- The convergence rate in these algorithms depends a lot on the condition number of the matrix A.
  → ill-conditioned matrices have v. slow convergence rate
  → well-conditioned matrices have v. fast convergence rate.

(see later)

Idea ②: Instead of solving the system

$$Ax = b$$

solve

$$KAx = Kb$$

where the matrix KA is well-conditioned

→ This is called pre-conditioning.

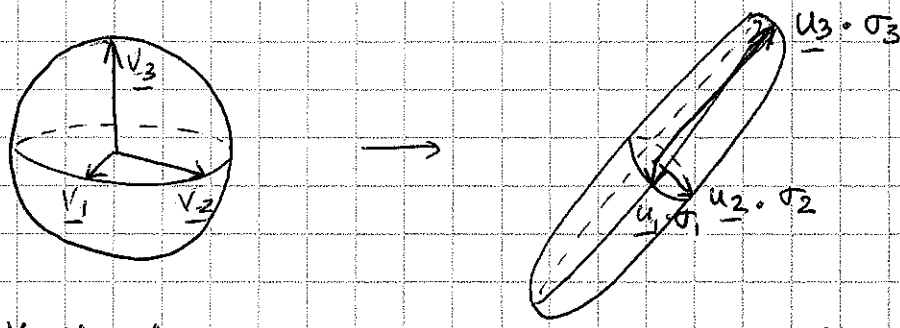
Good preconditioners can reduce the convergence time by orders of magnitude.

(7) Condition number and stability issues

This section is quite general, but will also be used to discuss / introduce the idea of preconditioning.

1. Singular values

Idea: The image of the unit sphere under any $n \times n$ matrix is a hyperellipse



$\underline{v}_1, \underline{v}_2, \underline{v}_3$ = original unit basis

$\underline{u}_1, \underline{u}_2, \underline{u}_3$ = unit vectors along the semi-major axes of the hyperellipse

$\underline{v}_i$ are determined such that

$$\boxed{A\underline{v}_i = \sigma_i \underline{u}_i}$$

$\sigma_1, \sigma_2, \sigma_3$ = "stretching factors" = singular values of A

Note

- If A is not a full-rank matrix, some of the singular values are 0.

  (if A is rank r, exactly r singular values are non-zero).

- By construction $\sigma_i \geq 0$

- By convention, the $\underline{u}_i$ and $\underline{v}_i$ are usually ordered such that

$$\sigma_1 \geq \sigma_2 \geq \sigma_3 \cdots \geq \sigma_n \geq 0$$

- This concept is entirely related to the Singular Value Decomposition (see Midterm project).

2 First notion of ill-conditioning

When applying a matrix A to a vector x, imagine first writing $\underline{x}$ on the basis of the vectors $\underline{v}_k$

$$\underline{x} = \sum \alpha_k \underline{v}_k$$

so $$A\underline{x} = \sum_k \alpha_k \sigma_k \underline{u}_k$$

Even if A is non-singular, problems may arise if $\sigma_1 \gg \sigma_n$. Indeed, in that case the term $\alpha_n \sigma_n \underline{u}_n$ may be negligible in front of $\alpha_1 \sigma_1 \underline{u}_1$ and roundoff errors may affect it.

Geometrically speaking, it is particularly easy to visualize when $\sigma_n \ll \sigma_{n-1} \sim \sigma_{n-2} \cdots \sigma_1$

(the hyperellipse is very flat in one direction)

or $\sigma_1 \gg \sigma_2 \sim \sigma_3 \cdots \sigma_n$

(the hyperellipse is very elongated in one direction)

$\Rightarrow$ in both cases (and any intermediate case $\sigma_1 \sim \sigma_2 \sim \cdots \sigma_k \gg \sigma_{k+1} \sim \cdots \sigma_n$) information is lost if roundoff errors occur.

$\rightarrow$ we expect the sensitivity of a problem to small roundoff errors to be particularly dependent on the value of

$$\kappa = \frac{\sigma_1}{\sigma_n}$$

We now formalize this idea in more mathematical terms.

3. Necessary mathematical tool: the norm of a matrix

We saw that the norm of a vector can be defined in many ways, provided it satisfies

- $\|x\| \geq 0 \quad \forall \underline{x}$ and $\|x\| = 0 \Leftrightarrow \underline{x} = \underline{0}$
- $\|x+y\| \leq \|x\| + \|y\|$
- $\|\alpha x\| = |\alpha| \|x\|$.

A particularly useful norm is the Euclidean norm:

$$\|x\|_2 = \sqrt{x^T x} = \left(\sum_{i=1}^{n} |x_i|^2 \right)^{1/2}$$

(see handout for other norms)

We now define the norm of a matrix as the maximum factor by which a matrix A can stretch a vector $\underline{x}$

$$\|A\| = \sup_{\underline{x}} \frac{\|Ax\|}{\|x\|} = \sup_{\substack{\underline{x} \\ \|x\|=1}} \|Ax\|.$$

(note that this holds even when A is not square)

Now by using the Euclidean norm and writing $x = \sum_k \alpha_k v_k$

it is easy to show that $\boxed{\|A\|_2 = \sigma_1}$

In addition, one can also show that

$$\|A^{-1}\|_2 = \frac{1}{\sigma_n}$$

4. Mathematical definition of conditioning

① **Idea**: Given a mathematical function

$$f: \mathbb{R}^m \to \mathbb{R}^n$$
$$\underline{x} \to f(\underline{x})$$

how do perturbations in the input $\underline{x}$ affect the output $f(\underline{x})$?

- If small "errors" in $\underline{x}$ result only in equivalently small "errors" in $f(\underline{x})$ then the problem is well-conditioned.
- If small "errors" in $\underline{x}$ result in much larger "errors" in $f(\underline{x})$ then the problem is ill-conditioned

Let's define the relative condition number as

$$\kappa(\underline{x}) = \lim_{\delta \to 0} \sup_{\|\delta x\| \leq \delta} \frac{\|\delta f\|/\|f\|}{\|\delta x\|/\|x\|}$$

← relative change in f / rel. change in x

where $\delta f = f(x+\delta x) - f(x)$.

⇒ $\kappa(\underline{x})$ large (for some $\underline{x}$) means that (for that $\underline{x}$), a small change $\underline{\delta x}$ results in a large change $\|\underline{\delta f}\|/\|f\|$

If $f(x)$ is a differentiable function, then

$$\delta f = J(x)\,\delta x$$

↑ Jacobian matrix.

So $$\sup_{\|\delta x\| \leq \delta} \frac{\|\delta f\|}{\|\delta x\|} = \sup \frac{\|J(x)\delta x\|}{\|\delta x\|} = \|J(x)\|$$ by definition

and $$\kappa(x) = \|J(x)\| \cdot \frac{\|x\|}{\|f(x)\|}$$

(2) Conditioning of the problem: what is the error on Ax when A is fixed?

Ax is a linear function so $J \equiv A$

$$\Rightarrow \quad \kappa(x) = \|J(x)\| \cdot \frac{\|x\|}{\|Ax\|}$$

$$= \|A\| \cdot \frac{\|x\|}{\|Ax\|}$$

If we use the 2-norm, $\|x\|_2 = \|A^{-1}Ax\|_2$

(by Cauchy's inequality) $\leq \|A^{-1}\| \|Ax\|_2$

so $\kappa_2(x) \leq \|A\|_2 \|A^{-1}\|_2 = \frac{\sigma_1}{\sigma_m}$

and for some $\underline{x}$ the upper bound is actually achieved.

→ this confirms our expectations that $\frac{\sigma_1}{\sigma_m}$ is a good descriptor of the conditioning of a problem.

The quantity $\|A\|\|A^{-1}\| \equiv \kappa(A)$ is called the condition number of A.

If A is singular, by convention $\kappa(A) = \infty$.

(3) Conditioning of a system of equations w.r.t. perturbations in A.

Let's now suppose we are designing an algorithm working on A which progressively introduces small roundoff errors. How is

$$\underline{x} = A^{-1}b \text{ affected?}$$

→ Similarly, it is possible to show that the condition number is $\kappa(A) = \|A\|\|A^{-1}\|$.

What this implies in practise is that an algorithm working on A introducing relative error $\frac{\|\delta A\|}{\|A\|}$ no larger than $O(\epsilon_{machine})$ in the coefficients result in an error on the outcome

$$\frac{\|\delta x\|}{\|x\|} = O\left(\kappa(A)\,\epsilon_{machine}\right).$$

→ if $\kappa(A)$ is very large, many significant digits in the solution are lost

Note that this assumes roundoff errors themselves are only $O(\epsilon_{machine})$. Some algorithms (cf G-E without pivoting) introduce roundoff errors $\gg \epsilon_{machine}$!

---

5. In practise, how to evaluate $\kappa(A)$?

- Calculating A^{-1}, then $\|A^{-1}\|$ is too CPU-expensive

- See LAPACK routines **CON (specialized) or DGESVX.f (driver routine) for an estimate of the reciprocal of the condition number (returned in RCOND)

- Note that the Lapack routines return either the reciprocal of $\kappa_1(A) = \|A\|_1 \|A^{-1}\|_1$ or $\kappa_\infty(A) = \|A\|_\infty \|A^{-1}\|_\infty$

where $\|A\|_1$ is the norm of A based on $\|x\|_1$ $\left(\|x\|_1 = \sum_i |x_i|\right)$

and

$\|A\|_\infty$ is the norm of A based on $\|x\|_\infty$

$\left(\|x\|_\infty = \max_i |a_i|\right)$

Example

$$A = \begin{pmatrix} 1+10^{-k} & 1 \\ 1 & 1 \end{pmatrix}$$

$$\|A\|_1 = \max_j \sum_i |a_{ij}| = \max\left(2+10^{-k}, 2\right) = 2+10^{-k}$$

$$\|A\|_\infty = \max_i \sum_j |a_{ij}| = 2+10^{-k} \quad \text{(same)}$$

look for A^{-1}:

$$\left(\begin{array}{cc|cc} 1+10^{-k} & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{array}\right)$$

$$\to \left(\begin{array}{cc|cc} 1 & \frac{1}{1+10^{-k}} & \frac{1}{1+10^{-k}} & 0 \\ 0 & \frac{10^{-k}}{1+10^{-k}} & -\frac{1}{1+10^{-k}} & 1 \end{array}\right)$$

$$\to \left(\begin{array}{cc|cc} 1 & 0 & \frac{1+10^{k}}{1+10^{-k}} & -10^{k} \\ 0 & 1 & -\frac{1}{10^{-k}} & \frac{1+10^{-k}}{10^{-k}} \end{array}\right)$$

$$\text{So} \quad A^{-1} = \begin{pmatrix} \frac{1+10^{k}}{1+10^{-k}} & -10^{k} \\ -10^{k} & 1+10^{k} \end{pmatrix}$$

$$\text{So} \quad \|A^{-1}\|_1 \cong 2+10^{k} \qquad \|A^{-1}\|_\infty \simeq 2+10^{k} \quad \Rightarrow \quad \boxed{K(A) \simeq 2\,10^{k} \text{ for large } k}$$

Note that if $b = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ then

$$x = A^{-1}(b) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

if $b = \begin{pmatrix} 1+\epsilon \\ 1 \end{pmatrix}$ then $A^{-1}b \cong \begin{pmatrix} \epsilon\, 10^{k} \\ 1 \end{pmatrix}$

(3) Convergence of conjugate Gradient algorithm

In fact, it can be shown that

$$\|e_n\|_A \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^n \|e_0\|_A$$

(see Trefethen & Bau), p 300.

where $\kappa = \left|\frac{\lambda_{max}}{\lambda_{min}}\right| = \frac{\text{largest evalue}}{\text{smallest evalue}}$.

(note that by assumption A is positive definite so all the evalues are >0)

- For quick convergence, we would hope that κ is "not too large" (ideally we want $\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1} \ll 1$)

in practise typical $\kappa \cong 1 \to 10$ is great!

- For ill-conditioned matrices, the gain of iterative methods vs standard direct methods is negligible

$\Rightarrow$ IDEA OF PRE-CONDITIONING

⑨ Preconditioning ideas

- The convergence rate of an iterative method depends on the condition number $\alpha = \frac{\lambda_{max}}{\lambda_{min}}$ of A.
- For ill-conditioned matrices $\alpha \gg 1$ and the convergence can be v. slow.
- Idea: Instead of solving $Ax=b$, solve the equivalent system

$$KAx = Kb \quad \text{where}$$

K = a non singular matrix
KA = a better-conditioned matrix.

- Note: Although the product KA is never actually formed in a preconditioned algorithm, it is vital that the product Kv or $K^T v$ or $K^{-1} v$ be fast for any vector v ($a < n^2$ process)
  ↳ K should be a simple or v. sparse matrix.
- Finding preconditioners that satisfy both requirements is ongoing research
  - different types of matrices A respond better to different types of preconditioners
  - Preconditioning can be mathematically based or physically base

Mathematical	Physical
Diagonal	Coarse grid / Multigrid
Incomplete Cholesky (Golub & V. Loan)	low-pass filters

See Appendix on preconditioning for actual implementation