Chapter III

Orthogonality

III.1 Orthogonality and Projections

III.1.1 Orthogonal vectors

Recall that the dot product, or inner product of two vectors

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

is denoted by $\mathbf{x} \cdot \mathbf{y}$ or $\langle \mathbf{x}, \mathbf{y} \rangle$ and defined by

$$\mathbf{x}^T \mathbf{y} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \sum_{i=1}^{n} x_i y_i$$

Some important properties of the inner product are symmetry

$$\mathbf{x} \cdot \mathbf{y} = \mathbf{y} \cdot \mathbf{x}$$

and linearity

$$(c_1 \mathbf{x}_1 + c_2 \mathbf{x}_2) \cdot \mathbf{y} = c_1 \mathbf{x}_1 \cdot \mathbf{y} + c_2 \mathbf{x}_2 \cdot \mathbf{y}.$$

The norm, or length, of a vector is given by

$$\|\mathbf{x}\| = \sqrt{\mathbf{x} \cdot \mathbf{x}} = \sqrt{\sum_{i=1}^{n} x_i^2}$$

An important property of the norm is that $\|\mathbf{x}\| = 0$ implies that $\mathbf{x} = \mathbf{0}$.

The geometrical meaning of the inner product is given by

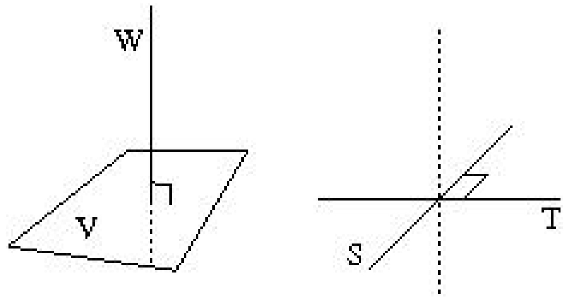
$$\mathbf{x} \cdot \mathbf{y} = \|\mathbf{x}\| \|\mathbf{y}\| \cos(\theta)$$

where θ is the angle between the vectors. The angle θ can take values from 0 to π.

The Cauchy–Schwarz inequality states

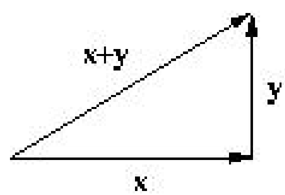
$$|\mathbf{x} \cdot \mathbf{y}| \leq \|\mathbf{x}\| \|\mathbf{y}\|.$$

It follows from the previous formula because $|\cos(\theta)| \leq 1$. The only time that equality occurs in the Cauchy–Schwarz inequality, that is $\mathbf{x} \cdot \mathbf{y} = \|\mathbf{x}\| \|\mathbf{y}\|$, is when $\cos(\theta) = \pm 1$ and θ is either 0 or π. This means that the vectors are pointed in the same or in the opposite directions.



The vectors $\mathbf{x}$ and $\mathbf{y}$ are orthogonal if $\mathbf{x} \cdot \mathbf{y} = 0$. Geometrically this means either that one of the vectors is zero or that they are at right angles. This follows from the formula above, since $\cos(\theta) = 0$ implies $\theta = \pi/2$.

Another way to see that $\mathbf{x} \cdot \mathbf{y} = 0$ means that vectors are orthogonal is from Pythagoras' formula. If $\mathbf{x}$ and $\mathbf{y}$ are at right angles then $\|\mathbf{x}\|^2 + \|\mathbf{y}\|^2 = \|\mathbf{x} + \mathbf{y}\|^2$.



But $\|\mathbf{x} + \mathbf{y}\|^2 = (\mathbf{x} + \mathbf{y}) \cdot (\mathbf{x} + \mathbf{y}) = \|\mathbf{x}\|^2 + \|\mathbf{y}\|^2 + 2\mathbf{x} \cdot \mathbf{y}$ so Pythagoras' formula holds exactly when $\mathbf{x} \cdot \mathbf{y} = 0$.

To compute the inner product of (column) vectors `X` and `Y` in MATLAB/Octave we use the formula $\mathbf{x} \cdot \mathbf{y} = \mathbf{x}^T \mathbf{y}$. Thus the inner product can be computed using `X'*Y`. (If `X` and `Y` are row vectors, the formula is `X*Y'`.)

The norm of a vector `X` is computed by `norm(X)`. In MATLAB/Octave inverse trig functions are computed with `asin(), acos()` etc. So the angle between column vectors `X` and `Y` could be computed as

```
> acos(X'*Y/(norm(X)*norm(Y)))
```

III.1.2 Orthogonal subspaces

Two subspaces V and W are said to be orthogonal if every vector in V is orthogonal to every vector in V. In this case we write $V \perp W$.

In this figure $V \perp W$ and also $S \perp T$.

A related concept is the orthogonal complement. The orthogonal complement of V, denoted $V^\perp$ is the subspace containing all vectors orthogonal to V. In the figure $W = V^\perp$ but $T \neq S^\perp$ since T contains only some of the vectors orthogonal to S.

If we take the orthogonal complement of $V^\perp$ we get back the original space V: This is certainly plausible from the pictures. It is also obvious that $V \subseteq (V^\perp)^\perp$, since any vector in V is perpendicular to vectors in $V^\perp$. If there were a vector in $(V^\perp)^\perp$ not contained in V we could subtract its projection onto V (defined below) and end up with a non-zero vector in $(V^\perp)^\perp$ that is also in $V^\perp$. Such a vector would be orthogonal to itself, which is impossible. This shows that

$$(V^\perp)^\perp = V.$$

One consequence of this formula is that $V = W^\perp$ implies $V^\perp = W$. Just take the orthogonal complement of both sides and use $(W^\perp)^\perp = W$.

III.1.3 The fundamental subspaces of a matrix revisited

Recall that we discovered orthogonality relations between vectors in the fundamental subspaces of an incidence matrix for a graph. Now we will show that these relations are valid for any matrix A.

These relations are based on the formula

$$(A^T\mathbf{x}) \cdot \mathbf{y} = \mathbf{x} \cdot (A\mathbf{y})$$

This formula follows from the product formula $(AB)^T = B^T A^T$ for transposes, since

$$(A^T\mathbf{x}) \cdot \mathbf{y} = (A^T\mathbf{x})^T\mathbf{y} = \mathbf{x}^T(A^T)^T\mathbf{y} = \mathbf{x}^T A\mathbf{y} = \mathbf{x} \cdot (A\mathbf{y})$$

Let A be an $m \times n$ matrix. Then $N(A)$ and $R(A^T)$ are subspaces of $\mathbb{R}^n$ while $N(A^T)$ and $R(A)$ are subspaces of $\mathbb{R}^m$. These two pairs of subspaces are orthogonal:

$$\begin{aligned} N(A) &= R(A^T)^\perp \\ N(A^T) &= R(A)^\perp \end{aligned}$$

We will show that the first equality holds for any A. The second equality then follows by applying the first one to A^T.

First, we show that $N(A) \subseteq R(A^T)^\perp$. To do this, start with any vector $\mathbf{x} \in N(A)$. This means that $A\mathbf{x} = \mathbf{0}$. If we compute the inner product of $\mathbf{x}$ with any vector in $R(A^T)$, that is, any vector of the form $A^T\mathbf{y}$, we get $(A^T\mathbf{y}) \cdot \mathbf{x} = \mathbf{y} \cdot A\mathbf{x} = \mathbf{y} \cdot \mathbf{0} = 0$. Thus $\mathbf{x} \in R(A^T)^\perp$. This shows $N(A) \subseteq R(A^T)^\perp$.

Now we show the opposite inclusion, $R(A^T)^\perp \subseteq N(A)$. This time we start with $\mathbf{x} \in R(A^T)^\perp$. This means that $\mathbf{x}$ is orthogonal to every vector in $R(A^T)$, that is, to every vector of the form $A^T\mathbf{0}$. So $(A^T\mathbf{0})\cdot\mathbf{x} = \mathbf{0}\cdot(A\mathbf{x}) = 0$ for every $\mathbf{0}$. Pick $\mathbf{0} = A\mathbf{x}$. Then $(A\mathbf{x})\cdot(A\mathbf{x}) = \|A\mathbf{x}\|^2 = 0$. This implies $A\mathbf{x} = \mathbf{0}$ so $\mathbf{x} \in N(A)$. We can conclude that $R(A^T)^\perp \subseteq N(A)$.

These two inclusions establish that $N(A) = R(A^T)^\perp$.

Let's verify these orthogonality relations in an example. Let

$$A = \begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 3 & 0 & 1 \\ 2 & 5 & 1 & 2 \end{bmatrix}$$

Then

$$\text{rref}(A) = \begin{bmatrix} 1 & 0 & 3 & 1 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \text{rref}(A^T) = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Thus we get

$$N(A) = \text{span}\left\{ \begin{bmatrix} -3 \\ 1 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \right\}$$

$$R(A) = \text{span}\left\{ \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 2 \\ 3 \\ 5 \end{bmatrix} \right\}$$

$$N(A^T) = \text{span}\left\{ \begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix} \right\}$$

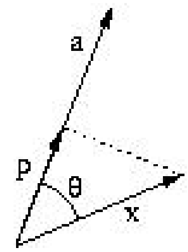
$$R(A^T) = \text{span}\left\{ \begin{bmatrix} 1 \\ 0 \\ 3 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ -1 \\ 0 \end{bmatrix} \right\}$$

We can now verify directly that every vector in the basis for $N(A)$ is orthogonal to every vector in the basis for $R(A^T)$, and similarly for $N(A^T)$ and $R(A)$.

Does the equation

$$A\mathbf{x} = \begin{bmatrix} 2 \\ 1 \\ 3 \end{bmatrix}$$

have a solution? We can use the ideas above to answer this question easily. We are really asking whether $\begin{bmatrix} 2 \\ 1 \\ 3 \end{bmatrix}$ is contained in $R(A)$. But, according to the orthogonality relations, this



is the same as asking whether $\begin{bmatrix}2\\1\\3\end{bmatrix}$ is contained in $N(A^T)^\perp$. This is easy to check. Simply compute the dot product

$$\begin{bmatrix}2\\1\\3\end{bmatrix} \cdot \begin{bmatrix}-1\\-1\\1\end{bmatrix} = -2 - 1 + 3 = 0.$$

Since the result is zero, we conclude that a solution exists.

III.1.4 Projections

Let start with the formula for the projection of a vector $\mathbf{x}$ onto a line containing the non-zero vector $\mathbf{a}$.

The length of the projected vector $\mathbf{p}$ is

$$\|\mathbf{p}\| = \|\mathbf{x}\|\cos(\theta) = \frac{\|\mathbf{a}\|\|\mathbf{x}\|\cos(\theta)}{\|\mathbf{a}\|} = \frac{\mathbf{a}\cdot\mathbf{x}}{\|\mathbf{a}\|} = \frac{\mathbf{a}^T\mathbf{x}}{\|\mathbf{a}\|}$$

To get the vector $\mathbf{p}$ start with the unit vector $\mathbf{a}/\|\mathbf{a}\|$ and stretch it by an amount $\|\mathbf{p}\|$. This gives

$$\mathbf{p} = \|\mathbf{p}\|\frac{\mathbf{a}}{\|\mathbf{a}\|} = \frac{1}{\|\mathbf{a}\|^2}\mathbf{a}\mathbf{a}^T\mathbf{x}$$

This can be written $\mathbf{p} = P\mathbf{x}$ where P is the projection matrix

$$P = \frac{1}{\|\mathbf{a}\|^2}\mathbf{a}\mathbf{a}^T.$$

The matrix P satisfies two properties:

$$(1) \quad P^2 = P$$

and

$$(2) \quad P^T = P.$$

Property (1) says that any vector in the range of P is not changed by P, since $P(P\mathbf{x}) = P^2\mathbf{x} = P\mathbf{x}$.

Property (2) implies that $N(P) = R(P)^{\perp}$. This follows from the orthogonality relation $N(P) = R(P^T)^{\perp}$ since $P = P^T$.

Any matrix satisfying (1) and (2) is called an orthogonal projection. (Warning: this is a different concept than that of an orthogonal matrix which we will see later.)

Example: What is the projection of $\mathbf{x} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$ in the direction of $\mathbf{a} = \begin{bmatrix} 1 \\ 2 \\ -1 \end{bmatrix}$? Let's calculate the projection matrix P and compute $P\mathbf{x}$. We can also verify that $P^2 = P$ and $P^T = P$.

```
>x = [1 1 1]';
>a = [1 2 -1]';
>P = (a'*a)^(-1)*a*a'

P =

   0.16667   0.33333  -0.16667
   0.33333   0.66667  -0.33333
  -0.16667  -0.33333   0.16667

>P*x

ans =

   0.33333
   0.66667
  -0.33333

>P*P

ans =

   0.16667   0.33333  -0.16667
   0.33333   0.66667  -0.33333
  -0.16667  -0.33333   0.16667
```

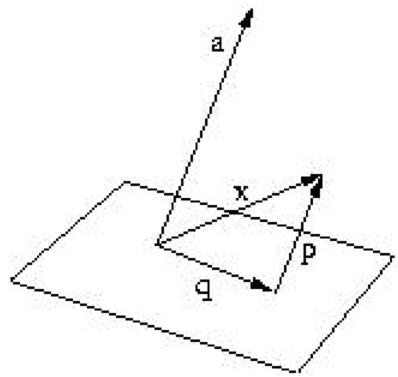
The projection of $\mathbf{x}$ on to the plane orthogonal to $\mathbf{a}$ is given by $\mathbf{q} = \mathbf{x} - \mathbf{p}$.

Thus we can write

$$\mathbf{q} = \mathbf{x} - P\mathbf{x} = (I - P)\mathbf{x} = Q\mathbf{x}$$

where

$$Q = I - P$$



The matrix Q is also an orthogonal projection, since

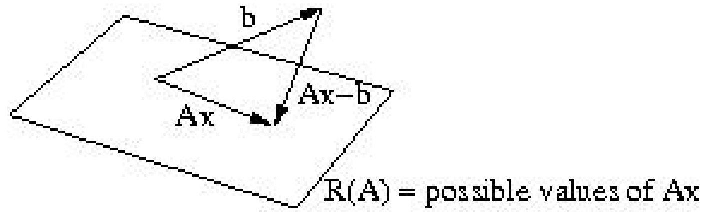
$$Q^2 = (I - P)(I - P) = I - 2P + P^2 = I - P = Q$$

and

$$Q^T = I^T - P^T = I - P = Q.$$

Continuing with the example above, if we want to compute the projection matrix onto the plane perpendicular to $\mathbf{a}$ we compute $Q = I - P$. Then $Q\mathbf{x}$ is the projection of $\mathbf{x}$ onto the plane. We can also check that $Q^2 = Q$.

```
> Q = eye(3) - P

Q =

   0.83333  -0.33333   0.16667
  -0.33333   0.33333   0.33333
   0.16667   0.33333   0.83333

>Q*x

ans =

   0.66667
   0.33333
   1.33333

>Q^2

ans =

   0.83333  -0.33333   0.16667
```




```
-0.33333   0.33333   0.33333
 0.16667   0.33333   0.83333
```

III.1.5 Least squares solutions

Suppose that $\mathbf{b} \notin R(A)$ so that $A\mathbf{x} = \mathbf{b}$ does not have a solution. What vector $\mathbf{x}$ is closest to being a solution?

We want to determine $\mathbf{x}$ so that $A\mathbf{x}$ is as close as possible to $\mathbf{b}$. From the picture, we can see that this will happen when $A\mathbf{x} - \mathbf{b}$ is orthogonal to $R(A)$. But the vectors orthogonal to $R(A)$ are exactly the vectors in $N(A^T)$. Thus the vector we are looking for will satisfy the equation

$$A^T A\mathbf{x} = A^T\mathbf{b}$$

This is the least squares equation, and a solution to this equation is called a least squares solution.

(Aside: We can also use Calculus to derive the least squares equation. We want to minimize $\|A\mathbf{x} - \mathbf{b}\|^2$. Computing the gradient and setting it to zero results in the same equations.)

It turns out that the least squares equation always has a solution. Another way of saying this is $R(A^T) = R(A^T A)$. Instead of checking this, we can verify that the orthogonal complements $N(A)$ and $N(A^T A)$ are the same. But this is something we showed before, when we considered the incidence matrix D for a graph.

If $\mathbf{x}$ solves the least squares equation, the vector $A\mathbf{x}$ is the projection of $\mathbf{b}$ onto the range $R(A)$. If $A^T A$ is invertible (this happens when $N(A) = N(A^T A) = \{\mathbf{0}\}$), we can obtain a formula for the projection. Starting with the least squares equation we multiply by $(A^T A)^{-1}$ to obtain

$$\mathbf{x} = (A^T A)^{-1} A^T\mathbf{b}$$

so that

$$A\mathbf{x} = A(A^T A)^{-1} A^T\mathbf{b}.$$

Thus the projection matrix is given by

$$P = A(A^T A)^{-1} A^T$$

It is worthwhile pointing out that if we say that the solution of the least squares equation gives the "best" approximation to a solution, what we really mean is that it minimizes the distance, or equivalently, its square

$$\|A\mathbf{x} - \mathbf{b}\|^2 = \sum ((A\mathbf{x})_i - \mathbf{b}_i)^2.$$

There are other ways of measuring how far $A\mathbf{x}$ is from $\mathbf{b}$, for example the so-called L^1 norm

$$\|A\mathbf{x} - \mathbf{b}\|_1 = \sum |(A\mathbf{x})_i - \mathbf{b}_i|$$

Minimizing the L^1 norm will result in a different "best" solution, that may be preferable under some circumstances. However, it is much more difficult to find!

III.1.6 Straight line fit

Suppose we have some data points $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ and we want to fit a straight line $y = ax + b$ through them. This is similar to the interpolation problems we considered before, but now there is no solution, unless, of course, the points actually do happen to all lie on a single line.

The equations we want to solve are

$$\begin{aligned} ax_1 + b &= y_1 \\ ax_2 + b &= y_2 \\ &\vdots \\ ax_n + b &= y_n \end{aligned}$$

The unknowns are a and b. The matrix form of the equation is

$$\begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \\ x_n & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

These equations will not have a solution (unless the points really do happen to lie on the same line.) To find the least squares solution, we compute

$$\begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ 1 & 1 & \cdots & 1 \end{bmatrix} \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \\ x_n & 1 \end{bmatrix} = \begin{bmatrix} \sum x_i^2 & \sum x_i \\ \sum x_i & n \end{bmatrix}$$

and

$$\begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ 1 & 1 & \cdots & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \sum x_i y_i \\ \sum y_i \end{bmatrix}$$

This results in the least squares equations

$$\begin{bmatrix} \sum x_i^2 & \sum x_i \\ \sum x_i & n \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum x_i y_i \\ \sum y_i \end{bmatrix}$$

which are easily solved.

III.1.7 Football rankings

We can try to use least squares to rank football teams. To start with, suppose we have three teams. We pretend each team has a value v_1, v_2 and v_3 such that when two teams play, the difference in scores is the difference in values. So, if the season's games had the following results

1 vs. 2	30	40
1 vs. 2	20	40
2 vs. 3	10	0
3 vs. 1	0	5
3 vs. 2	5	5

then the v_i's would satisfy the equations

$$\begin{aligned} v_2 - v_1 &= 10 \\ v_2 - v_1 &= 20 \\ v_2 - v_3 &= 10 \\ v_1 - v_3 &= 5 \\ v_2 - v_3 &= 0 \end{aligned}$$

Of course, there is no solution to these equations. Nevertheless we can find the least squares solution. The matrix form of the equations is

$$D\mathbf{v} = \mathbf{b}$$

with

$$D = \begin{bmatrix} -1 & 1 & 0 \\ -1 & 1 & 0 \\ 0 & 1 & -1 \\ -1 & 0 & 1 \\ 0 & 1 & -1 \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} 10 \\ 20 \\ 10 \\ 5 \\ 0 \end{bmatrix}$$

The least squares equation is

$$D^T D\mathbf{v} = D^T\mathbf{v}$$

or

$$\begin{bmatrix} 3 & -2 & -1 \\ -2 & 4 & -2 \\ -1 & -2 & 3 \end{bmatrix} \mathbf{v} = \begin{bmatrix} -35 \\ 40 \\ -5 \end{bmatrix}$$

Before going on, notice that D is an incidence matrix. What is the graph? (Answer: the nodes are the teams and they are joined by an edge with the arrow pointing from the losing team to the winning team. This graph may have more than one edge joining to nodes, if two teams play more than once. This is sometimes called a multi-graph.). We saw that in this situation $N(D)$ is not empty, but contains vectors whose entries are all the same. The situation is the same as for resistances, it is only differences in v_i's that have a meaning.

We can solve this equation in MATLAB/Octave. The straightforward way is to compute

```
>L = [3 -2 -1;-2 4 -2;-1 -2 3];
>b = [-35; 40; -5];
>rref([L b])

ans =

   1.00000   0.00000  -1.00000  -7.50000
   0.00000   1.00000  -1.00000   6.25000
   0.00000   0.00000   0.00000   0.00000
```

As expected, the solution is not unique. The general solution, depending on the parameter s is

$$\mathbf{v} = s \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} + \begin{bmatrix} -7.5 \\ 6.25 \\ 0 \end{bmatrix}$$

We can choose s so that the v_i for one of the teams is zero. This is like grounding a node in a circuit. So, by choosing $s = 7.5$, $s = -6.25$ and $s = 0$ we obtain the solutions $\begin{bmatrix} 0 \\ 13.75 \\ 7.5 \end{bmatrix}$, $\begin{bmatrix} -13.75 \\ 0 \\ -6.25 \end{bmatrix}$ or $\begin{bmatrix} -7.5 \\ 6.25 \\ 0 \end{bmatrix}$.

Actually, it is easier to compute a solution with one of the v_i's equal to zero directly. If $\mathbf{v} = \begin{bmatrix} 0 \\ v_2 \\ v_3 \end{bmatrix}$ then $\mathbf{v}_2 = \begin{bmatrix} v_2 \\ v_3 \end{bmatrix}$ satisfies the equation $L_2\mathbf{v}_2 = \mathbf{b}_2$ where the matrix L_2 is the bottom right 2×2 block of L and $\mathbf{b}_2$ are the last two entries of $\mathbf{b}$.

```
>L2 = L(2:3,2:3);
>b2 = b(2:3);
>L2\b2

ans =
   13.7500
    7.5000
```

We can try this on real data. The football scores for the 2007 CFL season can be found at `http://www.cfl.ca/index.php?module=sked&func=view&year=2007`. The differences in scores for the first 20 games are in `cfl.m`. The order of the teams is BC, Calgary, Edmonton, Hamilton, Montreal, Saskatchewan, Toronto, Winnipeg. Repeating the computation above for this data we find the ranking to be (running the file `cfl.m`)

```
v =

    0.00000
  -12.85980
  -17.71983
  -22.01884
  -11.37097
   -1.21812
    0.87588
  -20.36966
```

Not very impressive, if you consider that the second-lowest ranked team (Winnipeg) ended up in the Grey Cup game!

Summary: Math Concepts

- dot product, norm and orthogonal vectors
- Cauchy–Schwarz inequality
- orthogonal subspaces
- orthogonal complement
- orthogonality of $N(A)$, $R(A^T)$ and $N(A^T)$, $R(A)$
- projection onto a line, onto a (hyper)plane orthogonal to a line
- least squares solution
- projection onto the range of a matrix A with $N(A) = \{\mathbf{0}\}$
- straight line fit using least squares
- ranking using least squares

Summary: MATLAB/Octave Concepts

- Computing inner (dot) products, vector norms, inverse trig functions.
- Computing projections
- Least squares calculations.

III.2 Orthonormal bases, Orthogonal Matrices, Gram–Schmidt and QR decomposition

III.2.1 Orthonormal bases

A basis $\mathbf{q}_1, \mathbf{q}_2, \ldots$ is called orthonormal if

1. $\|\mathbf{q}_i\| = 1$ for every i (normal)
2. $\mathbf{q}_i \cdot \mathbf{q}_j = 0$ for $i \neq j$ (ortho).

The standard basis for $\mathbb{R}^n$ given by

$$\mathbf{e}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \end{bmatrix}, \quad \mathbf{e}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \end{bmatrix}, \quad \mathbf{e}_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ \vdots \end{bmatrix}, \quad \cdots$$

is an orthonormal basis for $\mathbb{R}^n$. Another orthonormal basis for $\mathbb{R}^2$ is

$$\mathbf{q}_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad \mathbf{q}_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

If you expand a vector in an orthonormal basis, it's very easy to find the coefficients in the expansion. Suppose

$$\mathbf{v} = c_1\mathbf{q}_1 + c_2\mathbf{q}_2 + \cdots + c_n\mathbf{q}_n$$

for some orthonormal basis $\mathbf{q}_1, \mathbf{q}_2, \ldots$. Then, if we take the dot product of both sides with $\mathbf{q}_k$, we get

$$\begin{aligned} \mathbf{q}_k \cdot \mathbf{v} &= c_1\mathbf{q}_k \cdot \mathbf{q}_1 + c_2\mathbf{q}_k \cdot \mathbf{q}_2 + \cdots + c_k\mathbf{q}_k \cdot \mathbf{q}_k \cdots + c_n\mathbf{q}_k \cdot \mathbf{q}_n \\ &= 0 + 0 + \cdots + c_k + \cdots + 0 \\ &= c_k \end{aligned}$$

This gives a convenient formula for each c_k. For example, in the expansion

$$\begin{bmatrix} 1 \\ 2 \end{bmatrix} = c_1 \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} + c_2 \frac{1}{\sqrt{2}} \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

we have

$$\begin{aligned} c_1 &= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \frac{3}{\sqrt{2}} \\ c_2 &= \frac{1}{\sqrt{2}} \begin{bmatrix} -1 \\ 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \frac{1}{\sqrt{2}} \end{aligned}$$

III.2.2 Orthogonal matrices

An $n \times n$ matrix Q is called *orthogonal* if $Q^T Q = I$ (equivalently if $Q^T = Q^{-1}$). If the columns of Q are $\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n$ then Q is orthogonal if

$$Q^T Q = \begin{bmatrix} \mathbf{q}_1^T \\ \mathbf{q}_2^T \\ \vdots \\ \mathbf{q}_n^T \end{bmatrix} \begin{bmatrix} \mathbf{q}_1 & \mathbf{q}_2 & \cdots & \mathbf{q}_n \end{bmatrix} = \begin{bmatrix} \mathbf{q}_1 \cdot \mathbf{q}_1 & \mathbf{q}_1 \cdot \mathbf{q}_2 & \cdots & \mathbf{q}_1 \cdot \mathbf{q}_n \\ \mathbf{q}_2 \cdot \mathbf{q}_1 & \mathbf{q}_2 \cdot \mathbf{q}_2 & \cdots & \mathbf{q}_2 \cdot \mathbf{q}_n \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{q}_n \cdot \mathbf{q}_1 & \mathbf{q}_n \cdot \mathbf{q}_2 & \cdots & \mathbf{q}_n \cdot \mathbf{q}_n \end{bmatrix} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}.$$

This is the same as saying that the columns of Q form an orthonormal basis.

Recall that for square matrices a left inverse is automatically also a right inverse. So if $Q^T Q = I$ then $QQ^T = I$ too. This means that Q^T is an orthogonal matrix whenever Q is. This proves the (non-obvious) fact that if the columns of an square matrix form an orthonormal basis, then so do the rows!

If you multiply together two orthogonal matrices Q_1 and Q_2, then the product $Q_1 Q_2$ is also orthogonal. This follows from the fact that $(Q_1 Q_2)^T = Q_2^T Q_1^T$ so that

$$(Q_1 Q_2)^T Q_1 Q_2 = Q_2^T Q_1^T Q_1 Q_2 = Q_2^T I Q_2 = I$$

Another way of recognizing orthogonal matrices is by their action on vectors. Suppose Q is orthogonal. Then

$$\|Q\mathbf{v}\|^2 = (Q\mathbf{v}) \cdot (Q\mathbf{v}) = \mathbf{v} \cdot (Q^T Q\mathbf{v}) = \mathbf{v} \cdot \mathbf{v} = \|\mathbf{v}\|^2$$

This implies that $\|Q\mathbf{v}\| = \|\mathbf{v}\|$. In other words, orthogonal matrices don't change the lengths of vectors.

The converse is also true. If a matrix Q doesn't change the lengths of vectors then it must be orthogonal. To see this, suppose that $\|Q\mathbf{v}\| = \|\mathbf{v}\|$ for every $\mathbf{v}$. Then the calculation above shows that $\mathbf{v} \cdot (Q^T Q\mathbf{v}) = \mathbf{v} \cdot \mathbf{v}$ for every $\mathbf{v}$. Applying this to $\mathbf{v} + \mathbf{w}$ we find

$$(\mathbf{v} + \mathbf{w}) \cdot \big(Q^T Q(\mathbf{v} + \mathbf{w})\big) = (\mathbf{v} + \mathbf{w}) \cdot (\mathbf{v} + \mathbf{w})$$

Expanding, this gives

$$\mathbf{v} \cdot (Q^T Q\mathbf{v}) + \mathbf{w} \cdot (Q^T Q\mathbf{w}) + \mathbf{v} \cdot (Q^T Q\mathbf{w}) + \mathbf{w} \cdot (Q^T Q\mathbf{v}) = \mathbf{v} \cdot \mathbf{v} + \mathbf{w} \cdot \mathbf{w} + \mathbf{v} \cdot \mathbf{w} + \mathbf{w} \cdot \mathbf{v}$$

Since $\mathbf{v} \cdot (Q^T Q\mathbf{v}) = \mathbf{v} \cdot \mathbf{v}$ and $\mathbf{w} \cdot (Q^T Q\mathbf{w}) = \mathbf{w} \cdot \mathbf{w}$ we can cancel these terms. Also $\mathbf{w} \cdot (Q^T Q\mathbf{v}) = ((Q^T Q)^T \mathbf{w}) \cdot \mathbf{v} = (Q^T Q\mathbf{w}) \cdot \mathbf{v} = \mathbf{v} \cdot (Q^T Q\mathbf{w})$. So on each side of the equation, the two remaining terms are the same. Thus

$$\mathbf{v} \cdot (Q^T Q\mathbf{w}) = \mathbf{v} \cdot \mathbf{w}$$

This equation holds for every choice of vectors $\mathbf{v}$ and $\mathbf{w}$. If $\mathbf{v} = \mathbf{e}_i$ and $\mathbf{w} = \mathbf{e}_j$ then the left side is the i, jth matrix element $Q_{i,j}$ of Q while the right side is the $\mathbf{e}_i \cdot \mathbf{e}_j$, which is i, jth matrix element of the identity matrix. Thus $Q^T Q = I$ and Q is orthogonal.

III.2.3 Matrices with orthogonal columns

An orthogonal matrix Q is a square matrix whose columns form an orthonormal basis. We can also think about matrices whose columns are orthonormal, but with too few columns to form a basis. Such a matrix looks like

$$Q = \begin{bmatrix} \mathbf{q}_1 & \mathbf{q}_2 & \cdots & \mathbf{q}_k \end{bmatrix}$$

where the $\mathbf{q}_i$'s form an orthonormal set (that is, $\mathbf{q}_i \cdot \mathbf{q}_i = 1$ for all i and $\mathbf{q}_i \cdot \mathbf{q}_j = 0$ if $i \neq j$), but k is less than the dimension n. For such a matrix Q^TQ and QQ^T are both well defined, but they have different sizes. The matrix Q^TQ is the $k \times k$ identity matrix. This follows from the same calculation we did for orthogonal matrices:

$$Q^TQ = \begin{bmatrix} \mathbf{q}_1^T \\ \mathbf{q}_2^T \\ \vdots \\ \mathbf{q}_k^T \end{bmatrix} \begin{bmatrix} \mathbf{q}_1 & \mathbf{q}_2 & \cdots & \mathbf{q}_k \end{bmatrix} = \begin{bmatrix} \mathbf{q}_1 \cdot \mathbf{q}_1 & \mathbf{q}_1 \cdot \mathbf{q}_2 & \cdots & \mathbf{q}_1 \cdot \mathbf{q}_k \\ \mathbf{q}_2 \cdot \mathbf{q}_1 & \mathbf{q}_2 \cdot \mathbf{q}_2 & \cdots & \mathbf{q}_2 \cdot \mathbf{q}_k \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{q}_k \cdot \mathbf{q}_1 & \mathbf{q}_k \cdot \mathbf{q}_2 & \cdots & \mathbf{q}_k \cdot \mathbf{q}_k \end{bmatrix}$$

But if $k < n$ then QQ^T cannot possibly be the $n \times n$ identity matrix. You can see this, for example, from the fact that the $k \times n$ matrix Q^T must have a non-trivial null space. In fact, QQ^T is a projection matrix that projects onto the vector space spanned by the columns of Q. To see this recall that the general formula for this projection is $P = Q(Q^TQ)^{-1}Q^T$. When $Q^TQ = I$ this reduces to $P = QQ^T$.

III.2.4 Gram–Schmidt procedure

The Gram–Schmidt procedure takes any collection of linearly independent vectors $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_k$ and produces an orthonormal set $\mathbf{q}_1, \mathbf{q}_2, \ldots \mathbf{q}_k$. Each $\mathbf{q}_j$ is a linear combination of $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_j$, so that for every j, the first j $\mathbf{q}_i$'s span the same subspace as the first j $\mathbf{a}_i$'s.

So let's start with a collection $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_k$ of linearly independent vectors.

Define $\mathbf{q}_1$ to be the unit vector in the direction of $\mathbf{a}_1$:

$$\mathbf{q}_1 = \frac{1}{\|\mathbf{a}_1\|}\mathbf{a}_1$$

To define $\mathbf{q}_2$ we start with $\mathbf{a}_2$. The component of $\mathbf{a}_2$ in the direction of $\mathbf{q}_1$ is the projection $(\mathbf{a}_2 \cdot \mathbf{q}_1)\mathbf{q}_1$. If we start with $\mathbf{a}_2$ and subtract this component we end up with the vector $\mathbf{a}_2 - (\mathbf{a}_2 \cdot \mathbf{q}_1)\mathbf{q}_1$. This vector is orthogonal to $\mathbf{q}_1$, which can be verified by computing its dot product with $\mathbf{q}_1$ and checking that it is zero. We define $\mathbf{q}_2$ to be the unit vector in the same direction.

$$\mathbf{q}_2 = \frac{1}{\|\mathbf{a}_2 - (\mathbf{a}_2 \cdot \mathbf{q}_1)\mathbf{q}_1\|}\left(\mathbf{a}_2 - (\mathbf{a}_2 \cdot \mathbf{q}_1)\mathbf{q}_1\right)$$

The remaining $\mathbf{q}_i$'s are defined using the same strategy. We start with $\mathbf{a}_j$ and subtract its components in the directions $\mathbf{q}_1$ to $\mathbf{q}_{j-1}$. Then we normalize to make the resulting vector unit length. Thus

$$\mathbf{q}_3 = \frac{1}{\|\mathbf{a}_3 - (\mathbf{a}_3 \cdot \mathbf{q}_2)\mathbf{q}_2 - (\mathbf{a}_3 \cdot \mathbf{q}_1)\mathbf{q}_1\|}\left(\mathbf{a}_3 - (\mathbf{a}_3 \cdot \mathbf{q}_2)\mathbf{q}_2 - (\mathbf{a}_3 \cdot \mathbf{q}_1)\mathbf{q}_1\right)$$

and so on.

III.2.5 The QR factorization

The Gram–Schmidt procedure can be interpreted as a matrix factorization. If we solve the equations above for $\mathbf{a}_i$ we get

$$\begin{aligned}
\mathbf{a}_1 &= \|\mathbf{a}_1\|\mathbf{q}_1 \\
\mathbf{a}_2 &= (\mathbf{a}_2 \cdot \mathbf{q}_1)\mathbf{q}_1 + \|\mathbf{a}_2 - (\mathbf{a}_2 \cdot \mathbf{q}_1)\mathbf{q}_1\|\mathbf{q}_2 \\
\mathbf{a}_3 &= (\mathbf{a}_3 \cdot \mathbf{q}_1)\mathbf{q}_1 + (\mathbf{a}_3 \cdot \mathbf{q}_2)\mathbf{q}_2 + \|\mathbf{a}_3 - (\mathbf{a}_3 \cdot \mathbf{q}_2)\mathbf{q}_2 - (\mathbf{a}_3 \cdot \mathbf{q}_1)\mathbf{q}_1\|\mathbf{q}_3 \\
&\vdots
\end{aligned}$$

This can be written as a matrix equation:

$$\begin{bmatrix}\mathbf{a}_1 & \mathbf{a}_2 & \mathbf{a}_3 & \cdots & \mathbf{a}_k\end{bmatrix} =$$
$$\begin{bmatrix}\mathbf{q}_1 & \mathbf{q}_2 & \mathbf{q}_3 & \cdots & \mathbf{q}_k\end{bmatrix}\begin{bmatrix}\|\mathbf{a}_1\| & \mathbf{q}_1 \cdot \mathbf{a}_2 & \mathbf{q}_1 \cdot \mathbf{a}_3 & \cdots & \mathbf{q}_1 \cdot \mathbf{a}_k \\ 0 & \|\mathbf{a}_2 - (\mathbf{a}_2 \cdot \mathbf{q}_1)\mathbf{q}_1\| & \mathbf{q}_2 \cdot \mathbf{a}_3 & \cdots & \mathbf{q}_2 \cdot \mathbf{a}_k \\ 0 & 0 & \|\mathbf{a}_3 - (\mathbf{a}_3 \cdot \mathbf{q}_2)\mathbf{q}_2 - (\mathbf{a}_3 \cdot \mathbf{q}_1)\mathbf{q}_1\| & \cdots & \mathbf{q}_3 \cdot \mathbf{a}_k \\ 0 & 0 & 0 & \cdots & \mathbf{q}_4 \cdot \mathbf{a}_k \\ \vdots & \vdots & \vdots & \ddots & \vdots\end{bmatrix}$$

So any matrix A with linearly independent columns can be factored as

$$A = QR$$

where Q has orthonormal columns and the same dimensions as A, and R is an upper triangular matrix with positive entries on the diagonal.

Here is an example. Let

$$\mathbf{a}_1 = \begin{bmatrix}1\\0\\-1\end{bmatrix} \quad \mathbf{a}_2 = \begin{bmatrix}1\\-1\\0\end{bmatrix} \quad A = \begin{bmatrix}\mathbf{a}_1 & \mathbf{a}_2\end{bmatrix} = \begin{bmatrix}1 & 1\\0 & -1\\-1 & 0\end{bmatrix}$$

Then

$$
\begin{aligned}
\|\mathbf{a}_1\| &= \sqrt{2} \\
\mathbf{q}_1 &= \frac{1}{\|\mathbf{a}_1\|}\mathbf{a}_1 = \begin{bmatrix} 1/\sqrt{2} \\ 0 \\ -1/\sqrt{2} \end{bmatrix} \\
\mathbf{q}_1 \cdot \mathbf{a}_2 &= \begin{bmatrix} 1/\sqrt{2} \\ 0 \\ -1/\sqrt{2} \end{bmatrix} \cdot \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix} = 1/\sqrt{2} \\
\mathbf{a}_2 - (\mathbf{q}_1 \cdot \mathbf{a}_2)\mathbf{q}_1 &= \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix} - \begin{bmatrix} 1/2 \\ 0 \\ -1/2 \end{bmatrix} = \begin{bmatrix} 1/2 \\ -1 \\ 1/2 \end{bmatrix} \\
\|\mathbf{a}_2 - (\mathbf{q}_1 \cdot \mathbf{a}_2)\mathbf{q}_1\| &= \sqrt{1/4 + 1 + 1/4} = \sqrt{3/2} \\
\mathbf{q}_2 = \sqrt{2/3} \begin{bmatrix} 1/2 \\ -1 \\ 1/2 \end{bmatrix} &= \begin{bmatrix} 1/\sqrt{6} \\ -\sqrt{2/3} \\ 1/\sqrt{6} \end{bmatrix}
\end{aligned}
$$

With this information we can perform the QR factorization using the formula above.

$$
\begin{bmatrix} 1 & 1 \\ 0 & -1 \\ -1 & 0 \end{bmatrix} = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{6} \\ 0 & -\sqrt{2/3} \\ -1/\sqrt{2} & 1/\sqrt{6} \end{bmatrix} \begin{bmatrix} \sqrt{2} & 1/\sqrt{2} \\ 0 & \sqrt{3/2} \end{bmatrix}
$$

III.2.6 Using MATLAB/Octave for QR

MATLAB/Octave has a built in function for computing the QR factorization. If `A` is a matrix, then `[Q R] = qr(A,0)` produces the factorization above ... almost. Let's try it with the example above:

```
>A=[1 1; 0 -1; -1 0];
>[Q R] = qr(A,0)

Q =

  -0.70711   0.40825
  -0.00000  -0.81650
   0.70711   0.40825

R =

  -1.41421  -0.70711
   0.00000   1.22474
```

This is almost the same factorization as we obtained, except that the diagonal elements of R are not all positive. In fact, comparing this to the factorization we computed by hand, the whole first row of R has been multiplied by -1. To compensate, the first column of Q has also been multiplied by -1. In general, the command `[Q R] = qr(A,0)` will give the factorization above, except that some of the columns of Q and the corresponding rows of R have a sign change. Notice that if we start off with an orthonormal basis and flip the sign of some of the vectors in that basis, the result is still an orthonormal basis. So the difference between our factorization and MATLAB/Octave's is not important.

Running the `qr` command without the second argument, that is `[Q R] = qr(A)`, gives a slightly different factorization if A is $n \times m$ with $n \geq m$. In this factorization Q is a square $n \times n$ orthogonal matrix whose first m columns are identical to the columns of Q in our factorization. The matrix R is padded with rows of zeros at the bottom, so that the product QR is the same as before.

Summary: Math Concepts

- Orthonormal bases
- Orthogonal matrices
- Gram–Schmidt procedure
- QR factorization

Summary: MATLAB/Octave Concepts

- Using `[Q R]=qr(A,0)` to find the QR factorization.

III.3 Complex inner product

III.3.1 Complex numbers (review)

Complex numbers can be thought of as points on the (x, y) plane. The point $\begin{bmatrix} x \\ y \end{bmatrix}$, thought of as a complex number, is written $x + iy$ (or $x + jy$ if you are an electrical engineer).

If $z = x + iy$ then x is called the real part of z and y is called the imaginary part of z.

Complex numbers are added just as if they were vectors in two dimensions. If $z = x + iy$ and $w = s + it$, then

$$z + w = (x + iy) + (s + it) = (x + s) + i(y + t)$$

To multiply two complex numbers, just remember that $i^2 = -1$. So if $z = x + iy$ and $w = s + it$, then

$$zw = (x + iy)(s + it) = xs + i^2yt + iys + ixt = (xs - yt) + i(xt + ys)$$

The modulus of a complex number, denoted $|z|$ is simply the length of the corresponding vector in two dimensions. If $z = x + iy$

$$|z| = |x + iy| = \sqrt{x^2 + y^2}$$

An important property is

$$|zw| = |z||w|$$

The complex conjugate of a complex number z, denoted $\bar{z}$, is the reflection of z across the x axis. Thus $\overline{x + iy} = x - iy$. Thus complex conjugate is obtained by changing all the i's to $-i$'s. We have

$$\overline{zw} = \bar{z}\bar{w}$$

and

$$z\bar{z} = |z|^2$$

This last equality is useful for simplifying fractions of complex numbers by turning the denominator into a real number, since

$$\frac{z}{w} = \frac{z\bar{w}}{|w|^2}$$

For example, to simplify $(1 + i)/(1 - i)$ we can write

$$\frac{1 + i}{1 - i} = \frac{(1 + i)^2}{(1 - i)(1 + i)} = \frac{1 - 1 + 2i}{2} = i$$

A complex number z is real (i.e. the y part in $x + iy$ is zero) whenever $\bar{z} = z$. We also have the following formulas for the real and imaginary part. If $z = x + iy$ then $x = (z + \bar{z})/2$ and $y = (z - \bar{z})/(2i)$

We define the exponential, e^{it}, of a purely imaginary number it to be the number

$$e^{it} = \cos(t) + i\sin(t)$$

lying on the unit circle in the complex plane.

The complex exponential satisfies the familiar rule $e^{i(s+t)} = e^{is}e^{it}$ since by the addition formulas for sine and cosine

$$\begin{aligned} e^{i(s+t)} &= \cos(s+t) + i\sin(s+t) \\ &= \cos(s)\cos(t) - \sin(s)\sin(t) + i(\sin(s)\cos(t) + \cos(s)\sin(t)) \\ &= (\cos(s) + i\sin(s))(\cos(t) + i\sin(t)) \\ &= e^{is}e^{it} \end{aligned}$$

The exponential of a number that has both a real and imaginary part is defined in the natural way.

$$e^{a+ib} = e^a e^{ib} = e^a(\cos(b) + i\sin(b))$$

The derivative of a complex exponential is given by the formula

$$\frac{d}{dt}e^{(a+ib)t} = (a+ib)e^{(a+ib)t}$$

while the anti-derivative, for $(a + ib) \neq 0$ is

$$\int e^{(a+ib)t}dt = \frac{1}{(a+ib)}e^{(a+ib)t} + C$$

If $(a + ib) = 0$ then $e^{(a+ib)t} = e^0 = 1$ so in this case

$$\int e^{(a+ib)t}dt = \int dt = t + C$$

III.3.2 Complex inner product

If $A = [a_{i,j}]$ is a matrix (or a vector) the complex conjugate is the matrix (or vector) obtained by conjugating each entry. Thus

$$\overline{A} = [\overline{a}_{i,j}].$$

The product rule for complex conjugation extends to matrices and we have

$$\overline{AB} = \bar{A}\bar{B}$$

The complex inner product of two vectors $\mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}$ and $\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix}$ is defined by

$$\langle \mathbf{z}, \mathbf{w} \rangle = \overline{\mathbf{w}}^T \mathbf{z} = \sum_{i=1}^{n} \overline{w}_i z_i$$

With this definition the norm of $\mathbf{z}$ is always positive since

$$\langle \mathbf{z}, \mathbf{z} \rangle = \|\mathbf{z}\|^2 = \sum_{i=1}^{n} |z_i|^2$$

For complex matrices and vectors we have to modify the rule for bringing a matrix to the other side of an inner product.

$$\langle \mathbf{z}, A\mathbf{w} \rangle = \langle \overline{A}^T \mathbf{z}, \mathbf{w} \rangle$$

This leads to the definition of the adjoint of a matrix

$$A^* = \overline{A}^T.$$

(In physics you will also see the notation $A^\dagger$.) With this notation $\langle \mathbf{z}, A\mathbf{w} \rangle = \langle A^* \mathbf{z}, \mathbf{w} \rangle$.

The complex analogue of an orthogonal matrix is called a unitary matrix. A unitary matrix U is a square matrix satisfying

$$U^*U = UU^* = I.$$

Notice that a unitary matrix with real entries is an orthogonal matrix since in that case $U^* = U^T$. The columns of a unitary matrix form an orthonormal basis (with respect to the complex inner product.)

MATLAB/Octave deals seamlessly with complex matrices and vectors. Complex numbers can be entered like this

```
>z= 1 + 2i

z =  1 + 2i
```

There is a slight danger here in that if `i` has be defined to be something else (*e.g.* `i =16`) then `z=i` would set `z` to be `16`. You could use `z=1i` to get the desired result, or use the alternative syntax

```
>z= complex(0,1)

z =  0 + 1i
```

The functions `real(z)`, `imag(z)`, `conj(z)`, `abs(z)` compute the real part, imaginary part, conjugate and modulus of `z`.

The function `exp(z)` computes the complex exponential if `z` is complex.

If a matrix `A` has complex entries then `A'` is *not* the transpose, but the adjoint (conjugate transpose).

```
>z = [1; 1i]

z =

   1 + 0i
   0 + 1i

z'

ans =

   1 - 0i   0 - 1i
```

Thus the square of the norm of a complex vector is given by

```
>z'*z

ans =  2
```

This gives the same answer as

```
>norm(z)^2

ans =  2.0000
```

Summary: Math Concepts

- Complex numbers: addition, complex conjugate, modulus
- Complex exponential, addition formula, differentiation and integration
- Complex inner product: definition, norm of a complex vector
- Matrix adjoint, moving a matrix from one side of inner product to the other
- Unitary matrices

Summary: MATLAB/Octave Concepts

- Entering complex numbers
- `real(z)`, `imag(z)`, `conj(z)`, `abs(z)`
- `exp(z)`
- `A'` for complex matrices.

III.4 Fourier series

Fourier's theorem states that any (sufficiently nice) function $f(x)$ defined for x in the interval $[0, 1]$ can be expanded in a Fourier series:

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} (a_n \cos(2\pi n x) + b_n \sin(2\pi n x))$$

We wish to understand this expansion from the point of view of linear algebra. In particular, given $f(x)$, how can we find the coefficients a_n and b_n? It turns out that we can interpret this expansion as an expansion of a vector in an orthonormal basis.

III.4.1 Complex form

To start we will rewrite this expansion in the complex exponential form

$$f(x) = \sum_{n=-\infty}^{\infty} c_n e^{i2\pi n x}.$$

The complex exponential form simplifies some of the computations.

Recall that

$$e^{it} = \cos(t) + i \sin(t).$$

Therefore

$$\begin{aligned} \cos(t) &= \frac{e^{it} + e^{-it}}{2} \\ \sin(t) &= \frac{e^{it} - e^{-it}}{2i}. \end{aligned}$$

To obtain the complex exponential form of the Fourier series we simply substitute these expressions into the original series. This gives

$$\begin{aligned} f(x) &= \frac{a_0}{2} + \sum_{n=1}^{\infty} \left(\frac{a_n}{2} \left(e^{i2\pi n x} + e^{-i2\pi n x} \right) + \frac{b_n}{2i} \left(e^{i2\pi n x} - e^{-i2\pi n x} \right) \right) \\ &= \frac{a_0}{2} + \sum_{n=1}^{\infty} \left(\left(\frac{a_n}{2} + \frac{b_n}{2i} \right) e^{i2\pi n x} + \left(\frac{a_n}{2} - \frac{b_n}{2i} \right) e^{-i2\pi n x} \right) \\ &= \sum_{n=-\infty}^{\infty} c_n e^{i2\pi n x} \end{aligned}$$

where

$$c_0 = \frac{a_0}{2}$$
$$c_n = \frac{a_n}{2} + \frac{b_n}{2i} \quad \text{for} \quad n > 0$$
$$c_n = \frac{a_{-n}}{2} - \frac{b_{-n}}{2i} \quad \text{for} \quad n < 0.$$

This complex form of the Fourier series is completely equivalent to the original series. Given the a_n's and b_n's we can compute the c_n's using the formula above, and conversely, given the c_n's we can solve for

$$a_0 = 2c_0$$
$$a_n = c_n + c_{-n} \quad \text{for} \quad n > 0$$
$$b_n = ic_n - ic_{-n} \quad \text{for} \quad n > 0$$

III.4.2 Inner product for a space of functions

We will consider the space of complex valued functions $f(x)$ on the interval $[0,1]$ that obey

$$\int_0^1 |f(x)|^2 dx < \infty$$

This space is called $L^2([0,1])$ and is an example of a Hilbert space. If f and g are two functions in this space, then we define the inner product to be

$$\langle f, g \rangle = \int_0^1 \overline{f}(x) g(x) dx$$

Here $\overline{f}(x)$ denotes the complex conjugate of f.

III.4.3 An orthonormal basis

Now we will show that the complex exponential functions appearing in the Fourier expansion are an orthonormal set. Let

$$e_n(x) = e^{i2\pi nx}$$

for $n = 0, \pm 1, \pm 2, \ldots$. We must compute $\langle e_n, e_m \rangle$. Since $\overline{e}_n(x) = e^{-i2\pi nx}$, we find

$$\langle e_n, e_m \rangle = \int_0^1 e^{-i2\pi nx} e^{i2\pi mx} dx$$
$$= \int_0^1 e^{i2\pi(m-n)x} dx$$

If $n = m$ then $e^{i2\pi(m-n)x} = 1$ so the integral equals 1. On the other hand if $n \neq m$ then $e^{i2\pi(m-n)x}$ has an anti-derivative $e^{i2\pi(m-n)x}/2\pi(m-n)$ that takes on the same value (namely $1/2\pi(m-n)$) at both endpoints $x = 0$ and $x = 1$. Hence the integral is zero in this case.

Thus the functions $\{e_n(x)\}$ form an orthonormal set. In fact, they are a basis for our space of functions. The fact that they span the space, i.e., that every function can be written as an infinite linear combination of the e_n's, is more difficult to show. (For a start, it would require a discussion of what it means for an infinite linear combination to converge!)

However, if you accept the fact that the e_n's do indeed form an infinite basis, then it is very easy to compute the coefficients. Starting with

$$f(x) = \sum_{n=-\infty}^{\infty} c_n e_n(x)$$

we simply take the inner product of both sides with e_m. The only term in the infinite sum that survives is the one with $n = m$. Thus

$$\langle e_m, f\rangle = \sum_{n=-\infty}^{\infty} c_n \langle e_m, e_n\rangle = c_m$$

and we obtain the formula

$$c_m = \int_0^1 e^{-i2\pi mx} f(x)dx$$

III.4.4 An example

Let's compute the Fourier coefficients for the square wave function

$$f(x) = \begin{cases} 1 & \text{if } 0 \le x \le 1/2 \\ -1 & \text{if } 1/2 < x \le 1 \end{cases}$$

If $n = 0$ then $e^{-i2\pi nx} = e^0 = 1$ so c_0 is simply the integral of f.

$$c_0 = \int_0^1 f(x)dx = \int_0^{1/2} 1dx - \int_{1/2}^1 1dx = 0$$

Otherwise, we have

$$\begin{aligned} c_n &= \int_0^1 e^{-i2\pi nx} f(x)dx \\ &= \int_0^{1/2} e^{-i2\pi nx} dx - \int_{1/2}^1 e^{-i2\pi nx} dx \\ &= \frac{e^{-i2\pi nx}}{-i2\pi n}\Big|_{x=0}^{x=1/2} - \frac{e^{-i2\pi nx}}{-i2\pi n}\Big|_{x=1/2}^{x=1} \\ &= \frac{2-2e^{i\pi n}}{2\pi i n} \\ &= \begin{cases} 0 & \text{if} \quad n \text{ is even} \\ 2/i\pi n & \text{if} \quad n \text{ is odd} \end{cases} \end{aligned}$$

Thus we conclude that

$$f(x) = \sum_{\substack{n=-\infty \\ n \text{ odd}}}^{\infty} \frac{2}{i\pi n} e^{i2\pi nx}$$

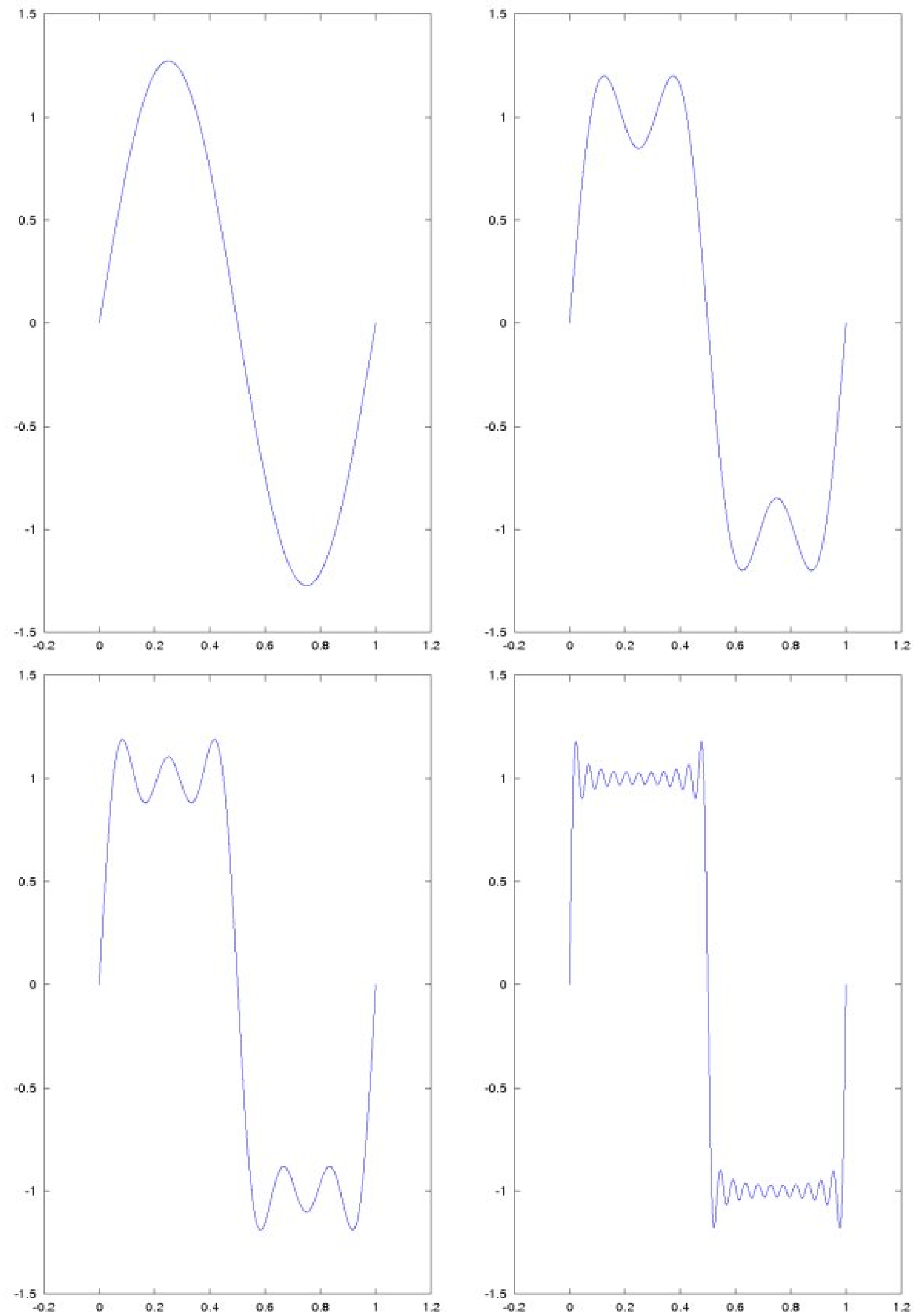
To see how well this series is approximating $f(x)$ we go back to the real form of the series. Using $a_n = c_n + c_{-n}$ and $b_n = ic_n - ic_{-n}$ we find that $a_n = 0$ for all n, $b_n = 0$ for n even and $b_n = 4/\pi n$ for n odd. Thus

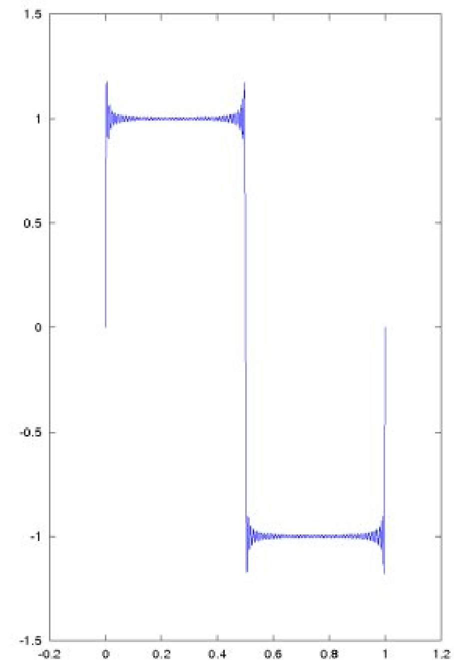
$$f(x) = \sum_{\substack{n=1 \\ n \text{ odd}}}^{\infty} \frac{4}{\pi n} \sin(2\pi nx) = \sum_{n=0}^{\infty} \frac{4}{\pi(2n+1)} \sin(2\pi(2n+1)x)$$

We can use MATLAB/Octave to see how well this series is converging. The file `ftdemo1.m` contains a function that take an integer N as an argument and plots the sum of the first $2N+1$ terms in the Fourier series above. Here is a listing:

```
function ftdemo1(N)

        X=linspace(0,1,1000);
        F=zeros(1,1000);

        for n=[0:N]
                F = F + 4*sin(2*pi*(2*n+1)*X)/(pi*(2*n+1));
        end

        plot(X,F)

end
```

Here are the outputs for $N = 0, 1, 2, 10, 50$:


1.5
1
0.5
0
-0.5
-1
-1.5
-0.2
0
0.2
0.4
0.6
0.8
1
1.2



III.4.5 Parseval's formula

If $\mathbf{v}_1$, $\mathbf{v}_2$, ..., $\mathbf{v}_n$ is an orthonormal basis in a finite dimensional vector space and the vector $\mathbf{v}$ has the expansion

$$\mathbf{v} = c_1\mathbf{v}_1 + \cdots + c_n\mathbf{v}_n = \sum_{i=1}^{n} c_i\mathbf{v}_i$$

then, taking the inner product of $\mathbf{v}$ with itself, and using the fact that the basis is orthonormal, we obtain

$$\langle \mathbf{v}, \mathbf{v} \rangle = \sum_{i=1}^{n}\sum_{j=1}^{n} \overline{c}_i c_j \langle \mathbf{v}_i, \mathbf{v}_j \rangle = \sum_{i=1}^{n} |c_i|^2$$

The same formula is true in Hilbert space. If

$$f(x) = \sum_{n=-\infty}^{\infty} c_n e_n(x)$$

Then

$$\int_0^1 |f(x)|^2 dx = \langle f, f \rangle = \sum_{n=-\infty}^{\infty} |c_n|^2$$

In the example above, we have $\langle f, f \rangle = \int_0^1 1dx = 1$ so we obtain

$$1 = \sum_{\substack{n=-\infty \\ n \text{ odd}}}^{n=\infty} \frac{4}{\pi^2 n^2} = 2\sum_{\substack{n=0 \\ n \text{ odd}}}^{n=\infty} \frac{4}{\pi^2 n^2} = \frac{8}{\pi^2}\sum_{n=0}^{\infty} \frac{1}{(2n+1)^2}$$

or

$$\sum_{n=0}^{\infty} \frac{1}{(2n+1)^2} = \frac{\pi^2}{8}$$

Summary: Math Concepts

- Fourier series: real and complex form
- Inner product for functions
- Interpretation of Fourier series as expansion in orthonormal basis
- Computation of Fourier series coefficients
- Parseval's formula

Summary: MATLAB/Octave Concepts

- Plotting partial sums of Fourier series

III.5 The Discrete Fourier Transform

III.5.1 Definition

Suppose that we don't know the function f everywhere on the interval, but just at N discrete points $0, 1/N, 2/N, \ldots, (N-1)/N$. Define $f_j = f(j/N)$. Then we can write down an approximation for the Fourier coefficient c_k by using the Riemann sum in place of the integral. The resulting formula is the definition of the discrete Fourier transform.

$$c_k = \frac{1}{N}\sum_{j=0}^{N-1} e^{-i2\pi kj/N} f_j$$

The first thing to notice about this definition is that although the formula makes sense for all k, the c_k's start repeating themselves after a while. In fact $c_{k+N} = c_k$ for all k. This follows from the fact that $e^{-i2\pi j} = 1$ which implies that

$$e^{-i2\pi(k+N)j/N} = e^{-i2\pi kj/N} e^{-i2\pi j} = e^{-i2\pi kj/N},$$

so the formulas for c_k and c_{k+N} are the same. So we only need to compute $c_0, \ldots, c_{N-1}$.

Next, notice that the transformation that sends the vector $[f_0, \ldots, f_{N-1}]$ to the vector $[c_0, \ldots, c_{N-1}]$ is a linear transformation, given by multiplication by the matrix $F = [F_{k,j}]$ with $F_{k,j} = \frac{1}{N} e^{-i2\pi kj/N}$. If we define $\omega_N = e^{-i2\pi/N}$ then the matrix has the form

$$F = \frac{1}{N}\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega_N & \omega_N^2 & \cdots & \omega_N^{N-1} \\ 1 & \omega_N^2 & \omega_N^4 & \cdots & \omega_N^{2(N-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \omega_N^{N-1} & \omega_N^{2(N-1)} & \cdots & \omega_N^{(N-1)(N-1)} \end{bmatrix}$$

The inverse of F is given by

$$F^{-1} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \overline{\omega_N} & \overline{\omega_N}^2 & \cdots & \overline{\omega_N}^{N-1} \\ 1 & \overline{\omega_N}^2 & \overline{\omega_N}^4 & \cdots & \overline{\omega_N}^{2(N-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \overline{\omega_N}^{N-1} & \overline{\omega_N}^{2(N-1)} & \cdots & \overline{\omega_N}^{(N-1)(N-1)} \end{bmatrix}$$

where $\overline{\omega_N}$ is the complex conjugate of ω_N given by

$$\overline{\omega_N} = e^{i2\pi/N} = \frac{1}{\omega_N}.$$

To see that this is the inverse, notice that ${\omega_N}^N = e^{-i2\pi} = 1$ so that ${\omega_N}^{kN} = 1$ as well, for any integer k. The points ${\omega_N}^k$ for $k = 0, 1, 2, \ldots, N-1$ are equally spaced points on the unit circle in the complex plane, repeated periodically. So when k is not an integer multiple of N the $\omega_N^k \neq 1$

Now recall the formula for the sum of a geometric series:

$$(1 + z + z^2 + \cdots + z^{N-1}) = \frac{1 - z^N}{1 - z}.$$

When $z = \omega_N^k$ the numerator vanishes, and as long as k is not a multiple of N, the denominator doesn't, so the sum is zero. On the other hand, when k is a multiple of N so that $z = \omega_N^k = 1$ then the geometric sum adds up to N.

If you multiply the matrices above, each term in the product has the form of a geometric sum (try it!) and the formula shows that the result is the identity matrix.

Notice that $\tilde{F} = \sqrt{N}F$ is a unitary matrix ($\tilde{F}^{-1} = \tilde{F}^*$). Recall that unitary matrices preserve the length of complex vectors. This implies that the lengths of the vectors $\mathbf{f} = [f_0, f_1, \ldots, f_{N-1}]$ and $\mathbf{c} = [c_0, c_1, \ldots, c_{N-1}]$ are related by

$$\|\mathbf{c}\|^2 = \frac{1}{N}\|\mathbf{f}\|^2$$

or

$$\sum_{k=0}^{N-1} |c_k|^2 = \frac{1}{N}\sum_{k=0}^{N-1} |f_k|^2$$

This is the discrete version of Parseval's formula.

III.5.2 The Fast Fourier transform

Multiplying an $N \times N$ matrix with a vector of length N normally requires N^2 multiplications, since each entry of the product requires N, and there are N entries. In turns out that the discrete Fourier transform, that is, multiplication by the matrix F, can be carried out using only $N\log_2(N)$ multiplications (at least if N is a power of 2). The algorithm that achieves this is called the Fast Fourier Transform, or FFT. This represents a tremendous saving in time: calculations that would require weeks of computer time can be carried out in seconds.

The basic idea of the FFT is to break the sum defining the Fourier coefficients c_k into a sum of the even terms and a sum of the odd terms. Each of these turns out to be (up to a factor we can compute) a discrete Fourier transform of half the length. This idea is then applied recursively. Starting with $N = 2^n$ and halving the size of the Fourier transform at each step, it takes $n = \log_2(N)$ steps to arrive at Fourier transforms of length 1. This is where the $\log_2(N)$ comes in.

To simplify the notation, we will ignore the factor of $1/N$ in the definition of the discrete Fourier transform (so one should divide by N at the end of the calculation.) We now also assume that

$$N = 2^n$$

so that we can divide N by 2 repeatedly. The basic formula, splitting the sum for c_k into a sum over odd and even j's is

$$\begin{aligned}
Nc_k &= \sum_{j=0}^{N-1} e^{-i2\pi kj/N} f_j \\
&= \sum_{\substack{j=0\\ j \text{ even}}}^{N-1} e^{-i2\pi kj/N} f_j + \sum_{\substack{j=0\\ j \text{ odd}}}^{N-1} e^{-i2\pi kj/N} f_j \\
&= \sum_{j=0}^{N/2-1} e^{-i2\pi k2j/N} f_{2j} + \sum_{j=0}^{N/2-1} e^{-i2\pi k(2j+1)/N} f_{2j+1} \\
&= \sum_{j=0}^{N/2-1} e^{-i2\pi kj/(N/2)} f_{2j} + e^{-i2\pi k/N} \sum_{j=0}^{N/2-1} e^{-i2\pi kj/(N/2)} f_{2j+1}
\end{aligned}$$

Notice that the two sums on the right are discrete Fourier transforms of length $N/2$.

To continue, it is useful to write the integers j in base 2. Lets assume that $N = 2^3 = 8$. Once you understand this case, the general case $N = 2^n$ will be easy. Recall that

$$\begin{aligned}
0 &= 000 \quad \text{(base 2)} \\
1 &= 001 \quad \text{(base 2)} \\
2 &= 010 \quad \text{(base 2)} \\
3 &= 011 \quad \text{(base 2)} \\
4 &= 100 \quad \text{(base 2)} \\
5 &= 101 \quad \text{(base 2)} \\
6 &= 110 \quad \text{(base 2)} \\
7 &= 111 \quad \text{(base 2)}
\end{aligned}$$

The even j's are the ones whose binary expansions have the form $**0$, while the odd j's have binary expansions of the form $**1$.

For any pattern of bits like $**0$, I will use the notation $F^{<pattern>}$ to denote the discrete Fourier transform[1] where the input data is given by all the f_j's whose j's have binary expansion fitting the pattern. Here are some examples. To start, $F_k^{***} = Nc_k$ is the original discrete Fourier transform, since every j fits the pattern $***$. In this example k ranges over $0, \ldots, 7$, that is, the values start repeating after that.

[1] Up to the factor of the number of points in the transform.

Only even j's fit the pattern $**0$, so F^{**0} is the discrete Fourier transform of the even j's given by

$$F_k^{**0} = \sum_{j=0}^{N/2-1} e^{-i2\pi kj/(N/2)} f_{2j}.$$

Here k runs from 0 to 3 before the values start repeating. Similarly, F^{*00} is a transform of length $N/4 = 2$ given by

$$F_k^{*00} = \sum_{j=0}^{N/4-1} e^{-i2\pi kj/(N/4)} f_{4j}.$$

In this case $k = 0, 1$ and then the values repeat. Finally, the only j matching the pattern 010 is $j = 2$, so F_k^{010} is a transform of length one term given by

$$F_k^{010} = e^{-i2\pi k \cdot 2/(N/8)} f_2 = f_2.$$

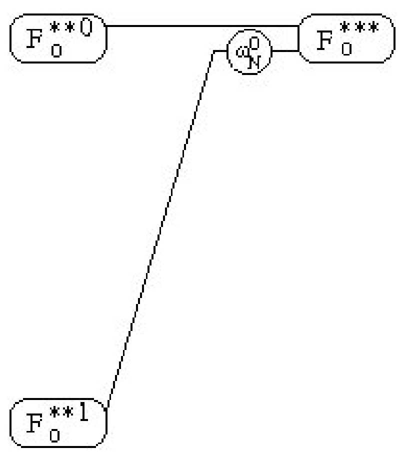
Here we used that $N/8 = 1$ so that $e^{-i2\pi k\cdot 2/(N/8)} = e^{-i2\pi k \cdot 2} = 1$.

With this notation, the basic even–odd formula can be written

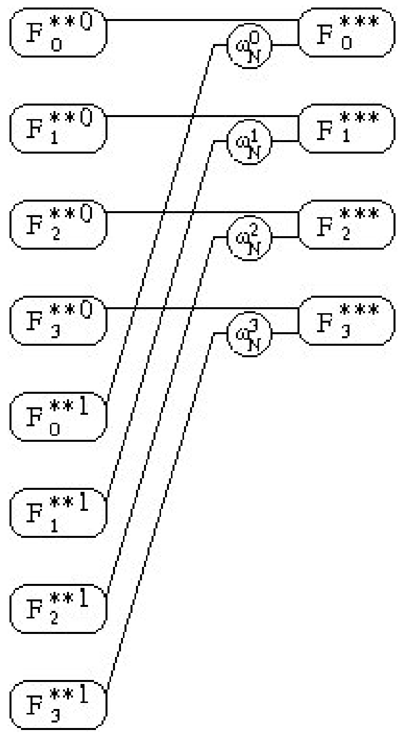
$$F_k^{***} = F_k^{**0} + \omega_N^k F_k^{**1}.$$

Recall that $\omega_N = e^{-i2\pi/N}$.

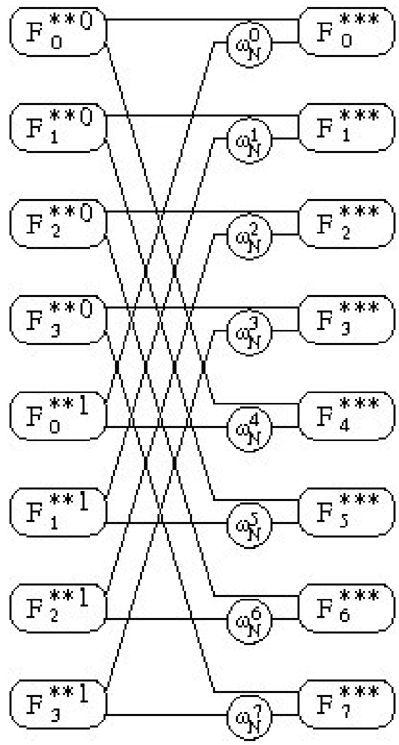
Lets look at this equation when $k = 0$. We will represent the formula by the following diagram.



This diagram means that F_0^{***} is obtained by adding F_0^{**0} to $\omega_N^0 F_0^{**1}$. (Of course $\omega_N^0 = 1$ so we could omit it.) Now lets add the diagrams for $k = 1, 2, 3$.



Now when we get to $k = 4$, we recall that F^{**0} and F^{**1} are discrete transforms of length $N/2 = 4$. Therefore, by periodicity $F_4^{**0} = F_0^{**0}$, $F_5^{**0} = F_1^{**0}$, and so on. So in the formula $F_4^{***} = F_4^{**0} + \omega_N^4 F_4^{**1}$ we may replace F_4^{**0} and F_4^{**1} with F_0^{**0} and F_0^{**1} respectively. Making such replacements, we complete the first part of the diagram as follows.



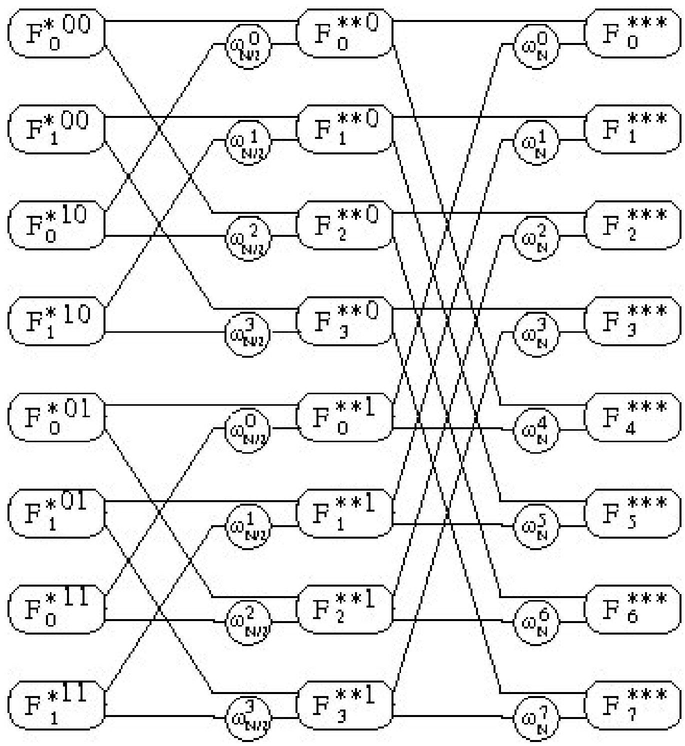
To move to the next level we analyze the discrete Fourier transforms on the left of this diagram in the same way. This time we use the basic formula for the transform of length $N/2$, namely

$$F_k^{**0} = F_k^{*00} + \omega_{N/2}^k F_k^{*10}$$

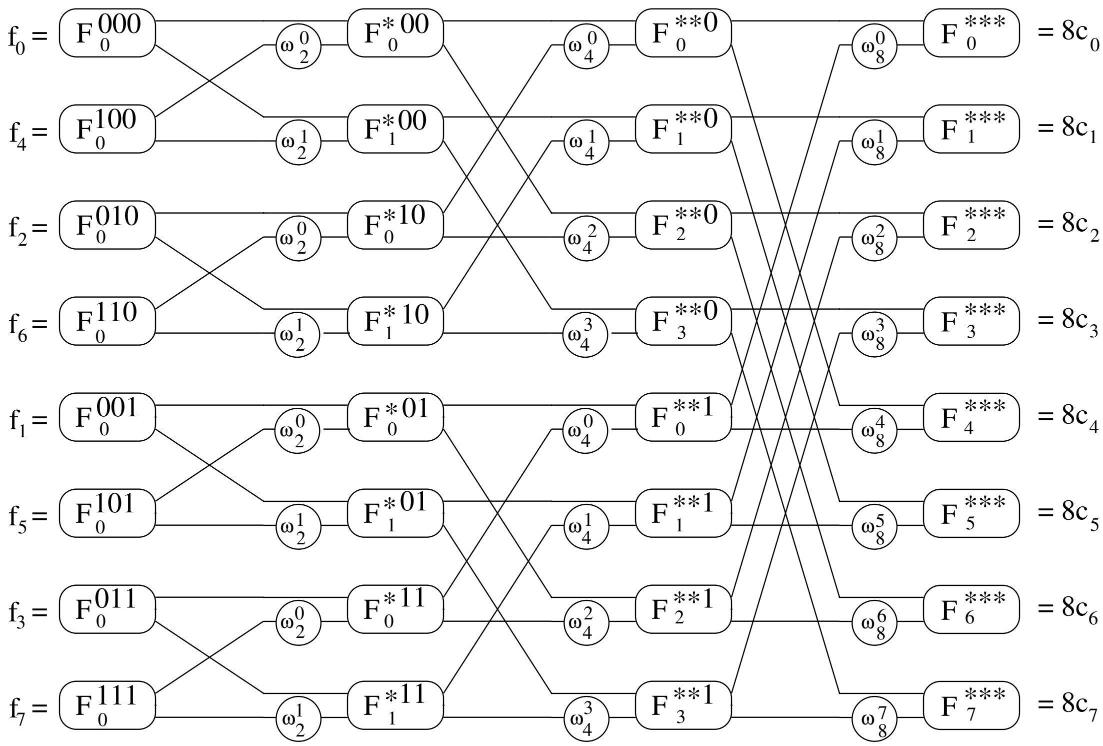
and

$$F_k^{**1} = F_k^{*01} + \omega_{N/2}^k F_k^{*11}.$$

The resulting diagram shows how to go from the length two transforms to the final transform on the right.



Now we go down one more level. Each transform of length two can be constructed from transforms of length one, i.e., from the original data in some order. We complete the diagram as follows. Here we have inserted the value $N = 8$.

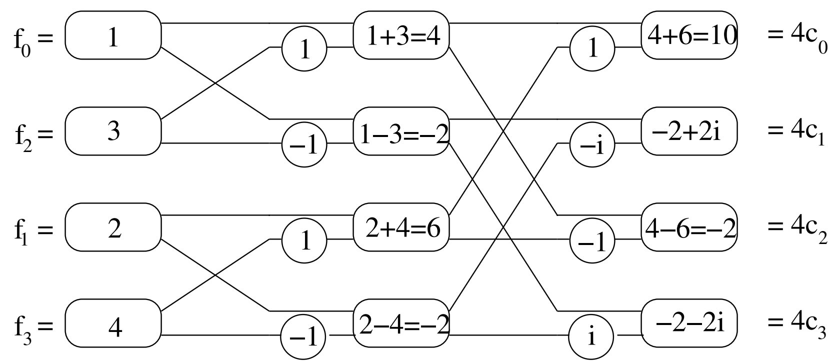


Notice that the f_j's on the left of the diagram are in *bit reversed* order. In other words, if we reverse the order of the bits in the binary expansion of the j's, the resulting numbers are ordered from 0 (000) to 7 (111).

Now we can describe the algorithm for the fast Fourier transform. Starting with the original data $[f_0, \ldots, f_7]$ we arrange the values in bit reversed order. Then we combine them pairwise, as indicated by the left side of the diagram, to form the transforms of length 2. To do this we we need to compute $\omega_2 = e^{-i\pi} = -1$. Next we combine the transforms of length 2 according to the middle part of the diagram to form the transforms of length 4. Here we use that $\omega_4 = e^{-i\pi/2} = -i$. Finally we combine the transforms of length 4 to obtain the transform of length 8. Here we need $\omega_8 = e^{-i\pi/4} = 2^{-1/2} - i2^{-1/2}$.

The algorithm for values of N other than 8 is entirely analogous. For $N = 2$ or 4 we stop at the first or second stage. For larger values of $N = 2^n$ we simply add more stages. How many multiplications do we need to do? Well there are $N = 2^n$ multiplications per stage of the algorithm (one for each circle on the diagram), and there are $n = \log_2(N)$ stages. So the number of multiplications is $2^n n = N \log_2(N)$

As an example let us compute the discrete Fourier transform with $N = 4$ of the data $[f_0, f_1, f_2, f_3] = [1, 2, 3, 4]$. First we compute the bit reversed order of $0 = (00), 1 = (01), 2 = (10), 3 = (11)$ to be $(00) = 0, (10) = 2, (01) = 1, (11) = 3$. We then do the rest of the computation right on the diagram as follows.



Summary: Math Concepts

- Discrete Fourier Transform definition
- Discrete version of Parseval's formula
- Fast Fourier Transform

Summary: MATLAB/Octave Concepts