Conditioning

- Absolute Condition Number:

- (Relative) Condition Number:
Posedness and Conditioning

- What is the condition number of an ill-posed problem?
Matrix Condition Number

- The matrix condition number $\kappa(A)$ is the ratio between the max and min distance from the surface to the center of the unit ball (norm-1 vectors) transformed by $A$:

- The matrix condition number bounds the worst-case amplification of error in a matrix-vector product:
Singular Value Decomposition

- The singular value decomposition (SVD)

- Condition number in terms of singular values
Visualization of Matrix Conditioning

\[ \{ x : x \in \mathbb{R}^2, \| x \|_2 = 1 \} \xrightarrow{A} \{ Ax : x \in \mathbb{R}^2, \| x \|_2 = 1 \} \]

\[ A = U \begin{bmatrix} \sigma_{\text{max}} & 0 \\ 0 & \sigma_{\text{min}} \end{bmatrix} V^T \]

\[ \kappa(A) = \frac{\sigma_{\text{max}}}{\sigma_{\text{min}}} \]

\[ 1/\| A^{-1} \|_2 = \sigma_{\text{min}} \]

\[ \| A \|_2 = \sigma_{\text{max}} \]
Linear Least Squares

- Find $x^* = \arg\min_{x \in \mathbb{R}^n} \|Ax - b\|_2$ where $A \in \mathbb{R}^{m \times n}$:

- Given the SVD $A = U\Sigma V^T$ we have $x^* = V\Sigma^\dagger U^T b$, where $\Sigma^\dagger$ contains the reciprocal of all nonzeros in $\Sigma$, and more generally $\dagger$ denotes pseudoinverse:
Normal Equations

- *Normal equations* are given by solving $A^T A x = A^T b$:

- However, solving the normal equations is a more ill-conditioned problem than the original least squares algorithm.
Solving the Normal Equations

- If $A$ is full-rank, then $A^T A$ is symmetric positive definite (SPD):

- Since $A^T A$ is SPD we can use Cholesky factorization, to factorize it and solve linear systems:
QR Factorization

- If $A$ is full-rank there exists an orthogonal matrix $Q$ and a unique upper-triangular matrix $R$ with a positive diagonal such that $A = QR$

- A reduced QR factorization (unique part of general QR) is defined so that $Q \in \mathbb{R}^{m \times n}$ has orthonormal columns and $R$ is square and upper-triangular

- We can solve the normal equations (and consequently the linear least squares problem) via reduced QR as follows
Computing the QR Factorization

- The Cholesky-QR algorithm uses the normal equations to obtain the QR factorization

- Orthogonalization-based methods are most efficient and stable for QR factorization of dense matrices
Eigenvalue Decomposition

- If a matrix $A$ is diagonalizable, it has an *eigenvalue decomposition*

- $A$ and $B$ are *similar*, if there exist $Z$ such that $A = ZBZ^{-1}$
## Similarity of Matrices

<table>
<thead>
<tr>
<th>matrix</th>
<th>similarity</th>
<th>reduced form</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>real symmetric</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hermitian</td>
<td></td>
<td></td>
</tr>
<tr>
<td>normal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>real</td>
<td></td>
<td></td>
</tr>
<tr>
<td>diagonalizable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>arbitrary</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Rayleigh Quotient

- For any vector $x$ that is close to an eigenvector, the Rayleigh quotient provides an estimate of the associated eigenvalue of $A$: 
Introduction to Krylov Subspace Methods

- **Krylov subspace methods** work with information contained in the $n \times k$ matrix

$$K_k = \begin{bmatrix} x_0 & Ax_0 & \cdots & A^{k-1}x_0 \end{bmatrix}$$

span \( \{k_k\} \)

\[
\begin{aligned}
\min_{x \in \text{span}(k_k)} \left\{ \frac{x^T A x}{x^T x}, \| A x - b \|_2 \right\}
\end{aligned}
\]

- $A$ is similar to **companion matrix** $C = K_n^{-1} A K_n$. 
Krylov Subspaces

- Given \( Q_k R_k = K_k \), we obtain an orthonormal basis for the Krylov subspace,

\[ K_k(A, x_0) = \text{span}(Q_k) = \{ p(A)x_0 : \text{deg}(p) < k \}, \]

where \( p \) is any polynomial of degree less than \( k \).

- The Krylov subspace includes the \( k - 1 \) approximate dominant eigenvectors generated by \( k - 1 \) steps of power iteration:

\[
\begin{align*}
\text{Power Iteration} \\
x^{(k+1)} &= A x^{(k-1)} \\
\text{compute } \rho &= \frac{x^{(k+1)} \cdot A x^{(k-1)}}{x^{(k-1)} \cdot x^{(k-1)}} \\
x^{(k)} &= x^{(k+1)} / \rho
\end{align*}
\]
The $k \times k$ matrix $H_k = Q_k^T A Q_k$ minimizes $\|A Q_k - Q_k H_k\|_2$:

$A X = X \Lambda$

$H_k$ is upper-Hessenberg, because the companion matrix $C_n$ is upper-Hessenberg:

if $A$ is symmetric
Rayleigh-Ritz Procedure

- The eigenvalues/eigenvectors of $H_k$ are the *Ritz values/vectors*:

- The Ritz vectors and values are the *ideal approximations* of the actual eigenvalues and eigenvectors based on only $H_k$ and $Q_k$: 
Low Rank Matrix Approximation

- Given a matrix $A \in \mathbb{R}^{m \times n}$ seek rank $r < m, n$ approximation

$$\min_{\tilde{A}} \left\| \tilde{A} - A \right\|_{F}$$

$$\text{rank}(\tilde{A}) = r$$

- Eckart-Young (optimal low-rank approximation by SVD) theorem

$$A = \begin{bmatrix} \sigma_1 \cdots \sigma_r \end{bmatrix} \begin{bmatrix} u_1 & \cdots & u_r \end{bmatrix}^{T}$$
Rank Revealing Matrix Factorizations

- Computing the SVD

\[
A^TA \rightarrow \text{eigens of } A^TA = \text{eigens of } A
\]

\[
A = U \Sigma V^T \rightarrow U^T \Sigma^2 V
\]

- QR with column pivoting
QR with column pivoting

\[
\begin{bmatrix}
\end{bmatrix} \Rightarrow 
\begin{bmatrix}
\end{bmatrix}
\]

after \( k \) steps

\[
\begin{bmatrix}
\end{bmatrix}
\]

\( \mathbf{Q}_1 \ldots \mathbf{Q}_n \)
Orthogonal Iteration

- For sparse matrices, QR factorization creates fill, so must revert to iterative methods

\[ R = \text{L factor of } A^T A \]

\[ R_k = A^T Q_k \]

Orthogonal iteration interleaves deflation and power iteration
Randomized SVD

- Orthogonal iteration for SVD can also be viewed as a randomized algorithm

\[ QR = col-piv \quad O(mnr) \]

- Pick \( Q_0 \) to be random and orthogonal

\[ AQ_0 \] if \( A \) is low rank (rank \( r \))

\[ U \text{sv}^+ Q \]

\[ Q_0 = \text{span}(V) \] where \( V \) is a set of random normalized linear combinations of \( \text{cols} \) of \( US \)

\[ QR = QR(AQ_0) \]

\[ \text{span}(Q_i) = \text{span}(U_i) \quad i = 0, \ldots, r, \sigma \]
Generalized Nyström Algorithm

- The generalized Nyström algorithm provides an efficient way of computing a sketched low-rank factorization

\[ \tilde{A} = A S_1 (S_2^T A S_1)^+ S_2^T A \]

where \( S_1 \) and \( S_2 \) are sketching matrices

- Gaussian random
- FFT sketch SRFT
- leverage-score sampling
Minimize $f(x)$

Quadratic optimization $f(x) = \frac{1}{2} x^T A x - b^T x$
Basic Multidimensional Optimization Methods

- Steepest descent: minimize $f$ in the direction of the negative gradient:

- Given quadratic optimization problem $f(x) = \frac{1}{2}x^TAx + b^Tx$ where $A$ is symmetric positive definite, the error $e_k = x_k - x^*$ satisfies

$$\|e_{k+1}\|_A =$$

- When sufficiently close to a local minima, general nonlinear optimization problems are described by such an SPD quadratic problem.

- Convergence rate depends on the conditioning of $A$, since
Gradient Methods with Extrapolation

- We can improve the constant in the linear rate of convergence of steepest descent by leveraging *extrapolation methods*, which consider two previous iterates (maintain *momentum* in the direction $x_k - x_{k-1}$):

- The *heavy ball method*, which uses constant $\alpha_k = \alpha$ and $\beta_k = \beta$, achieves better convergence than steepest descent:
Conjugate Gradient Method

- The *conjugate gradient method* is capable of making the optimal (for a quadratic objective) choice of $\alpha_k$ and $\beta_k$ at each iteration of an extrapolation method:

- *Parallel tangents* implementation of the method proceeds as follows
Krylov Optimization

- Conjugate gradient (CG) finds the minimizer of $f(x) = \frac{1}{2} x^T A x - b^T x$ (which satisfies optimality condition $A x = b$) within the Krylov subspace of $A$: 
The solution at the $k$th step, $y_k = \|b\|_2 T_k^{-1} e_1$ is obtained by CG from $y_{k+1}$ with a single matrix-vector product with $A$ and vector operations with $O(n)$ cost.
Preconditioning

Convergence of iterative methods for $Ax = b$ depends on $\kappa(A)$, the goal of a preconditioner $M$ is to obtain $x$ by solving

$$M^{-1}Ax = M^{-1}b$$

with $\kappa(M^{-1}A) < \kappa(A)$

Common preconditioners select parts of $A$ or perform inexact factorization
Conjugate Gradient Convergence Analysis

- In previous discussion, we assumed $K_n$ is invertible, which may not be the case if $A$ has $m < n$ distinct eigenvalues, however, in exact arithmetic CG converges in $m - 1$ iterations\(^1\)

\(^1\)This derivation follows *Applied Numerical Linear Algebra* by James Demmel, Section 6.6.4
Conjugate Gradient Convergence Analysis (II)

- Using \( z = \rho_{k-1}(A)A \mathbf{x} \), we can simplify \( \phi(z) = (\mathbf{x} - z)^T A(\mathbf{x} - z) \) as

- We can bound the objective based on the eigenvalues of \( A = Q \Lambda Q^T \) using the identity \( p(A) = Q p(\Lambda) Q^T \),
Conjugate Gradient Convergence Analysis (III)

- Using our bound on the square of the residual norm $\phi(z)$, we can see why CG converges after $m - 1$ iterations if there are only $m < n$ distinct eigenvalues.

- To see that the residual goes to 0, we find a suitable polynomial in $Q_m$ (the set of polynomials $q_m$ of degree $m$ with $q_m(0) = 1$).
Round-off Error in Conjugate Gradient

- CG provides strong convergence guarantees for SPD matrices in exact arithmetic

- Due to round-off CG may stagnate / have plateaus in convergence
Graph and Matrix Duality

- graphs have a natural correspondence with sparse matrices

- matrix-based representations of graphs can be used to devise algorithms
Graph Partitioning from Eigenvectors

- The Laplacian matrix provides a model of interactions on a graph that is useful in many contexts.

- The second-smallest-eigenvalue eigenvector of the Laplacian (the Fiedler vector), gives a good partitioning of the graph.
Newton’s Method

- Newton’s method in $n$ dimensions is given by finding minima of $n$-dimensional quadratic approximation using the gradient and Hessian of $f$: 
Nonlinear Least Squares

- An important special case of multidimensional optimization is nonlinear least squares, the problem of fitting a nonlinear function \( f_x(t) \) so that \( f_x(t_i) \approx y_i \):

- We can cast nonlinear least squares as an optimization problem to minimize residual error and solve it by Newton’s method:
Gauss-Newton Method

- The Hessian for nonlinear least squares problems has the form:

- The *Gauss-Newton* method is Newton iteration with an approximate Hessian:
Constrained Optimization Problems

- We now return to the general case of \textit{constrained} optimization problems:

\[
\min_x f(x) \quad \text{subject to} \quad g(x) = 0 \quad \text{and} \quad h(x) \leq 0
\]

- Generally, we will seek to reduce constrained optimization problems to a series of simpler optimization problems: