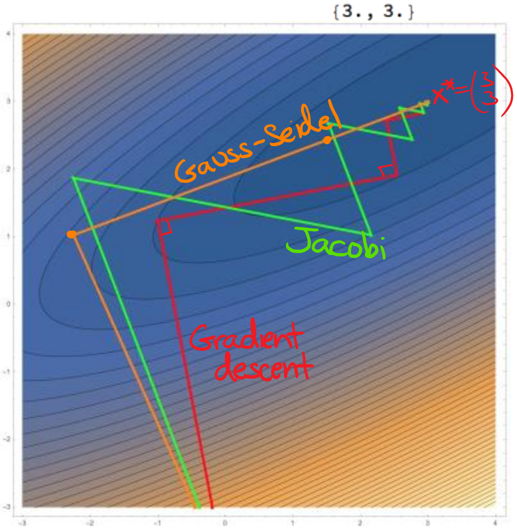
```
"Gauss-Seidel iteration";
e = -Inverse[Diag].(L + U);
q = LinearSolve[Diag + L, b];

x = {0, -4};
listGS = {x};
For[k = 1, k ≤ 1000, k++,
  x = -LinearSolve[Diag + L, U.x] + q;
  AppendTo[listGS, x];
 ];
```

```
    x = e.x + q;
    AppendTo[listJacobi, x];
  ];
x
```

{3., 3.}

```
    x = -LinearSolve[Diag + L, U.x] + q;
    AppendTo[listGS, x];
  ];
x
```

{3., 3.}



PRECONDITIONING

<u>Intuition</u>: Iterative methods are slow when κ the condition # is large, i.e., the ellipse is long and thin.

Let's round the ellipsoid.

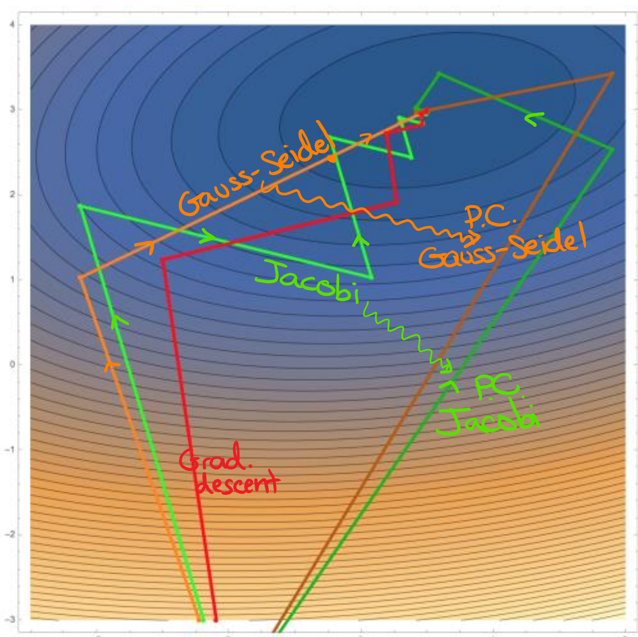
$$A\vec{x} = \vec{b} \iff \underbrace{M^{-1}A}_{\text{hopefully closer to } I \text{ (rounder)}}\vec{x} = M^{-1}\vec{b}$$

M should be
- "close to" A
- easy to solve $M\vec{x} = \vec{b}$

<u>Example</u>: Jacobi preconditioner

$$M = \underset{\text{diagonal part of A}}{D}$$

<u>Example</u>: $A = \begin{pmatrix} 2 & -1.5 \\ -1.5 & 4 \end{pmatrix}$ $\quad M = \begin{pmatrix} 3 & -1.5 \\ -1.5 & 3 \end{pmatrix}$



Stopping criteria

When should you stop?

$$x - x^* = A^{-1}(Ax - b) \qquad b = A \cdot A^{-1} b$$

$$\Rightarrow \|x - x^*\| \le \|A^{-1}\| \cdot \|Ax - b\| \qquad \Rightarrow \|b\| \le \|A\| \cdot \|x^*\|$$

$$\Rightarrow \frac{\|x - x^*\|}{\|x^*\|} \le \underbrace{\|A^{-1}\| \cdot \|A\|}_{\substack{\kappa \\ \text{condition } \#}} \cdot \frac{\|Ax - b\|}{\|b\|}$$

$\Rightarrow$ You are close to $\vec{x}^*$ if $\kappa \frac{\|Ax - b\|}{\|b\|}$ is small.

If you don't know κ, hopefully you can guess an upper bound.

Convergence analysis: for $A \succeq 0$

$$\vec{x}_t = \vec{x}_{t-1} - \beta(A\vec{x}_{t-1} - \vec{b})$$

$$= x_{t-1} - \beta A(x_{t-1} - x^*)$$

$$\Rightarrow (x_t - x^*) = (x_{t-1} - x^*) - \beta A(x_{t-1} - x^*)$$

$$= (I - \beta A)(x_{t-1} - x^*)$$

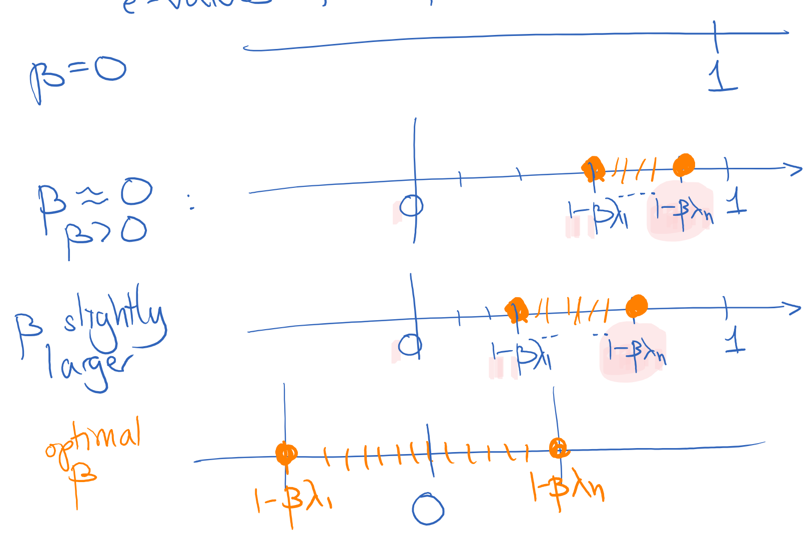
$$\|Ax\| \le \|A\| \cdot \|x\|$$

$$\Rightarrow \|x_t - x^*\| \le \|I - \beta A\| \cdot \|x_{t-1} - x^*\|$$

$$\kappa = \frac{\lambda_1}{\lambda_n}$$

A has evalues $\lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_n \ge 0$

e-values of $I - \beta A$



$\Rightarrow I - \beta A$ has evalues $1 - \beta\lambda_1 \le 1 - \beta\lambda_2 \le \cdots \le 1 - \beta\lambda_n$

$$\|I - \beta A\| = \max\{1 - \beta\lambda_n, \beta\lambda_1 - 1\}$$

best choice for β is

$$1 - \beta\lambda_n = \beta\lambda_1 - 1$$

$$\Rightarrow \beta = \frac{2}{\lambda_1 + \lambda_n}$$

$$= 1 - \frac{2}{\lambda_1 + \lambda_n}\lambda_n$$

$$\stackrel{!}{=} 1 - \frac{2}{\lambda_1+\lambda_n}\lambda_n$$

$$\Rightarrow \|x_t - x^*\| \le \|I - \beta A\|^t \cdot \|x_0 - x^*\| = \left(1 - \frac{2\lambda_n}{\underset{\le 2\lambda_1}{\lambda_1+\lambda_n}}\right)^t \|x_0 - x^*\|$$

$$\le \left(1 - \frac{1}{\kappa}\right)^t \|x_0 - x^*\|$$

✓

$$\Rightarrow x_t - x^* = (x_{t-1} - x^*) - \beta A(x_{t-1} - x^*) \quad \text{since } Ax^* = b$$

$$= (I - \beta A)(x_{t-1} - x^*)$$

$$\Rightarrow \|x_t - x^*\| \le \|I - \beta A\| \cdot \|x_{t-1} - x^*\|$$

$$\le \cdots$$

$$\le \|I - \beta A\|^t \cdot \underbrace{\|x_0 - x^*\|}_{\text{initial error}}$$

If A's e-values are $\lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_n \ge 0$,
then $I - \beta A$ has e-values

$$1 - \beta\lambda_1 \le 1 - \beta\lambda_2 \le \cdots \le 1 - \beta\lambda_n$$

$$\Rightarrow \|I - \beta A\| = \max\{|1 - \beta\lambda_1|, |1 - \beta\lambda_n|\}.$$

The <u>best</u> choice of β will make these equal in magnitude, centered on 0:

$$\Rightarrow -(1 - \beta\lambda_1) = +(1 - \beta\lambda_n)$$

$$\Rightarrow \beta = \frac{2}{\lambda_1 + \lambda_n}$$

With this choice for β,

$$\|I - \beta A\| = 1 - \frac{2\lambda_n}{\underset{\le 2\lambda_1}{\lambda_1 + \lambda_n}} \le 1 - \frac{\lambda_n}{\lambda_1}$$

$$= 1 - \frac{1}{(\text{condition \# of } A)}$$

$$\Rightarrow \|x_t - x^*\| \le \left(1 - \frac{1}{\kappa}\right)^t \cdot \|x_0 - x^*\|$$

$$\Rightarrow t = \kappa \cdot \left(\log\frac{1}{\varepsilon} + \log\frac{1}{\|x_0 - x^*\|}\right) \text{ ensures } \|x_t - x^*\| \le \varepsilon$$

using $\left(1 - \frac{1}{\kappa}\right)^\kappa \le \frac{1}{e}$

<u>Convergence analysis</u>: general A

Convergence analysis: general A

$$x_t = x_{t-1} - 2\alpha A^T(Ax_{t-1} - b)$$

$$\Rightarrow x_t - x^* = (x_{t-1} - x^*) - 2\alpha A^TA(x_{t-1} - x^*)$$
$$= (I - 2\alpha A^TA)(x_{t-1} - x^*)$$

$\Rightarrow$ error magnitude

$$\|\vec{x}_t - \vec{x}^*\| \le \|I - 2\alpha A^TA\| \cdot \|\vec{x}_{t-1} - \vec{x}^*\|$$

If the e-values of A^TA are

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq 0$$

these are (singular values of A)^2

$I - 2\alpha A^TA$ has e-values

$$1 - 2\alpha\lambda_1 \le \cdots \le 1 - 2\alpha\lambda_n$$

$$\Rightarrow \|I - 2\alpha A^TA\| = \max_i |1 - 2\alpha\lambda_i|$$
$$\max\{|1-2\alpha\lambda_1|, |1-2\alpha\lambda_n|\}$$

The <u>best</u> choice of α will make these equal in magnitude, centered on 0:

$$\Rightarrow -(1 - 2\alpha\lambda_1) = +(1 - 2\alpha\lambda_n)$$

$$\Rightarrow \alpha = \frac{1}{\lambda_1 + \lambda_n}$$

With this choice for α,

$$\|I - 2\alpha A^TA\| = 1 - \frac{2\lambda_n}{\underbrace{\lambda_1 + \lambda_n}_{\le 2\lambda_1}} \le 1 - \frac{\lambda_n}{\lambda_1}$$

$$= 1 - \frac{1}{\text{condition \# of } A^TA}$$

$$= 1 - \frac{1}{(\text{condition \# of } A)^2}$$

$$\Rightarrow \|x^t - x^*\| \le \left(1 - \frac{1}{\kappa^2}\right)^t \cdot \|x^0 - x^*\|$$

$$\Rightarrow t = \kappa^2\left(\log\frac{1}{\varepsilon} + \log\frac{1}{\|x^0 - x^*\|}\right) \quad \text{since } \left(1 - \frac{1}{\kappa^2}\right)^{\kappa^2} \le e^{-1}$$

ensures $\|x^t - x^*\| \le \varepsilon$.

Simultaneous Diagonalizability of Matrices

"Normal" matrices are nice because they can be unitarily diagonalized.

A normal $\Rightarrow$ in some orthonormal basis,

$$A = \begin{pmatrix} \lambda_1 & & & 0 \\ & \lambda_2 & & \\ & & \ddots & \\ 0 & & & \lambda_n \end{pmatrix}$$

This makes matrix operations easy.

$$A^{100} = \begin{pmatrix} \lambda_1^{100} & & & 0 \\ & \lambda_2^{100} & & \\ & & \ddots & \\ 0 & & & \lambda_n^{100} \end{pmatrix} \quad \text{in this basis}$$

$$e^A = \begin{pmatrix} e^{\lambda_1} & & & 0 \\ & e^{\lambda_2} & & \\ & & \ddots & \\ 0 & & & e^{\lambda_n} \end{pmatrix}$$

$\Rightarrow$ trivial to solve differential equations $\frac{d}{dt}\vec{v} = A\vec{v}$

$A\vec{x} = \vec{b}$ is solved by

$$\vec{x} = \sum_{j=1}^{n} \frac{(v_j \cdot b)}{\lambda_j} \cdot \vec{v}_j \qquad \left(\text{since } \vec{b} = \sum_j (v_j \cdot b)\vec{v}_j\right)$$

$\text{Rank}(A) = \#$ of nonzero λ_j

$\text{Det}(A) = \lambda_1 \cdot \lambda_2 \cdots \lambda_n$

etc...

But what if you want to work with **two** matrices, at the same time?

Examples: (from physics)

Total energy (a Hermitian matrix) $=$ Kinetic energy $+$ Potential energy

Energy of two-particle system $=$ Energy of 1^{st} particle $+$ Energy of 2^{nd} particle $+$ Interaction energy

⋮

Problem:

Say that matrices A and B are both unitarily diagonalizable. $\Leftrightarrow$ normal

When are they diagonalizable using the same unitary?

Equivalently, when do they share a basis of eigenvectors?

Definition: Matrices A and B commute if

$$AB = BA.$$

Examples:

- Any two $n \times n$ diagonal matrices commute:

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}, \quad B = \begin{pmatrix} 3 & 0 \\ 0 & 4 \end{pmatrix}$$

$$\Rightarrow AB = \begin{pmatrix} 1\cdot 3 & 0 \\ 0 & 2\cdot 4 \end{pmatrix}, \quad BA = \begin{pmatrix} 3\cdot 1 & 0 \\ 0 & 4\cdot 2 \end{pmatrix} \checkmark$$

- All matrices commute with the identity:

$$I \cdot A = A \cdot I = A \quad \checkmark$$

- Not all matrices commute:

$$A = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}, \quad B = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

$AB = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \neq BA = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$

Theorem: For normal matrices A and B, there is an orthonormal basis of e-vectors for both A and B $\iff$ $AB = BA$.

Proof: Simultaneous diagonalized $\Rightarrow AB = BA$

$\exists$ unitary U st.

$A = U D_A U^{-1}$

same U

$B = U D_B U^{-1}$

$AB = U D_A D_B U^{-1}$
$\quad \|$
$BA = U D_B D_A U^{-1}$ ✓

Proof: $AB = BA \Rightarrow$ simultaneously diagonalized

Let's work in a basis where A is diagonal.

$$A = \left(\begin{array}{c|c|c} \begin{matrix} \lambda_1 & 0 \\ 0 & \lambda_1 \end{matrix} & 0 & 0 \\ \hline 0 & \begin{matrix} \lambda_2 & 0 \\ 0 & \lambda_2 \end{matrix} & 0 \\ \hline 0 & 0 & \ddots \end{array}\right) \qquad B = \left(\begin{array}{c|c|c} B_{11} & B_{12} & \cdots \\ \hline B_{21} & B_{22} & \cdots \\ \hline \vdots & \vdots & \ddots \end{array}\right)$$

$$AB = \left(\begin{array}{c|c|c} \lambda_1 B_{11} & \boxed{\lambda_1 B_{12}} & \cdots \\ \hline \lambda_2 B_{21} & \lambda_2 B_{22} & \cdots \\ \hline \vdots & \vdots & \ddots \end{array}\right) \qquad BA = \left(\begin{array}{c|c|c} \lambda_1 B_{11} & \boxed{\lambda_2 B_{12}} & \cdots \\ \hline \lambda_1 B_{21} & \lambda_2 B_{22} & \cdots \\ \hline \vdots & \vdots & \end{array}\right)$$

$\Rightarrow \lambda_1 B_{12} = \lambda_2 B_{12}$

but $\lambda_1 \neq \lambda_2$ $\Rightarrow B_{12} = (0)$

same for all B_{jk} with $j \neq k$

$$\Rightarrow \quad B = \left(\begin{array}{c|c|c} B_{11} & 0 & 0 \\ \hline 0 & B_{22} & 0 \\ \hline 0 & 0 & \ddots \end{array}\right) \quad \cdots \quad ✓$$

Proof:

Easy direction $\Rightarrow$:

Assume that A and B are diagonalized by the same unitary U:

$A = U D_A U^{-1}$

$B = U D_B U^{-1}$,

with D_A and D_B diagonal.

$$\Rightarrow AB = (U D_A U^{-1})(U D_B U^{-1})$$
$$= U D_A D_B U^{-1}$$
$$BA = U D_B D_A U^{-1}$$

Since D_A and D_B are diagonal, $D_A D_B = D_B D_A$, so $AB = BA$. ✓

Hard direction $\Leftarrow$:

Assume that A and B commute: $AB = BA$.

Work in a basis so that A is diagonal:

$$A = \begin{pmatrix} \lambda_1 \cdots \lambda_1 & & & \\ & \lambda_2 \cdots \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_k \cdots \lambda_k \end{pmatrix}$$ with $\lambda_1, \dots, \lambda_k$ distinct e-values

$$B = \begin{pmatrix} B_{11} & B_{12} & \cdots \\ B_{21} & B_{22} & \cdots \\ \vdots & \vdots & \ddots & B_{kk} \end{pmatrix}$$ written in the same basis

$$\Rightarrow AB = \begin{pmatrix} \lambda_1 B_{11} & \lambda_1 B_{12} & \cdots & \lambda_1 B_{1k} \\ \lambda_2 B_{21} & \lambda_2 B_{22} & \cdots & \lambda_2 B_{2k} \\ \vdots & \vdots & & \vdots \\ \lambda_k B_{k1} & \lambda_k B_{k2} & \cdots & \lambda_k B_{kk} \end{pmatrix}$$

$$BA = \begin{pmatrix} \lambda_1 B_{11} & \lambda_2 B_{12} & \cdots & \lambda_k B_{1k} \\ \lambda_1 B_{21} & \lambda_2 B_{22} & \cdots & \lambda_k B_{2k} \\ \vdots & \vdots & & \vdots \\ \lambda_1 B_{k1} & \lambda_2 B_{k2} & \cdots & \lambda_k B_{kk} \end{pmatrix}$$

Since $AB = BA$,

$\lambda_i B_{ij} = \lambda_j B_{ij}$ for all i, j

$\Rightarrow$ If $i \neq j$ then $B_{ij} = (0)$

Thus in this basis,

$$A = \begin{pmatrix} \lambda_1 \cdots \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 \cdots \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_k \cdots \lambda_k \end{pmatrix} \qquad B = \begin{pmatrix} B_{11} & 0 & \cdots & 0 \\ 0 & B_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & B_{kk} \end{pmatrix}$$

Since B is normal, $BB^\dagger = B^\dagger B = \begin{pmatrix} B_{11}B_{11}^\dagger & & 0 \\ & \ddots & \\ 0 & & B_{kk}B_{kk}^\dagger \end{pmatrix}$

(note that this is true in any basis, as $(UBU^\dagger)(UBU^\dagger)^\dagger = UBB^\dagger U^\dagger$)

$\Rightarrow$ each block B_{ii} is normal, too.

Choose a basis for each coordinate block so as to diagonalize the B_{ii}'s.

This does not affect A! (Since A is proportional to the identity on each block, $\lambda_i I$, and the identity I is the same in any basis.)

$\Rightarrow$ Now A and B are both diagonal. ✓ $\square$

Remark: The same theorem also extends to three or more matrices, with pairwise commutation, e.g.,

$$AB = BA, \quad AC = CA, \quad BC = CB.$$

Theorem: If A, B and C are normal matrices that pairwise commute, then there is a basis $\vec{v}_1, \dots, \vec{v}_n$ of simultaneous eigenvectors for all three:

$$A\vec{v}_j = \lambda_j^A \vec{v}_j, \quad B\vec{v}_j = \lambda_j^B \vec{v}_j, \quad C\vec{v}_j = \lambda_j^C \vec{v}_j.$$

In this basis all three matrices are diagonal:

$$A = \begin{pmatrix} \lambda_1^A & & 0 \\ & \lambda_2^A & \\ 0 & & \ddots \ \lambda_n^A \end{pmatrix}, \ B = \begin{pmatrix} \lambda_1^B & & 0 \\ & \lambda_2^B & \\ 0 & & \ddots \ \lambda_n^B \end{pmatrix}, \ C = \begin{pmatrix} \lambda_1^C & & 0 \\ & \ddots & \\ 0 & & \lambda_n^C \end{pmatrix}.$$

$$\frac{d}{dt} y_t = c \cdot y_t \Rightarrow y_t = e^{ct} y_0$$

$$\frac{d}{dt} \vec{y}_t = C\vec{y}_t \Rightarrow \vec{y}_t = e^{Ct} \vec{y}_0$$

$$" \sum_{j=0}^{\infty} \frac{(Ct)^j}{j!} "$$

$$e^t e^s = e^{t+s} = e^s e^t$$

$$\underset{\text{in general}}{e^A e^B \neq e^B e^A} \underset{\text{in general}}{\neq e^{A+B}}$$

Example: Matrix exponentials

For real or complex numbers a and b,

$$e^{a+b} = e^a e^b$$

$$(\text{eg. } e^{r_1 t_1} e^{r_2 t_2} = e^{r_1 t_1 + r_2 t_2})$$

What about matrices?

If $A = \mathcal{U}_A \underset{\text{diagonal}}{D_A} \mathcal{U}_A^{-1}$,

$$\exp(A) = e^A = \sum_{j=0}^{\infty} \frac{1}{j!} A^j = \mathcal{U}_A \cdot e^{D_A} \cdot \mathcal{U}_A^{-1}.$$

Similarly if $B = \mathcal{U}_B D_B \mathcal{U}_B^{-1}$.

What is e^{A+B}?

```
>> A = randn(2,2);  A(2,1) = A(1,2); A
B = randn(2,2);  B(2,1) = B(1,2); B
```

} random symmetric 2×2 matrices

```
A =

   4.8889e-01   7.2689e-01
   7.2689e-01  -3.0344e-01


B =

   2.9387e-01   8.8840e-01
   8.8840e-01  -1.1471e+00

>> expm(A+B)

ans =

   4.0240e+00   2.0574e+00
   2.0574e+00   1.1794e+00

>> expm(A) * expm(B)

ans =

   4.0214e+00   1.9054e+00
   2.2445e+00   1.1911e+00

>> expm(B) * expm(A)

ans =

   4.0214e+00   2.2445e+00
   1.9054e+00   1.1911e+00
```

$$\begin{pmatrix} 4.0214 & 1.9054 \\ 2.2445 & 1.1911 \end{pmatrix} \neq \begin{pmatrix} 4.0214 & 2.2445 \\ 1.9054 & 1.1911 \end{pmatrix}$$

Problem: $AB - BA \neq 0$

```
>> A * B - B * A

ans =

            0  -3.4349e-01
   3.4349e-01            0
```

$$\exp(A+B) = I + A + B + \underbrace{\frac{(A+B)^2}{2}}_{= \frac{A^2+AB+BA+B^2}{2}} + \cdots$$

$$\exp(A)\exp(B) = \left(I + A + \frac{A^2}{2} + \cdots\right)\left(I + B + \frac{B^2}{2} + \cdots\right)$$

$$= I + A + B + \left(AB + \tfrac{1}{2}A^2 + \tfrac{1}{2}B^2 + \right) \cdots$$

But if A and B commute, then $BA = AB$, so all the A's can be pulled to the left, and

$$e^{A+B} = e^A e^B = e^B e^A \quad (\text{if they commute!})$$

$$e^{A+B} = I + (A+B) + \frac{(A+B)^2}{2} + \cdots$$

$$= I + (A+B) + \frac{1}{2}\begin{pmatrix} A^2 + B^2 \\ + AB + BA \end{pmatrix} + \cdots$$

$$e^A e^B = \left(I + A + \frac{A^2}{2} + \cdots\right)\left(I + B + \frac{B^2}{2} + \cdots\right)$$

$$= I + (A+B) + \left(\frac{A^2}{2} + \frac{B^2}{2} + AB\right) + \cdots$$

$$e^B e^A = I + (A+B) + \tfrac{1}{2}(A^2 + B^2 + 2BA) + \cdots$$

if $AB = BA$, then

$$e^A e^B = e^{A+B} = e^B e^A$$

$$e^{A+B} = \lim_{m \to \infty} \left(e^{\frac{A}{m}} e^{\frac{B}{m}}\right)^m$$

$$\frac{d}{dt} \vec{y}_t = (A+B)\vec{y}_t$$

Lie-Trotter

Application: Time-dependent differential equations:

$$\frac{dy}{dt} = f(t)\,y(t)$$

$$\Rightarrow y(t) = y(0)\cdot \exp\left(\int_0^t f(s)\,ds\right).$$

But this is false for matrices

$$\frac{d\vec{y}}{dt} = F(t)\,\vec{y}(t)$$

$$\not\Rightarrow \vec{y}(t) = \exp\left(\int_0^t F(s)\,ds\right)\vec{y}(0)$$

unless all matrices $F(t)$ commute with each other!

If they don't commute, then there isn't a clean solution, and you'll have to solve the differential equation numerically, using a sufficiently fine discretization of time.

```
>> expm(A+B)

ans =

   4.0240e+00   2.0574e+00
   2.0574e+00   1.1794e+00
```

```
>> m = 100;
( expm(A/m) * expm(B/m) )^m

ans =

   4.0240e+00   2.0552e+00
   2.0596e+00   1.1794e+00

>> m = 1000;
( expm(A/m) * expm(B/m) )^m

ans =

   4.0240e+00   2.0572e+00
   2.0576e+00   1.1794e+00
```

$$\lim_{m\to\infty}\left(e^{A/m}e^{B/m}\right)^m = e^{A+B}$$

"Lie-Trotter formula"

https://en.wikipedia.org/wiki/Lie_product_formula

TRIVIA: Is this robust?

What if $\|AB-BA\| < \delta$?

Are they nearly simultaneously diagonalizable?

P. R. Rosenthal: Are almost commuting matrices near commuting pairs?. AMS Monthly **76**, 925 (1969).

first asked in the 1950s

H. Lin: Almost commuting self-adjoint matrices and applications. Fields. Inst. Commun. **13**, 193 (1995).

Theorem: There exists a function $\varepsilon = \varepsilon(\delta)$ such that, for any $n\times n$ Hermitian matrices A and B, with

$$\|AB-BA\| < \delta,$$

there exist commuting matrices A' and B' with

$$\|A-A'\| < \varepsilon(\delta)\ ,\ \|B-B'\| < \varepsilon(\delta).$$

Note: ε does not depend on n!

Theorem: [Hastings, 2008]

This holds for $\varepsilon(\delta) \sim \delta^{1/5}$.

Remark: These theorems are all false for almost-commuting triples of Hermitian matrices, and pairs of unitary matrices.

Example: "circulant matrix":

$$A = \begin{pmatrix} a_0 & a_1 & a_2 & a_3 & \cdots & a_{n-1} \\ & & & & & \\ a_{n-2} & a_{n-1} & a_0 & a_1 & a_2 & \cdots \\ a_{n-1} & a_0 & a_1 & a_2 & a_3 & \cdots & a_2 \\ \vdots & & & & & & a_1 \\ a_1 & a_2 & a_3 & \cdots & & & a_0 \end{pmatrix} = a_0 \underset{T^0}{I} + a_1 T + a_2 T^2 + \cdots$$

$$T = \begin{pmatrix} 0 & 1 & & \\ & 0 & 1 & \\ & & 0 & 1 \\ & & & \ddots \end{pmatrix} \qquad T^2 = \begin{pmatrix} 0 & 0 & 1 & & \\ & 0 & 0 & 1 & \\ & & 0 & 0 & 1 \\ & & & & \ddots \end{pmatrix}$$

$$T = \begin{pmatrix} 0 & 1 & & & \\ & 0 & 1 & & 0 \\ & & 0 & 1 & \\ & 0 & & 0 & 1 \\ & & & & 0 & 1 \\ 1 & & & & & 0 \end{pmatrix} \qquad T^2 = \begin{pmatrix} 0 & 0 & 1 & & & \\ & 0 & 0 & 1 & & \\ & & 0 & 0 & \ddots & \\ & & & \ddots & 0 & 1 \\ 1 & 0 & & & 0 & 0 \\ 0 & 1 & & & & 0 \end{pmatrix}$$

for any matrix C

$C \cdot C^p = C^p \cdot C = C^{p+1}$

$AT = TA$

$\Rightarrow$ they can be simultaneously diagonalized

let $\omega = e^{2\pi i/n}$

$$T \begin{pmatrix} 1 \\ \omega^j \\ \omega^{2j} \\ \vdots \\ \omega^{-j} \end{pmatrix} = \begin{pmatrix} \omega^j \\ \omega^{2j} \\ \omega^{3j} \\ \vdots \\ 1 \end{pmatrix} = \omega^j \begin{pmatrix} 1 \\ \omega^j \\ \omega^{2j} \\ \vdots \\ \omega^{-j} \end{pmatrix} \checkmark$$

$\uparrow$

Fourier basis for $j = 0, 1, \ldots, n-1$

orthogonal basis

$\Rightarrow$ Fourier basis diagonalizes A

$A = a_0 I + a_1 T^{-1} + a_2 T^{-2} + a_3 T^{-3} + \cdots + a_{n-1} T$

where $T_n = \begin{pmatrix} 0 & 0 & 0 & & 1 \\ 1 & & & & \\ & 1 & \ddots & & \\ & & & \ddots & \\ 0 & & & 1 & 0 \end{pmatrix}$

$= e_2 e_1^T + e_3 e_2^T + \cdots + e_n e_{n-1}^T + e_1 e_n^T$

the cyclic shift/translation, an orthogonal (permutation) matrix

Observe: "Waves are eigenvectors of translations"

Precisely: Let $\omega_n = e^{2\pi i/n}$

$\vec{u}_k = \frac{1}{\sqrt{n}}(1, \omega^k, \omega^{2k}, \omega^{3k}, \ldots, \omega^{(n-1)k})$

$\Rightarrow T\vec{u}_k = \frac{1}{\sqrt{n}}(\omega^{(n-1)k}, 1, \omega^k, \omega^{2k}, \ldots, \omega^{(n-2)k})$

$= \omega^{-k} \cdot \vec{u}_k$

an eigenvector!

(e.g., $k = 0$: $T \frac{1}{\sqrt{n}}(1, 1, \ldots, 1) = \frac{1}{\sqrt{n}}(1, 1, \ldots, 1)$.)

Corollary: The wave vectors $\vec{u}_0, \vec{u}_1, \ldots, \vec{u}_{n-1}$ form an orthonormal basis.

(This is called the Fourier basis.)

They are e-vectors of a normal matrix (T

Corollary: The matrix

$$\begin{array}{c} \\ \\ \\ j \end{array} \frac{1}{\sqrt{n}} \begin{pmatrix} 1 & 1 & 1 & 1 \\ \omega & \omega^2 & & \\ \omega^2 & & & \\ & & \omega^{jk} & \end{pmatrix} \begin{array}{c} k \\ \\ \\ \\ \end{array}$$

is unitary.

$\Rightarrow$ A commutes with T $\quad ([A,T]=0)$
$\Rightarrow$ It is diagonal in the Fourier basis
(Of course this can also be verified directly.)

~~Kronecker~~ product
"Tensor product"

$$\underset{m\times n}{A} \otimes \underset{o\times p}{B} = \begin{pmatrix} a_{11}B & a_{12}B & a_{13}B & \cdots \\ a_{21}B & a_{22}B & a_{23}B & \\ & \vdots & & \end{pmatrix}_{mo \times np}$$

kron(A, B)

$$\begin{array}{c} A \\ \| \end{array} \begin{array}{c} H \\ T \end{array} \begin{pmatrix} p \\ 1-p \end{pmatrix} \otimes \begin{array}{c} B \\ \| \end{array} \begin{pmatrix} q \\ 1-q \end{pmatrix} \begin{array}{c} H \\ T \end{array} = \begin{pmatrix} pq \\ p(1-q) \\ (1-p)(q) \\ (1-p)(1-q) \end{pmatrix} \begin{array}{c} HH \\ HT \\ TH \\ TT \end{array}$$

HW problem

$A\vec{u} = \lambda\vec{u} \qquad B\vec{v} = \sigma\vec{v}$

$$\Rightarrow (A\otimes B)(\vec{u}\otimes\vec{v}) = (\lambda\sigma)(\vec{u}\otimes\vec{v})$$

$$\begin{pmatrix} B & 2B \\ 0 & 3B \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 0 & 3 \end{pmatrix} \otimes B$$

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$