Thomas Finley, `tomf@cs.cornell.edu`

Linear Algebra

A *subspace* is a set $S \subseteq \mathbb{R}^n$ such that $\mathbf{0} \in S$ and $\forall \mathbf{x}, \mathbf{y} \in S, \alpha, \beta \in \mathbb{R} \,.\, \alpha\mathbf{x} + \beta\mathbf{y} \in S$.

$\mathbf{x} \in \mathbb{R}^n$ is a *linear combination* of $\mathbf{v}_1, \cdots, \mathbf{v}_k$ if $\exists \beta_1, \cdots, \beta_k \in \mathbb{R}$ such that $\mathbf{x} = \beta_1\mathbf{v}_1 + \cdots + \beta_k\mathbf{v}_k$.

The *span* of $\{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ is the set of all vectors in $\mathbb{R}^n$ that are linear combinations of $\mathbf{v}_1, \ldots, \mathbf{v}_k$.

A *basis* B of subspace S, $B = \{\mathbf{v}_1, \ldots, \mathbf{v}_k\} \subset S$ has $Span(B) = S$ and all $\mathbf{v}_i$ linearly independent.

The *dimension* of S is $|B|$ for a basis B of S.

For subspaces S, T with $S \subseteq T$, $dim(S) \leq dim(T)$, and further if $dim(S) = dim(T)$, then $S = T$.

A *linear transformation* $T : \mathbb{R}^n \to \mathbb{R}^m$ has $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n, \alpha, \beta \in \mathbb{R} \,.\, T(\alpha\mathbf{x} + \beta\mathbf{y}) = \alpha T(\mathbf{x}) + \beta T(\mathbf{y})$. Further, $\exists A \in \mathbb{R}^{m\times n}$ such that $\forall \mathbf{x} \,.\, T(\mathbf{x}) \equiv A\mathbf{x}$.

For two linear transformations $T : \mathbb{R}^n \to \mathbb{R}^m$, $S : \mathbb{R}^m \to \mathbb{R}^p$, $S \circ T \equiv S(T(\mathbf{x}))$ is linear transformation. $(T(\mathbf{x}) \equiv A\mathbf{x}) \wedge (S(\mathbf{y}) \equiv B\mathbf{y}) \Rightarrow (S \circ T)(\mathbf{x}) \equiv BA\mathbf{x}$.

The matrix's *row space* is the span of its rows, its *column space* or *range* is the span of its columns, and its *rank* is the dimension of either of these spaces.

For $A \in \mathbb{R}^{m\times n}$, $rank(A) \leq \min(m, n)$. A has *full row* (or *column*) *rank* if $rank(A) = m$ (or n).

A *diagonal matrix* $D \in \mathbb{R}^{n\times n}$ has $d_{j,k} = 0$ for $j \neq k$. The diagonal *identity matrix* I has $i_{j,j} = 1$.

The *upper* (or *lower*) *bandwidth* of A is $\max|i - j|$ among i, j where $i \geq j$ (or $i \leq j$) such that $A_{i,j} \neq 0$.

A matrix with lower bandwidth 1 is *upper Hessenberg*.

For $A, B \in \mathbb{R}^{n\times n}$, B is A's *inverse* if $AB = BA = I$. If such a B exists, A is *invertible* or *nonsingular*. $B = A^{-1}$.

The inverse of A is $A^{-1} = [\mathbf{x}_1, \cdots, \mathbf{x}_n]$ where $A\mathbf{x}_i = \mathbf{e}_i$.

For $A \in \mathbb{R}^{n\times n}$ the following are equivalent: A is nonsingular, $rank(A) = n$, $A\mathbf{x} = \mathbf{b}$ is solvable for any $\mathbf{b}$, $A\mathbf{x} = \mathbf{0}$ iff $\mathbf{x} = \mathbf{0}$.

The *inner product* of $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ is $\mathbf{x}^T\mathbf{y} = \sum_{i=1}^n x_i y_i$.

Vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ are *orthogonal* if $\mathbf{x}^T\mathbf{y} = 0$.

The *nullspace* or *kernel* of $A \in \mathbb{R}^{m\times n}$ is $\{\mathbf{x} \in \mathbb{R}^n \;:\; A\mathbf{x} = \mathbf{0}\}$.

For $A \in \mathbb{R}^{m\times n}$, $Range(A)$ and $Nullspace(A^T)$ are *orthogonal complements*, i.e., $\mathbf{x} \in Range(A), \mathbf{y} \in Nullspace(A^T) \Rightarrow \mathbf{x}^T\mathbf{y} = 0$. For all $\mathbf{p} \in \mathbb{R}^m$, there exist unique $\mathbf{x}$ and $\mathbf{y}$ so that $\mathbf{p} = \mathbf{x} + \mathbf{y}$.

For a *permutation matrix* $P \in \mathbb{R}^{n\times n}$, PA permutes the rows of A, AP the columns of A. $P^{-1} = P^T$.

Gaussian Elimination

GE produces a factorization $A = LU$, GEPP $PA = LU$.

Plain GE

```
for $k = 1$ to $n - 1$ do
    if $a_{kk} = 0$ then stop
    $\ell_{k+1:n,k} = a_{k+1:n,k}/a_{kk}$
    $a_{k+1:n,k:n} = a_{k+1:n,k:n} - \ell_{k+1:n,k}a_{k,k:n}$
end for
```

GE with Partial Pivoting

```
for $k = 1$ to $n - 1$ do
    $\gamma = \operatorname{argmax}_{i\in\{k+1,\ldots,n\}} |a_{ik}|$
    $a_{[\gamma,k],k:n} = a_{[k,\gamma],k:n}$
    $\ell_{[\gamma,k],1:k-1} = \ell_{[k,\gamma],1:k-1}$
    $p_k = \gamma$
    $\ell_{k:n,k} = a_{k:n,k}/a_{kk}$
    $a_{k+1:n,k:n} = a_{k+1:n,k:n} - \ell_{k+1:n,k}a_{k,k:n}$
end for
```

Backward Substitution

```
$\mathbf{x} = zeros(n, 1)$
for $j = n$ to $1$ do
    $x_j = \frac{w_j - u_{j,j+1:n}x_{j+1:n}}{u_{j,j}}$
end for
```

To solve $A\mathbf{x} = \mathbf{b}$, factor $A = LU$ (or $A = P^TLU$), solve $L\mathbf{w} = \mathbf{b}$ (or $L\mathbf{w} = \hat{\mathbf{b}}$ where $\hat{\mathbf{b}} = P\mathbf{b}$) for $\mathbf{w}$ using forward substitution, then solve $U\mathbf{x} = \mathbf{w}$ for $\mathbf{x}$ using backward substitution. The complexity of GE and GEPP is $\frac{2}{3}n^3 + O(n^2)$. GEPP encounters an exact 0 pivot iff A is singular.

For banded A, $L + U$ has the same bandwidths as A.

Norms

A *vector norm* function $\|\cdot\| : \mathbb{R}^n \to \mathbb{R}$ satisfies:

1. $\|\mathbf{x}\| \geq 0$, and $\|\mathbf{x}\| = 0 \Leftrightarrow \mathbf{x} = \vec{0}$.
2. $\|\gamma\mathbf{x}\| = |\gamma| \cdot \|\mathbf{x}\|$ for all $\gamma \in \mathbb{R}$, and all $\mathbf{x} \in \mathbb{R}^n$.
3. $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$, for all $x, y \in \mathbb{R}^n$.

Common norms include:

1. $\|\mathbf{x}\|_1 = |x_1| + |x_2| + \cdots + |x_n|$
2. $\|\mathbf{x}\|_2 = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$
3. $\|\mathbf{x}\|_\infty = \lim_{p\to\infty} (|x_1|^p + \cdots + |x_n|^p)^{\frac{1}{p}} = \max_{i=1..n} |x_i|$

An *induced matrix norm* is $\|A\|_\square = \sup_{\mathbf{x}\neq 0} \frac{\|A\mathbf{x}\|_\square}{\|\mathbf{x}\|_\square}$. It satisfies the three properties of norms.

$\forall \mathbf{x} \in \mathbb{R}^n, A \in \mathbb{R}^{m\times n}$, $\|A\mathbf{x}\|_\square \leq \|A\|_\square\|\mathbf{x}\|_\square$.

$\|AB\|_\square \leq \|A\|_\square\|B\|_\square$, called *submultiplicativity.*

$\mathbf{a}^T\mathbf{b} \leq \|\mathbf{a}\|_2\|\mathbf{b}\|_2$, called *Cauchy-Schwarz inequality.*

1. $\|A\|_\infty = \max_{i=1,\ldots,m} \sum_{j=1}^n |a_{i,j}|$ (max row sum).
2. $\|A\|_1 = \max_{j=1,\ldots,n} \sum_{i=1}^m |a_{i,j}|$ (max column sum).
3. $\|A\|_2$ is hard: it takes $O(n^3)$, not $O(n^2)$ operations.
4. $\|A\|_F = \sqrt{\sum_{i=1}^n \sum_{j=1}^m a_{i,j}^2}$. $\|\cdot\|_F$ often replaces $\|\cdot\|_2$.

Numerical Stability

Six sources of error in scientific computing: modeling errors, measurement or data errors, blunders, discretization or truncation errors, convergence tolerance, and rounding errors.

$$\underbrace{\pm}_{\text{sign}}\ \underbrace{d_1.d_2d_3\cdots d_t}_{\text{mantissa}} \times \underbrace{\beta}_{\text{base}}{}^{\overbrace{e}^{\text{exponent}}}$$

For single and double:

$t = 24$, $e \in \{-126, \ldots, 127\}$

$t = 53$, $e \in \{-1022, \ldots, 1023\}$

The *relative error* in $\hat{\mathbf{x}}$ approximating $\mathbf{x}$ is $\frac{|\hat{\mathbf{x}} - \mathbf{x}|}{|\mathbf{x}|}$.

Unit roundoff or *machine epsilon* is $\epsilon_{mach} = \beta^{-t+1}$. Arithmetic operations have relative error bounded by ϵ_{mach}.

E.g., consider $z = x - y$ with input x, y. This program has three roundoff errors. $\hat{z} = ((1 + \delta_1)x - (1 + \delta_2)y)(1 + \delta_3)$, where $\delta_1, \delta_2, \delta_3 \in [-\epsilon_{mach}, \epsilon_{mach}]$. $\frac{|z - \hat{z}|}{|z|} = \frac{|(\delta_1+\delta_3)x - (\delta_2+\delta_3)y + O(\epsilon_{mach}^2)|}{|x-y|}$ The bad case is where $\delta_1 = \epsilon_{mach}$, $\delta_2 = -\epsilon_{mach}$, $\delta_3 = 0$: $\frac{|z - \hat{z}|}{|z|} = \epsilon_{mach}\frac{|x+y|}{|x-y|}$ Inaccuracy if $|x + y| \gg |x - y|$ called *catastrophic calcellation.*

Conditioning & Backwards Stability

A problem instance is *ill conditioned* if the solution is sensitive to perturbations of the data. For example, $\sin 1$ is well conditioned, but $\sin 12392193$ is ill conditioned.

Suppose we perturb $A\mathbf{x} = \mathbf{b}$ by $(A + E)\hat{\mathbf{x}} = \mathbf{b} + \mathbf{e}$ where $\frac{\|E\|}{\|A\|} \leq \delta, \frac{\|\mathbf{e}\|}{\|\mathbf{b}\|} \leq \delta$. Then $\frac{\|\hat{\mathbf{x}} + \mathbf{x}\|}{\|\mathbf{x}\|} \leq 2\delta\kappa(A) + O(\delta^2)$, where $\kappa(A) = \|A\|\|A^{-1}\|$ is the *condition number* of A.

1. $\forall A \in \mathbb{R}^{n\times n}$, $\kappa(A) \geq 1$.
2. $\kappa(I) = 1$.
3. If $\gamma \neq 0$, $\kappa(\gamma A) = \kappa(A)$.
4. For diagonal D and all p, $\|D\|_p = \max_{i=1..n} |d_{ii}|$. So, $\kappa(D) = \frac{\max_{i=1..n} |d_{ii}|}{\min_{i=1..n} |d_{ii}|}$.

If $\kappa(A) \geq \frac{1}{\epsilon_{mach}}$, A may as well be singular.

An algorithm is *backwards stable* if in the presence of roundoff error it returns the exact solution to a nearby problem instance.

GEPP solves $A\mathbf{x} = \mathbf{b}$ by returning $\hat{\mathbf{x}}$ where $(A + E)\hat{\mathbf{x}} = \mathbf{b}$. It is backwards stable if $\frac{\|E\|_\infty}{\|A\|_\infty} \leq O(\epsilon_{mach})$. With GEPP, $\frac{\|E\|_\infty}{\|A\|_\infty} \leq c_n\epsilon_{mach} + O(\epsilon_{mach}^2)$, where c_n is worst case exponential in n, but in practice almost always low order polynomial.

Combining stability and conditioning analysis yields $\frac{\|\hat{\mathbf{x}} - \mathbf{x}\|}{\|\mathbf{x}\|} \leq c_n \cdot \kappa(A)\epsilon_{mach} + O(\epsilon_{mach}^2)$.

Determinant

The *determinant* $\det : \mathbb{R}^{n\times n} \to \mathbb{R}$ satisfies:

1. $\det(AB) = \det(A)\det(B)$.
2. $\det(A) = 0$ iff A singular.
3. $\det(A) = \det(A^T)$.
4. $\det(L) = \ell_{1,1}\ell_{2,2}\cdots\ell_{n,n}$ for triangular L.

To compute $\det(A)$ factor $A = P^TLU$. $\det(P) = (-1)^s$ where P performs s swaps, $\det(L) = 1$. When calculating $\det(U)$, beware of overflow!

Orthogonal Matrices

For $Q \in \mathbb{R}^{n\times n}$, these statements are equivalent:

1. $Q^TQ = QQ^T = I$ (i.e., Q is *orthogonal*)
2. The $\|\cdot\|_2 = 1$ for each row and column of Q. The inner product of any row (or column) with another is 0.
3. For all $\mathbf{x} \in \mathbb{R}^n$, $\|Q\mathbf{x}\|_2 = \|\mathbf{x}\|_2$.

A matrix $Q \in \mathbb{R}^{m\times n}$ with $m > n$ has *orthonormal columns* if the columns are orthonormal, and $Q^TQ = I$. The product of orthogonal matrices is orthogonal. For orthogonal Q, $\|QA\|_2 = \|A\|_2$ and $\|AQ\|_2 = \|A\|_2$.

Positive Definite, $A = LDL^T$

$A \in \mathbb{R}^{n\times n}$ is *positive definite* (PD) (or *semidefinite* (PSD)) if $\mathbf{x}^TA\mathbf{x} > 0$ (or $\mathbf{x}^TA\mathbf{x} \geq 0$).

When LU-factorizing symmetric A, the result is $A = LDL^T$; L is unit lower triangular, D is diagonal. A is SPD iff D has all positive entries. The *Cholesky factorization* is $A = LDL^T = LD^{1/2}D^{1/2}L^T = GG^T$. Can be done directly in $\frac{n^3}{3} + O(n^2)$ flops. If G's diagonal is positive, A is SPD.

To solve $A\mathbf{x} = \mathbf{b}$ for SPD A, factor $A = GG^T$, solve $G\mathbf{w} = \mathbf{b}$ by forward substitution, then solve $G^T\mathbf{x} = \mathbf{w}$ with backwards substitution, which takes $\frac{n^3}{3} + O(n^2)$ flops.

For $A \in \mathbb{R}^{m\times n}$, if $rank(A) = n$, then A^TA is SPD.

QR-factorization

For any $A \in \mathbb{R}^{m\times n}$ with $m \geq n$, we can factor $A = QR$, where $Q \in \mathbb{R}^{m\times m}$ is orthogonal, and $R = [\; R_1 \quad 0 \;]^T \in \mathbb{R}^{m\times n}$ is upper triangular. $rank(A) = n$ iff R_1 is invertible.

Q's first n (or last $m - n$) columns form an orthonormal basis for $span(A)$ (or $nullspace(A^T)$).

A *Householder reflection* is $H = I - \frac{2\mathbf{v}\mathbf{v}^T}{\mathbf{v}^T\mathbf{v}}$. H is symmetric and orthogonal. Explicit H.H. QR-factorization is:

```
for $k = 1 : n$ do
    $\mathbf{v} = A(k : m, k) \pm \|A(k : m, k)\|_2\mathbf{e}_1$
    $A(k : m, k : n) = \left(I - \frac{2\mathbf{v}\mathbf{v}^T}{\mathbf{v}^T\mathbf{v}}\right) A(k : m, k : n)$
end for
```

We get $H_nH_{n-1}\cdots H_1A = R$, so then, $Q = H_1H_2\cdots H_n$. This takes $2mn^2 - \frac{2}{3}n^3 + O(mn)$ flops.

Givens requires 50% more flops. Preferable for sparse A.

The Gram-Schmidt produces a *skinny/reduced* QR-factorization $A = Q_1R_1$, where $Q_1 \in \mathbb{R}^{m\times n}$ has orthonormal columns. The *Gram-Schmidt* algorithm is:

Left Looking

```
1: for $k = 1 : n$ do
2:     $\mathbf{q}_k = \mathbf{a}_k$
3:     for $j = 1 : k - 1$ do
4:         $R(j, k) = \mathbf{q}_j^T\mathbf{a}_k$
5:         $\mathbf{q}_k = \mathbf{q}_k - R(j, k)\mathbf{q}_j$
6:     end for
7:     $R(k, k) = \|\mathbf{q}_k\|_2$
8:     $\mathbf{q}_k = \mathbf{q}_k/R(k, k)$
9: end for
```

Right Looking

```
1: $Q = A$
2: for $k = 1 : n$ do
3:     $R(k, k) = \|\mathbf{q}_k\|_2$
4:     $\mathbf{q}_k = \mathbf{q}_k/R(k, k)$
5:     for $j = k + 1 : n$ do
6:         $R(k, j) = \mathbf{q}_k^T\mathbf{q}_j$
7:         $\mathbf{q}_j = \mathbf{q}_j - R(k, j)\mathbf{q}_k$
8:     end for
9: end for
```

In left looking, let line 4 be $R(j, k) = \mathbf{q}_j^T\mathbf{q}_k$ for modified G.S. to make it backwards stable.

Basic Linear Algebra Subroutines

0. Scalar ops, like $\sqrt{x^2 + y^2}$. $O(1)$ flops, $O(1)$ data.
1. Vector ops, like $\mathbf{y} = a\mathbf{x} + \mathbf{y}$. $O(n)$ flops, $O(n)$ data.
2. Matrix-vector ops, like rank-one update $A = A + \mathbf{x}\mathbf{y}^T$. $O(n^2)$ flops, $O(n^2)$ data.
3. Matrix-matrix ops, like $C = C + AB$. $O(n^3)$ flops, $O(n^2)$ data.

Use the highest BLAS level possible. Operators are architecture tuned, e.g., data processed in cache-sized bites.

Linear Least Squares

Suppose we have points $(u_1, v_1), \ldots, (u_5, v_5)$ that we want to fit a quadratic curve $au^2 + bu + c$ through. We want to solve for:

$$\begin{bmatrix} u_1^2 & u_1 & 1 \\ \vdots & \vdots & \vdots \\ u_5^2 & u_5 & 1 \end{bmatrix}\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} v_1 \\ \vdots \\ v_5 \end{bmatrix}$$

This is *overdetermined* so an exact solution is out. Instead, find the *least squares* solution $\mathbf{x}$ that minimizes $\|A\mathbf{x} - \mathbf{b}\|_2$.

For the *method of normal equations*, solve for $\mathbf{x}$ in $A^TA\mathbf{x} = A^T\mathbf{b}$ with Cholesky factorization. This takes $mn^2 + \frac{n^3}{3} + O(mn)$ flops. It is conditionally but not backwards stable: A^TA doubles the condition number.

Alternatively, factor $A = QR$. Let $\mathbf{c} = [\; \mathbf{c}_1 \quad \mathbf{c}_2 \;]^T = Q^T\mathbf{b}$. The least squares solution is $\mathbf{x} = R_1^{-1}\mathbf{c}_1$.

If $rank(A) = r$ and $r < n$ (rank deficient), factor $A = U\Sigma V^T$, let $y = V^Tx$ and $c = U^Tb$. Then, $\min\|A\mathbf{x} - \mathbf{b}\|_2 = \min\sqrt{\sum_{i=1}^r(\sigma_iy_i - c_i)^2 + \sum_{i=r+1}^m c_i^2}$, so $y_i = \frac{c_i}{\sigma_i}$. For $i = r + 1 : n$, y_i is arbitrary.

Singular Value Decomposition

For any $A \in \mathbb{R}^{m\times n}$, we can express $A = U\Sigma V^T$ such that $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$ are orthogonal, and $\Sigma = diag(\sigma_1, \cdots, \sigma_p) \in \mathbb{R}^{m\times n}$ where $p = \min(m, n)$ and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p \geq 0$. The σ_i are singular values.

1. Matrix 2-norm, where $\|A\|_2 = \sigma_1$.
2. The condition number $\kappa_2(A) = \|A\|_2\|A^{-1}\|_2 = \frac{\sigma_1}{\sigma_n}$, or rectangular condition number $\kappa_2(A) = \frac{\sigma_1}{\sigma_{\min(m,n)}}$. Note that $\kappa_2(A^TA) = \kappa_2(A)^2$.
3. For a rank k approximation to A, let $\Sigma_k = diag(\sigma_1, \cdots, \sigma_k, \mathbf{0}^T)$. Then $A_k = U\Sigma_kV^T$. $rank(A_k) \leq k$ and $rank(A_k) = k$ iff $\sigma_k > 0$. Among rank k or lower matrices, A_k minimizes $\|A - A_k\|_2 = \sigma_{k+1}$.
4. Rank determination, since $rank(A) = r$ equals the number of nonzero σ, or in machine arithmetic, perhaps the number of $\sigma \geq \epsilon_{mach} \times \sigma_1$.

$$A = U\Sigma V^T = \begin{bmatrix} U_1 & U_2 \end{bmatrix}\begin{bmatrix} \Sigma(1 : r, 1 : r) & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix}$$

See that $range(U_1) = range(A)$. The SVD gives an orthonormal basis for the range and nullspace of A and A^T.

Compute the SVD by using shifted QR on A^TA.

Information Retrival & LSI

In the *bag of words* model, $\mathbf{w}_d \in \mathbb{R}^m$, where $\mathbf{w}_d(i)$ is the (perhaps weighted) frequency of term i in document d. The *corpus* matrix is $A = [\mathbf{w}_1, \cdots, \mathbf{w}_n] \in \mathbb{R}^{m\times n}$. For a query $\mathbf{q} \in \mathbb{R}^m$, rank documents according to a $\frac{\mathbf{q}^T\mathbf{w}_d}{\|\mathbf{w}_d\|_2}$ score.

In *latent semantic indexing*, you do the same, but in a k dimensional subspace. Factor $A = U\Sigma V^T$, then define $A^* = \Sigma_{1:k,1:k}V_{:,1:k}^T \in \mathbb{R}^{k\times n}$. Each $\mathbf{w}_d^* = A_{:,d}^* = U_{:,1:k}^T\mathbf{w}_d$, and $\mathbf{q}^* = U_{:,1:k}^T\mathbf{q}$.

In the Ando-Lee analysis, for a corpus with k topics, for $t \in 1 : k$ and $d \in 1 : n$, let $R_{t,d} \geq 0$ be document d's relevance to topic t. $\|R_{:,d}\|_2 = 1$. *True document similarity* is $RR^T = \mathbb{R}^{n\times n}$, where entry (i, j) is relevance of i to j. Using LSI, if A contains information about RR^T, then $(A^*)^TA^*$ will approximate RR^T well. LSI depends on even distribution of topics, where distribution is $\rho = \frac{\max_t \|R_{t,:}\|_2}{\min_t \|R_{t,:}\|_2}$. Great for ρ is near 1, but if $\rho \gg 1$, LSI does worse.

Complex Numbers

Complex numbers are written $z = x + iy \in \mathbb{C}$ for $i = \sqrt{-1}$. The *real part* is $x = \Re(z)$. The *imaginary part* is $y = \Re(z)$.

The *conjugate* of z is $\overline{z} = x - iy$. $\overline{A}\overline{\mathbf{x}} = \overline{(A\mathbf{x})}$, $\overline{A}\,\overline{B} = \overline{(AB)}$

The *absolute value* of z is $|z| = \sqrt{x^2 + y^2}$.

The *conjugate transpose* of $\mathbf{x}$ is $\mathbf{x}^H = (\overline{\mathbf{x}})^T$. $A \in \mathbb{C}^{n\times n}$ is *Hermitian* or *self-adjoint* if $A = A^H$.

If $Q^HQ = I$, Q is *unitary.*

Eigenvalues & Eigenvectors

For $A \in \mathbb{C}^{n\times n}$, if $A\mathbf{x} = \lambda\mathbf{x}$ where $\mathbf{x} \neq 0$, $\mathbf{x}$ is an *eigenvector* of A and λ is the corresponding *eigenvalue.*

Remember, $A - \lambda\mathbf{x}$ is singular iff $\det(A - \lambda I) = 0$. With λ as a variable, $\det(A - \lambda I)$ is A's *characteristic polynomial.*

For nonsingular $T \in \mathbb{C}^{n\times n}$, $T^{-1}AT$ (the *similarity transformation*) is *similar* to A. Similar matrices have the same characteristic polynomial and hence the same eigenvalues (though probably different eigenvectors). This relationship is reflexive, transitive, and symmetric.

A is *diagonalizable* if A is similar to a diagonal matrix $D = T^{-1}AT$. A's eigenvalues are D's diagonals, and the eigenvectors are columns of T since $AT_{:,i} = D_{i,i}T_{:,i}$. A is diagonalizable iff it has n linearly independent eigenvectors.

For symmetric $A \in \mathbb{R}^{n\times n}$, A is diagonalizable, has all real eigenvalues, and the eigenvectors can be the columns of an orthogonal matrix Q where $A = QDQ^T$ is the *eigendecomposition* of A. Further, for symmetric A:

1. The singular values are absolute values of eigenvalues.

2. Is SPD (or SPSD) iff eigenvalues > 0 (or ≥ 0).
3. For SPD, singular values equal eigenvalues.
4. For $B \in \mathbb{R}^{m\times n}$, $m \geq n$, singular values of B are the square roots of B^TB's eigenvalues.

For any $A \in \mathbb{C}^{n\times n}$, the *Schur form* of A is $A = QTQ^H$ with unitary $Q \in \mathbb{C}^{n\times n}$ and upper triangular $T \in \mathbb{C}^{n\times n}$.

In this sheet I denote $\lambda_{|\max|} = \max_{\lambda\in\{\lambda_1,\ldots,\lambda_n\}} |\lambda|$.

For $B \in \mathbb{C}^{n\times n}$, then $\lim_{k\to\infty} B^k = 0$ if $\lambda_{|\max|}(B) < 1$.

Power Methods for Eigenvalues

$\mathbf{x}^{(k+1)} = A\mathbf{x}^{(k)}$ converges to $\lambda_{|\max|}(A)$'s eigenvector.

Once you find an eigenvector $\mathbf{x}$, find the associated eigenvalue λ through the *Raleigh quotient* $\lambda = \frac{\mathbf{x}^{(k)T}A\mathbf{x}^{(k)}}{\mathbf{x}^{(k)T}\mathbf{x}^{(k)}}$.

The *inverse shifted power method* is $\mathbf{x}^{(k+1)} = (A-\sigma I)^{-1}\mathbf{x}^{(k)}$. If A has eigenpairs $(\lambda_1, \mathbf{u}_1), \ldots, (\lambda_n, \mathbf{u}_n)$, then $(A-\sigma I)^{-1}$ has eigenpairs $\left(\frac{1}{\lambda_1-\sigma}, \mathbf{u}_1\right), \ldots, \left(\frac{1}{\lambda_n-\sigma}, \mathbf{u}_n\right)$. Factor $A = QHQ^T$ where H is upper Hessenberg.

To factor $A = QHQ^T$, find successive Householder reflections $H_1, H_2, \ldots$ that zero out rows 2 and lower of column 1, rows 3 and lower of column 2, etc. Then $Q = H_1^T \cdots H_{n-2}^T$.

```
$A^{(0)} = A$
for $k = 0, 1, 2, \ldots$ do
    Set $A^{(k)} - \sigma^{(k)}I = Q^{(k)}R^{(k)}$
    $A^{(k+1)} = R^{(k)}Q^{(k)} + \sigma^{(k)}I$
end for
```

$A^{(k)}$ is similar to A by the orthogonal transform $U^{(k)} = Q^{(0)} \cdots Q^{(k+1)}$. Perhaps choose $\sigma^{(k)}$ as eigenvalues of submatrices of A.

Arnoldi and Lanczos

Given $A \in \mathbb{R}^{n\times n}$ and unit length $\mathbf{q}_1 \in \mathbb{R}^n$, output Q, H such that $A = QHQ^T$. Use Lanczos for symmetric A.

Arnoldi

```
for $k = 1 : n-1$ do
    $\tilde{\mathbf{q}}_{k+1} = A\mathbf{q}_k$
    for $\ell = 1 : k$ do
        $H(\ell, k) = \mathbf{q}_\ell^T \tilde{\mathbf{q}}_{k+1}$
        $\tilde{\mathbf{q}}_{k+1} = \tilde{\mathbf{q}}_{k+1} - H(\ell, k)\mathbf{q}_\ell$
    end for
    $H(k+1, k) = \|\tilde{\mathbf{q}}_{k+1}\|_2$
    $\mathbf{q}_{k+1} = \frac{\tilde{\mathbf{q}}_{k+1}}{H(k+1,k)}$
end for
```

Lanczos

```
$\beta_0 = \|\mathbf{w}_0\|_2$
for $k = 1, 2, \ldots$ do
    $\mathbf{q}_k = \frac{\mathbf{w}_{k-1}}{\beta_{k-1}}$
    $\mathbf{u}_k = A\mathbf{q}_k$
    $\mathbf{v}_k = \mathbf{u}_k - \beta_{k-1}\mathbf{q}_{k-1}$
    $\alpha_k = \mathbf{q}_k^T \mathbf{v}_k$
    $\mathbf{w}_k = \mathbf{v}_k - \alpha_k \mathbf{q}_k$
    $\beta_k = \|\mathbf{w}_k\|_2$
end for
```

For Lanczos, the α_k and β_k are diagonal and subdiagonal entries of the Hermitian tridiagonal T_k, and we have H in Arnoldi. After very few iterations of either method, the eigenvalues of T_k and H will be excellent approximations to the "extreme" eigenvalues of A.

For k iterations, Arnoldi is $O(nk^2)$ times and $O(nk)$ space, Lanczos is $O(nk) + k \cdot \mathcal{M}$ time ($\mathcal{M}$ is time for matrix-vector multiplication) and $O(nk)$ space, or $O(n+k)$ space if old $\mathbf{q}_k$'s are discarded.

Iterative Methods for $A\mathbf{x} = \mathbf{b}$

Useful for sparse A where GE would cause fill-in.

In the *splitting method*, $A = M - N$ and $M\mathbf{v} = \mathbf{c}$ is easily solvable. Then, $\mathbf{x}^{(k+1)} = M^{-1}\left(N\mathbf{x}^{(k)} + \mathbf{b}\right)$. If it converges, the limit point $\mathbf{x}^*$ is a solution to $A\mathbf{x} = \mathbf{b}$.

The error is $\mathbf{e}^{(k)} = (M^{-1}N)^k\mathbf{e}_0$, so splitting methods converge if $\lambda_{|\max|}(M^{-1}N) < 1$.

In the *Jacobi method*, consider M as the diagonals of A. This will fail if A has any zero diagonals.

Conjugate Gradient

Conjugate gradient iteratively solves $A\mathbf{x} = \mathbf{b}$ for SPD A. It is derived from Lanczos and exploits that if A is SPD then T is SPD. It produces the exact solution after n iterations. Time per iteration is $O(n) + \mathcal{M}$.

```
$\mathbf{x}^{(0)} =$ arbitrary ($\mathbf{0}$ is okay)
$\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}^{(0)}$
$\mathbf{p}_0 = \mathbf{r}_0$
for k=0,1,2,... do
    $\alpha_k = (\mathbf{r}_k^T\mathbf{r}_k)/(\mathbf{p}_k^T A\mathbf{p}_k)$
    $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k\mathbf{p}_k$
    $\mathbf{r}_{k+1} = \mathbf{r}_k - \alpha_k A\mathbf{p}_k$
    $\beta_{k+1} = (\mathbf{r}_{k+1}^T\mathbf{r}_{k+1})/(\mathbf{r}_k^T\mathbf{r}_k)$
    $\mathbf{p}_{k+1} = \mathbf{r}_{k+1} - \beta_{k+1}\mathbf{p}_k$
end for
```

Error is reduced by $(\sqrt{\kappa(A)} - 1)/(\sqrt{\kappa(A)} + 1)$ per iteration. Thus, for $\kappa(A) = 1$, CG converges after 1 iteration. To speed up CG, use a *perconditioner* M such that $\kappa(MA) \ll \kappa(A)$ and solve $MA\mathbf{x} = M\mathbf{b}$ instead.

Multivariate Calculus

Provided $f : \mathbb{R}^n \to \mathbb{R}$, the gradient and Hessian are

$$\nabla f = \begin{bmatrix} \frac{\delta f}{\delta x_1} \\ \vdots \\ \frac{\delta f}{\delta x_n} \end{bmatrix}, \nabla^2 f = \begin{bmatrix} \frac{\delta^2 f}{\delta x_1^2} & \frac{\delta^2 f}{\delta x_1 \delta x_2} & \cdots & \frac{\delta^2 f}{\delta x_1 \delta x_n} \\ \vdots & & & \vdots \\ \frac{\delta^2 f}{\delta x_n \delta x_1} & \frac{\delta^2 f}{\delta x_n \delta x_2} & \cdots & \frac{\delta^2 f}{\delta x_n^2} \end{bmatrix}$$

If f is c^2 (2^{nd} partials are all continuous), $\nabla^2 f$ is symmetric. The Taylor expansion for f is $f(\mathbf{x}+\mathbf{h}) = f(\mathbf{x}) + \mathbf{h}^T\nabla f(\mathbf{x}) + \frac{1}{2}\mathbf{h}^T\nabla^2 f(\mathbf{x})\mathbf{h} + O(\|\mathbf{h}\|^3)$

Provided $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$, the Jacobian is

$$\nabla\mathbf{f} = \begin{bmatrix} \delta f_1/\delta x_1 & \cdots & \delta f_1/\delta x_n \\ \vdots & \ddots & \vdots \\ \delta f_m/\delta x_1 & \cdots & \delta f_m/\delta x_n \end{bmatrix}$$

$\mathbf{f}$'s Taylor expansion is $\mathbf{f}(\mathbf{x}+\mathbf{h}) = \mathbf{f}(\mathbf{x}) + \nabla\mathbf{f}(\mathbf{x})\mathbf{h} + O(\|\mathbf{h}\|^2)$.

A *linear* (or *quadratic*) *model* approximates a function $\mathbf{f}$ by the first two (or three) terms of $\mathbf{f}$'s Taylor expansion.

Nonlinear Equation Solving

Given $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$, we want $\mathbf{x}$ such that $\mathbf{f}(\mathbf{x}) = \mathbf{0}$.

In *fixed point iteration*, we choose $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^n$ such that $\mathbf{x}^{(k+1)} = \mathbf{g}(\mathbf{x}^{(k)})$. If it converges to $\mathbf{x}^*$, $\mathbf{g}(\mathbf{x}^*) - \mathbf{x}^* = \mathbf{0}$.

$\mathbf{g}(\mathbf{x}^{(k)}) = \mathbf{g}(\mathbf{x}^*) + \nabla\mathbf{g}(\mathbf{x}^*)(\mathbf{x}^{(k)} - \mathbf{x}^*) + O(\|\mathbf{x}^{(k)} - \mathbf{x}^*\|^2)$ For small $\mathbf{e}^{(k)} = \mathbf{x}^{(k)} - \mathbf{x}^*$, ignore the last term. If $\nabla\mathbf{g}(\mathbf{x}^*)$ has $\lambda_{|\max|} < 1$, then $\mathbf{x}^{(k)} \to \mathbf{x}^*$ as $\|\mathbf{e}^{(k)}\| \leq c^k\|\mathbf{e}^{(0)}\|$ for large k, where $c = \lambda_{|\max|} + \epsilon$, where ϵ is the influence of the ignored last term. This indicates a *linear rate of convergence.*

Suppose for $\nabla\mathbf{g}(\mathbf{x}^*) = QTQ^H$, T is *non-normal*, i.e., T's superdiagonal portion is large relative to the diagonal. Then this may not converge as $\|(\nabla\mathbf{g}(\mathbf{x}^*))^k\|$ initially grows!

In *Newton's method*, $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - (\nabla\mathbf{f}(\mathbf{x}^{(k)}))^{-1}\mathbf{f}(\mathbf{x}^{(k)})$. This converges *quadratically*, i.e., $\|\mathbf{e}^{(k+1)}\| \leq c\|\mathbf{e}^{(k)}\|^2$.

Automatic differentiation takes advantage of the notion that a computer program is nothing but arithmetic operations, and one can apply the chain rule to get the derivative. This may be used to compute Jacobians and determinants.

Optimization

In continuous optimization, $f : \mathbb{R}^n \to \mathbb{R}$ is the *objective function*, $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^m$ holds *equality constraints*, $\mathbf{h} : \mathbb{R}^n \to \mathbb{R}^p$ holds *inequality constraints.*

$$\begin{array}{rl} \min & f(\mathbf{x}) \\ \text{s.t.} & \mathbf{g}(\mathbf{x}) = \mathbf{0} \\ & \mathbf{h}(\mathbf{x}) \geq \mathbf{0} \end{array}$$

We did unrestricted optimization $\min f(\mathbf{x})$ in the course.

A *ball* is a set $B(\mathbf{x}, r) = \{\mathbf{y} \in \mathbb{R}^n : \|\mathbf{x} - \mathbf{y}\| < r\}$.

We have *local minimizers* $\mathbf{x}^*$ which are the best in a region, i.e., $\exists r > 0$ such that $f(\mathbf{x}^*) \leq f(\mathbf{x})$ for all $\mathbf{x} \in B(\mathbf{x}^*, r)$. A *global minizer* is the best local minimizer.

Assume f is c^2. If $\mathbf{x}^*$ is a local minimizer, then $\nabla f(\mathbf{x}^*) = \mathbf{0}$ and $\nabla^2 f(\mathbf{x}^*)$ is PSD. Semi-conversely, if $\nabla f(\mathbf{x}^*) = \mathbf{0}$ and $\nabla^2 f(\mathbf{x}^*)$ is PD, then $\mathbf{x}^*$ is a local minimizer.

Steepest Descent

Go where the function (locally) decreases most rapidly via $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \alpha_k\nabla f(\mathbf{x}^{(k)}$. α_k is explained later. SD is stateless: depends only on the current point. Too slow.

Newton's Method for Unconstrained Min.

Iterate by $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - (\nabla^2 f(\mathbf{x}^{(k)}))^{-1}\nabla f(\mathbf{x}^{(k)})$, derived by solving for where $\nabla f(\mathbf{x}^*) = \mathbf{0}$. If $\nabla^2 f(\mathbf{x}^{(k)})$ is PD and $\nabla f(\mathbf{x}^{(k)}) \neq \mathbf{0}$, the step is a descent direction.

What if the Hessian isn't PD? Use (a) secant method, (b) direction of *negative curvature* where $\mathbf{h}^T\nabla^2 f(\mathbf{x}^{(k)})\mathbf{h} < 0$ where $\mathbf{h}$ or $-\mathbf{h}$ (doesn't work well in practice), (c) *trust region* idea so $\mathbf{h} = -(\nabla^2 f(\mathbf{x}^{(k)}) + tI)^{-1}\nabla f(\mathbf{x}^{(k)})$ (interpolation of NMUM and SD), (d) factor $\nabla^2 f(\mathbf{x}^{(k)})$ by Cholesky when checking for PD, detect 0 pivots, modify that diagonal in $\nabla^2 f(\mathbf{x}^{(k)})$ and keep going (unjustified by theory, but works in practice).

Line Search

Line search, given $\mathbf{x}^{(k)}$ and step $\mathbf{h}$ (perhaps derived from SD or NMUM), finds a $\alpha > 0$ for $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha\mathbf{h}$.

In *exact line search*, optimize $\min f(\mathbf{x}^{(k)} + \alpha\mathbf{h})$ over α. Frowned upon because it's computationally expensive.

In *Armijo* or *backtrack line search*, initialize α. While $f(\mathbf{x}^{(k)} + \alpha\mathbf{h}) > f(\mathbf{x}^{(k)}) + 0.1\alpha\nabla f(\mathbf{x}^{(k)})^T\mathbf{h}$, halve α.

Secant/quasi Newton methods use an approximate always PD $\nabla^2 f$. In Broyden-Fletcher-Goldfarb-Shanno:

```
$B_0 =$ initial approximate Hessian {OK to use $I$.}
for $k = 0, 1, 2, \ldots$ do
    $\mathbf{s}_k = -B_k^{-1}\nabla f(\mathbf{x}^{(k)})$
    $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k\mathbf{s}_k$ {Use special line search for $\alpha_k$!}
    $\mathbf{y}_k = \nabla f(\mathbf{x}^{(k+1)}) - \nabla f(\mathbf{x}^{(k)})$
    $B_{k+1} = B_k + \frac{\mathbf{y}_k\mathbf{y}_k^T}{\alpha\mathbf{y}_k^T\mathbf{s}_k} - \frac{B_k\mathbf{s}_k\mathbf{s}_k^T B_k}{\mathbf{s}_k^T B_k\mathbf{s}_k}$
end for
```

By maintaining B_k in factored form, can iterate in $O(n^2)$ flops. B_k is SPD provided $\mathbf{s}_k^T\mathbf{y} > 0$ (use line search to increase α_k if needed). The secant condition $\alpha_k B_{k+1}\mathbf{s}_k = \mathbf{y}_k$ holds. If BFCS converges, it converges superlinearly.

Non-linear Least Squares

For $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^m$, $m \geq n$, we want the $\mathbf{x}$ for $\min \|\mathbf{g}(\mathbf{x})\|_2$.

In the *Gauss-Newton* method, $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \mathbf{h}$ where $\mathbf{h} = (\nabla\mathbf{g}(\mathbf{x})^T\nabla\mathbf{g}(\mathbf{x}))^{-1}\nabla\mathbf{g}(\mathbf{x})^T\mathbf{g}(\mathbf{x})$. Note that $\mathbf{h}$ is a solution to a linear least squares problem $\min \|\nabla\mathbf{g}(\mathbf{x}^{(k)})\mathbf{h} - \mathbf{g}(\mathbf{x}^{(k)})\|$! GN is derived by applying NMUM to to $\mathbf{g}(\mathbf{x})^T\mathbf{g}(\mathbf{x})$, and dropping a resulting *tensor* (derivative of Jacobian). You keep the quadratic convergence when $\mathbf{g}(\mathbf{x}^*) = \mathbf{0}$, since the tensor $\to 0$ as $k \to \infty$.

Ordinary Differential Equations

ODE (or PDE) has one (or multiple) independent variables.

In *initial value problems*, given $\frac{d\mathbf{y}}{dt} = f(\mathbf{y}, t)$, $\mathbf{y}(t) \in \mathbb{R}^n$, and $\mathbf{y}(0) = \mathbf{y}_0$, we want $\mathbf{y}(t)$ for $t > 0$. Examples include:

1. Exponential growth/decay with $\frac{d\mathbf{y}}{dt} = a\mathbf{y}$, with closed form $\mathbf{y}(t) = \mathbf{y}_0 e^{at}$. Growth if $a > 0$, decay if $a < 0$.
2. Ecological models, $\frac{dy_i}{dt} = f_i(y_1, \ldots, y_n, t)$ for species $i = 1, \ldots, n$. y_i is population size, f_i encodes species relationships.
3. Mechanics, e.g. wall-spring-block models for $F = ma$ ($a = \frac{d^2x}{dt^2}$) and $F = -kx$, so $\frac{d^2x}{dt^2} = \frac{-kx}{m}$. Yields $\frac{d[x,v]^T}{dt} = \begin{bmatrix} v & \frac{-kx}{m} \end{bmatrix}^T$ with $\mathbf{y}_0$ as initial position and velocity.

For *stability of an ODE*, let $\frac{d\mathbf{y}}{dt} = A\mathbf{y}$ for $A \in \mathbb{C}^{n\times n}$. The *stable* or *neutrally spable* or *unstable* case is where $\max_i \Re(\lambda_i(A)) < 0$ or $= 0$ or > 0 respectively.

In *finite difference methods*, approximate $\mathbf{y}(t)$ by discrete points $\mathbf{y}_0$ (given), $\mathbf{y}_1, \mathbf{y}_2, \ldots$ so $\mathbf{y}_k \approx \mathbf{y}(t_k)$ for increasing t_k.

For many IVPs and FDMs, if the *local truncation error* (error at each step) is $O(h^{p+1})$, the *global truncation error* (error overall) is $O(h^p)$. Call p the *order of accuracy.*

To find p, substitute the exact solution into FDM formula, insert a remainder term $+R$ on RHS, use a Taylor series expansion, solve for R, keep only the leading term.

In *Euler's method*, let $\mathbf{y}_{k+1} = \mathbf{y}_k + \mathbf{f}(\mathbf{y}_k, t_k)h_k$ where $h_k = t_{k+1} - t_k$ is the *step size*, and $\mathbf{y}' = \mathbf{f}(\mathbf{y}, t)$ is perhaps computed by finite difference. $p = 1$, very low. Explicit!

A *stiff problem* has widely ranging time scales in the solution, e.g., a transient initial velocity that in the true solution disappears immediately, chemical reaction rate variability over temperature, transients in electical circuits. An explicit method requires h_k to be on the smallest scale!

Backward Euler has $\mathbf{y}_{k+1} = \mathbf{y}_k + h\mathbf{f}(\mathbf{y}_{k+1}, t_{k+1})$. BE is *implicit* ($\mathbf{y}_{k+1}$ on the RHS). If the original program is stable, any h will work!

Miscellaneous

$\sum_{k=1}^{n\pm\text{constant}} k^p = \frac{n^{p+1}}{p+1} + O(n^p)$

$ax^2 + bx + c = 0$. $r_1, r_2 = \frac{-b\pm\sqrt{b^2-4ac}}{2a}$. $r_1r_2 = \frac{c}{a}$

Exact arithmetic is slow, futile for inexact observations, and NA relies on approximate algorithms.

```
41981E07411F9B01BDFF261940159373C07AC167405C5F3DC100037440E8659641759C66418EFA5D41558A37C1CBAF46C196E88040E50EFC413A42A6C160850FC12C8D9BC0CD2F0DC0BDE9B0
C123D876412D0624C0EABD7B402D214B3F0A4E5B4106D552406FFC2840F0964AC0949650409F429E403E7359C0BDD4B93F7FD8ACBFDE0F6BC09EC408C11BEDDC3F8F803940D97DF84093DC28
41B34BB341C05DA0420B176A4207D117C0F7A989419CDAD0C1F8DFAD41E5C5834195F1D1418B3D06422C6A75C22A4E18C22CDE3DC1ADE35D42013BB7C2285485C1E1663DC1ACDB1FC17A6DF7
40FA52A2411987CA40DCBDF541A44735BE00EB0F40DF3240C15A8A0540C35A3F4138F2D9410C2DC4414E9086C18EEABBC19026C0C063C6464158FAD9C16A1481C14D5925C106BD65C00537C9
4202D6BB41DB298A420D8B4D422654DBC058CB0D41B93638C2291BD441EA849341432BF140211A3B42192392C224B279C21A623CC1F39916422C9E28C2144CEFC1EF5BDCC2000AE5C11B4178
41CC3330412812684214CE7F4207333FC154BD9942075FB5C1F700B7412FCDD84074C80C403D08FD421673FCC1D5728FC1CAC47DC1D284A542037446C1BA6373C1FA17FAC1EF9B6CC014B26A
C163F2CCC074096EC1799FFE40E4FC1840C07B6E410D770A41FA3CBF412B7F75C088D8174091DC9D4102F598C17BC8FF3E23BEB940E2D9CBC1CB43A1C18DA47A4163CB5C41865279C101D174
4220409641D895594242E8CC4240AB99C1530BE941D3FC5AC2453B7B41F61FAF41867D4440E2374D4246F36CC2385596C24579EEC20E2C8C424E3ACEC2173FB7C223860AC218DC20C14C4D01
C0A6E2AC3FB233A9C11575FFBFDC81A64033140140A330D740D2515940E3266941A6486541983485415DC696C1832CAAC126A19E410CF23DC0D3BDEBC18BD0F53F812F1C4021EB80C1221B86
41F9E74341E49DC241F356E941F21CA1C0E5408E415C8840C220AAF941A2DF7E41C3770241B8E7C242073D67C22E7DC8C2316977C195288E4227CC9FC204D34FC2054DE7C1D27A3BC122FAC5
3F20F6F540239D6D407CE8FF4131C69FC0D9CC44416A05B0C0CF9E5C416196E23FF192CB414ABA62420A0178C180DBE7C10FB02FC0C938A140A01794C1ACE03DC121E557C102D79BC05A0B9A
4175D137410A666C41BA03F141A11609C119ADFE416CF000C1D3AC7840CF8936413BE95F4152284941EC4CD3C1483DF5C1BCE0F7C138674041D88129C199E0E9C1E35B6FC1AE0421BF0B7F9D
40E9B16D415B8F0240C2914D4151B42BBFC6AAC141448D07C16917974181226041685A324194A32E41CE5FA0C1E1375CC17115ADC0CFB430417742E8C1E0BD4FC171DEE7C12546BAC0FE0C27
41980F6140FA939942014021420F7E01C14930BE41C5771EC200F696414496D940291CDA40952454421D90C1C1A89B86C1C3FFE5C193999B41F97E2EC1C69876C201193DC1DA288C3FFE4D59
40329D993FFB03ED412BD3E6419ED013C029891B4180279EC0B155BC415D858F4025361D4107EE4B41E18AE3C19F2909C1551577C10EE2DC4144D1B9C1D025BAC14F6FEBC1073B6AC0959483
41E6259441B5FEB44208EE924228B899C11B6D7241C5B937C20911A441B72EEE40EC50BCC00D65FF4210A2CFC2113849C20CAC27C201171542046B73C1B0223EC1BBCC08C1CDDF00C13A5785
40F00474411D672E4027EED93F03DF7BC0850F2C4092866FC171638B4122D01C41A3035E41C4A1F841A87B85C1AB7D24C1A1FB1E40652E054180AB93C19F5F5BC1295F66C1426F7BC04297A0
415D4E0D41629AE44177E6A74195DAC5C0BF9F0E41541266C16230F3419DBEC9415618FB4199C3E042029A8BC2072CB5C1EC52F7C0FB2260418ED6C8C1FE42DBC19C13AFC08524F8C15C1E8A
C02FB43F40CB7D46C171D9F9C10C9FC53FAD827BBD1BED2040B11CA141180061412507F34190D6B5408539A9C18D564EC0F56DAD4158BA12C007AFE5C1581D463F9846B74093F7C0408EA433
```