FFTs in Graphics and Vision

Moving Dot Products

Outline

Review

Moving Dot Products:

- One-Dimensional (Continuous)
- One-Dimensional (Discrete)
- Higher-Dimensional
- Computational Complexity

Representations

A <u>representation</u> of a group G on a vector space V is a map ρ that sends every element in G to an invertible linear transformation on V, satisfying:

$$\rho(g \cdot h) = \rho(g) \cdot \rho(h)$$

for all $g, h \in G$.

Sub-Representation

Given a representation, ρ, of a group, G, on a vector space, V, if there exists a subspace $W \subset V$, such that the representation fixes W:

$$\rho_g w \in W \qquad \forall g \in G \text{ and } w \in W$$

then we say that W is a <u>sub-representation</u> of V.

Irreducible Representations

Given a representation, ρ, of a group, G, on a vector space, V, the representation is said to be irreducible if the only subspaces of V that are sub-representations are:

$$W = V \quad \text{and} \quad W = \emptyset$$

Schur's Lemma (Corollary)

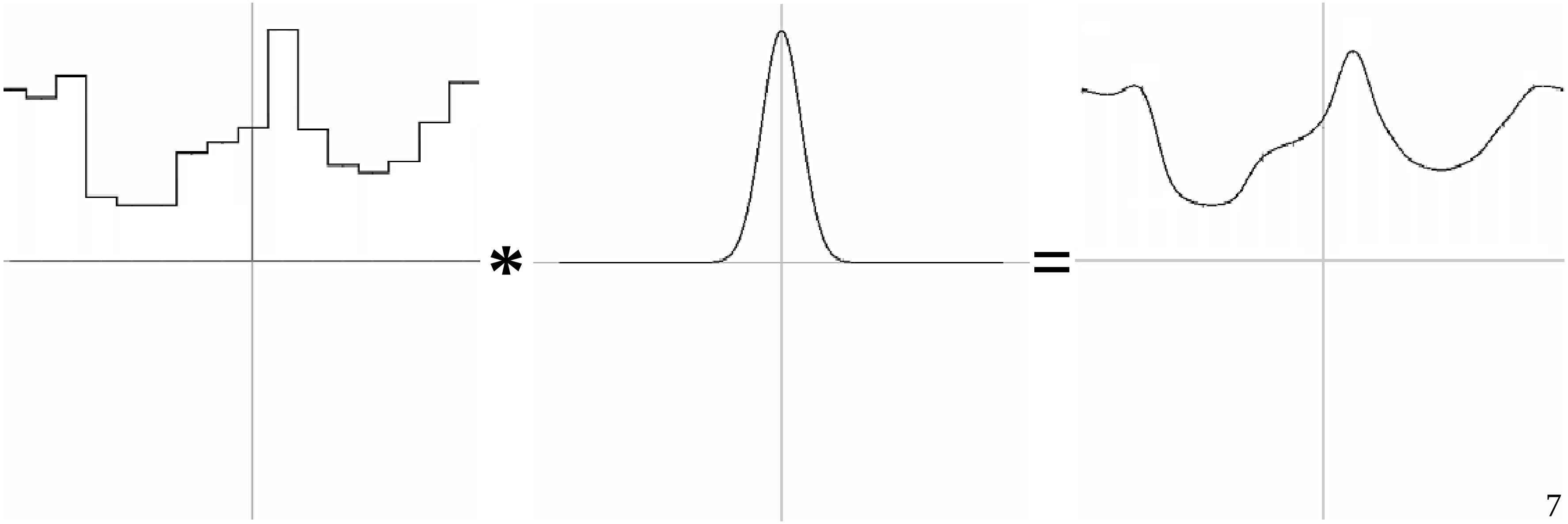
If ρ is an irreducible, unitary, representation of a commutative group G onto a complex vector space V, then:

- V must be one-dimensional
- For any $g \in G$, $\rho(g)$ is a unit-norm complex number

Why do we care?

In signal/image/voxel processing, we are often interested in applying a filter to some initial data.

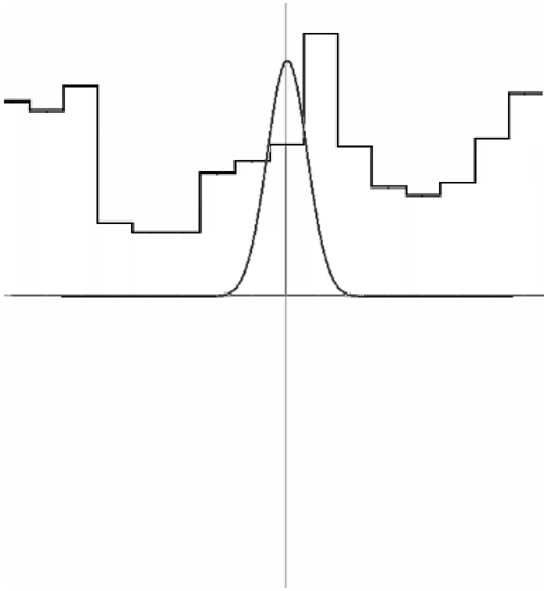
E.g. Smoothing:

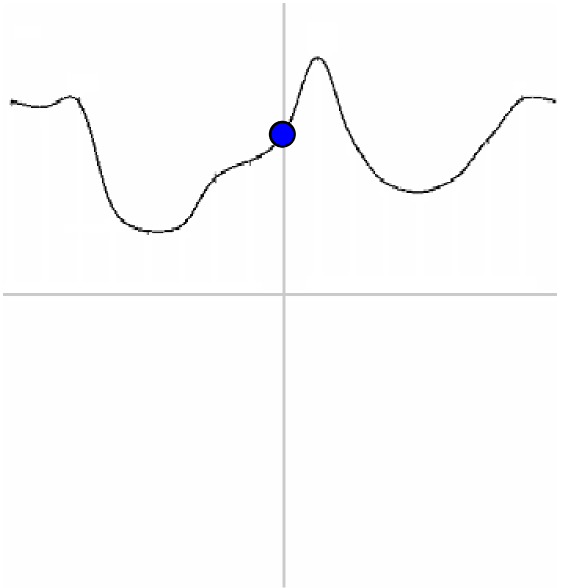


Why do we care?

In signal/image/voxel processing, we are often interested in applying a filter to some initial data.

E.g. Smoothing:

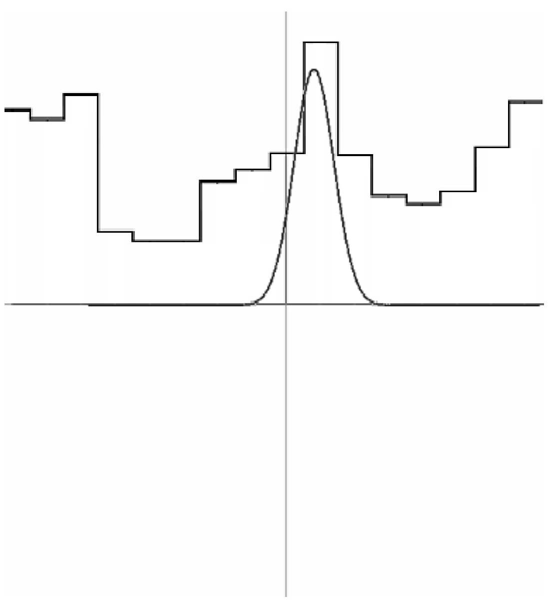


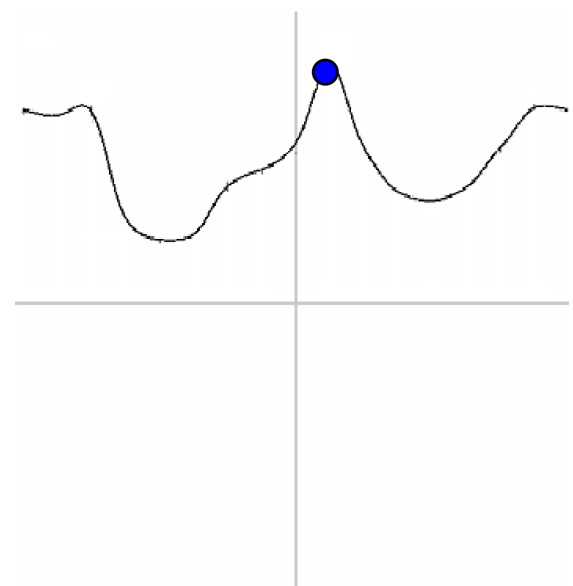


Why do we care?

In signal/image/voxel processing, we are often interested in applying a filter to some initial data.

E.g. Smoothing:

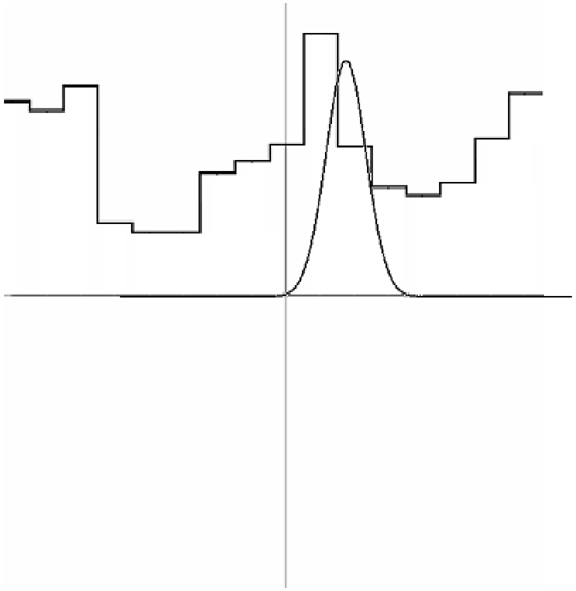


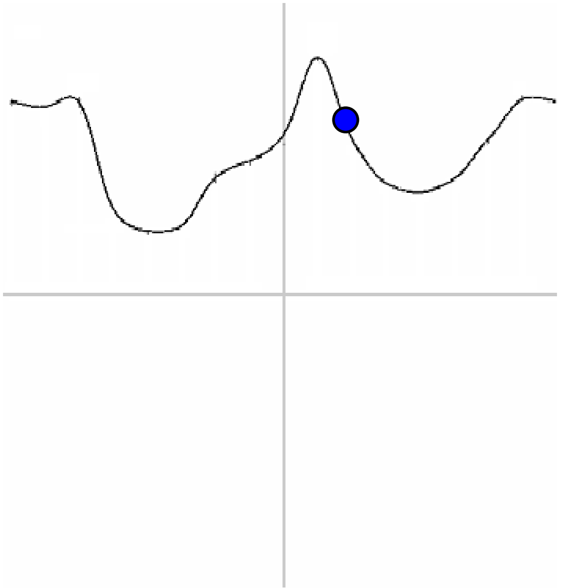


Why do we care?

In signal/image/voxel processing, we are often interested in applying a filter to some initial data.

E.g. Smoothing:

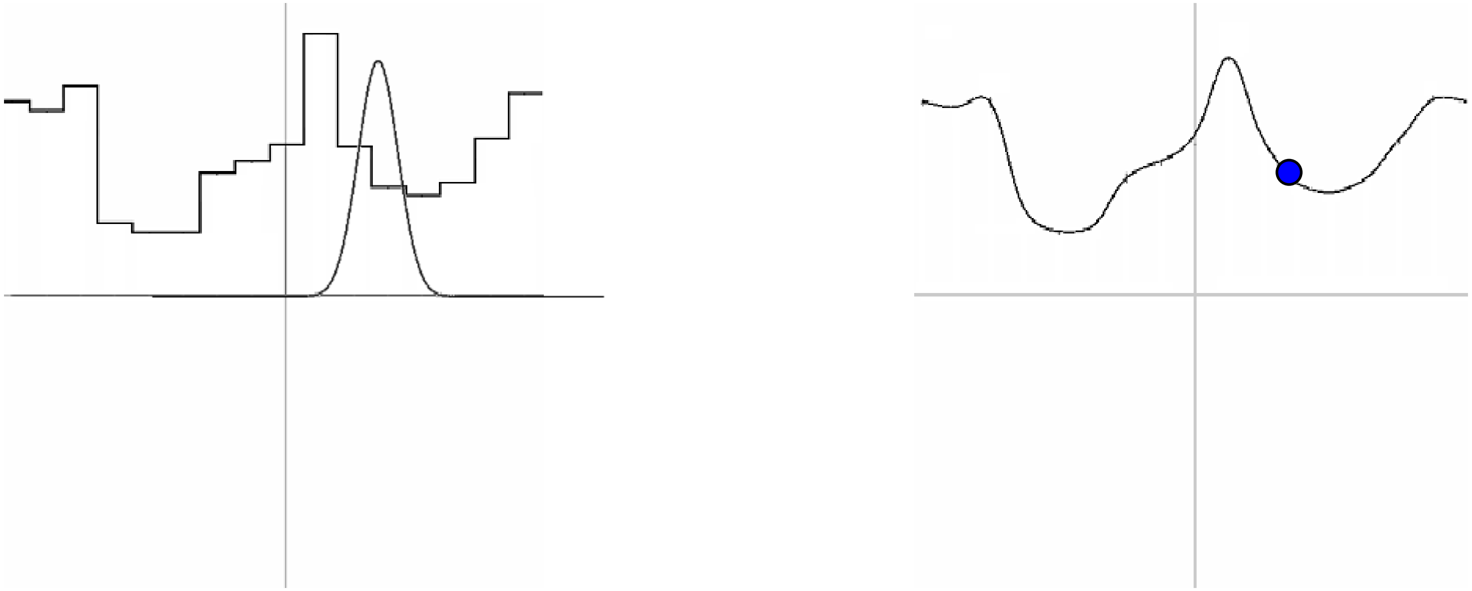




Why do we care?

In signal/image/voxel processing, we are often interested in applying a filter to some initial data.

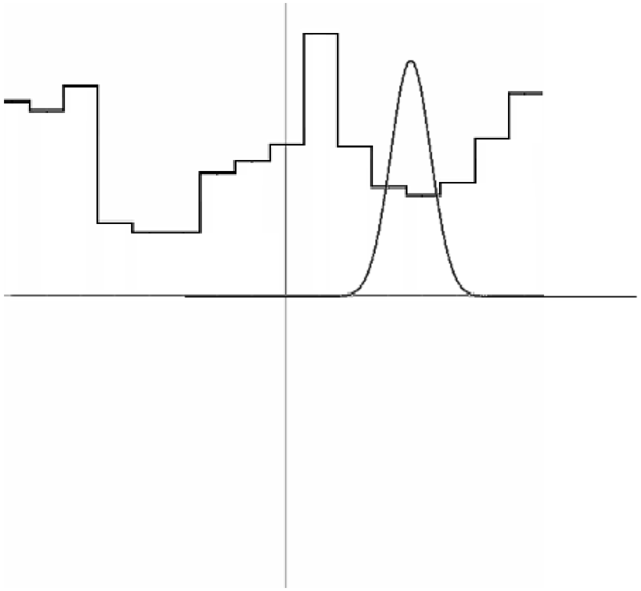
E.g. Smoothing:

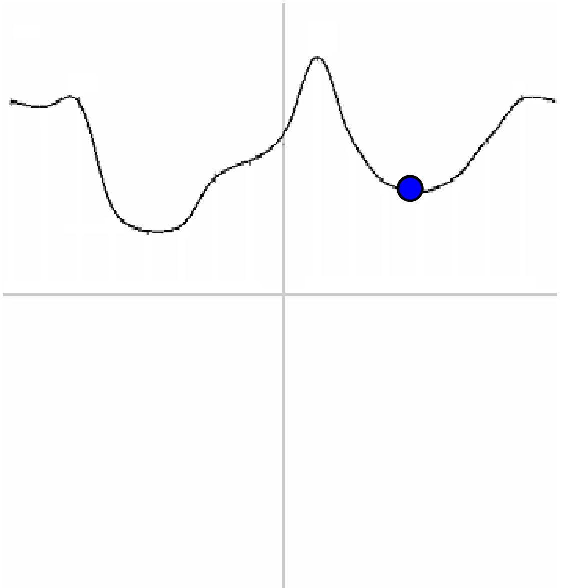


Why do we care?

In signal/image/voxel processing, we are often interested in applying a filter to some initial data.

E.g. Smoothing:

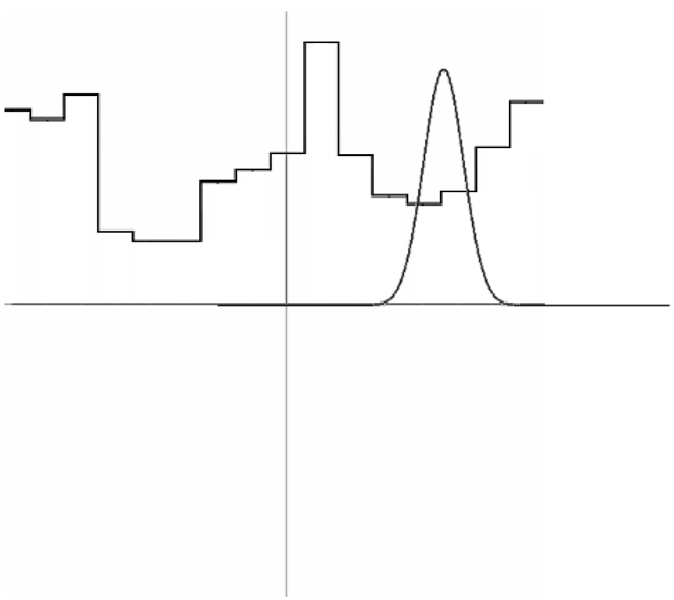


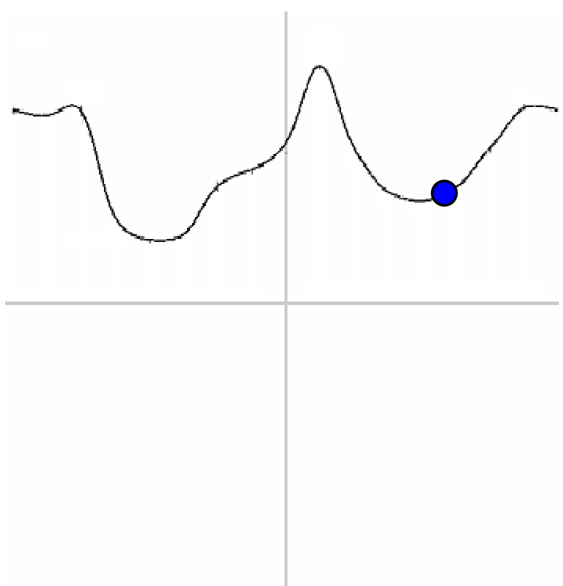


Why do we care?

In signal/image/voxel processing, we are often interested in applying a filter to some initial data.

E.g. Smoothing:

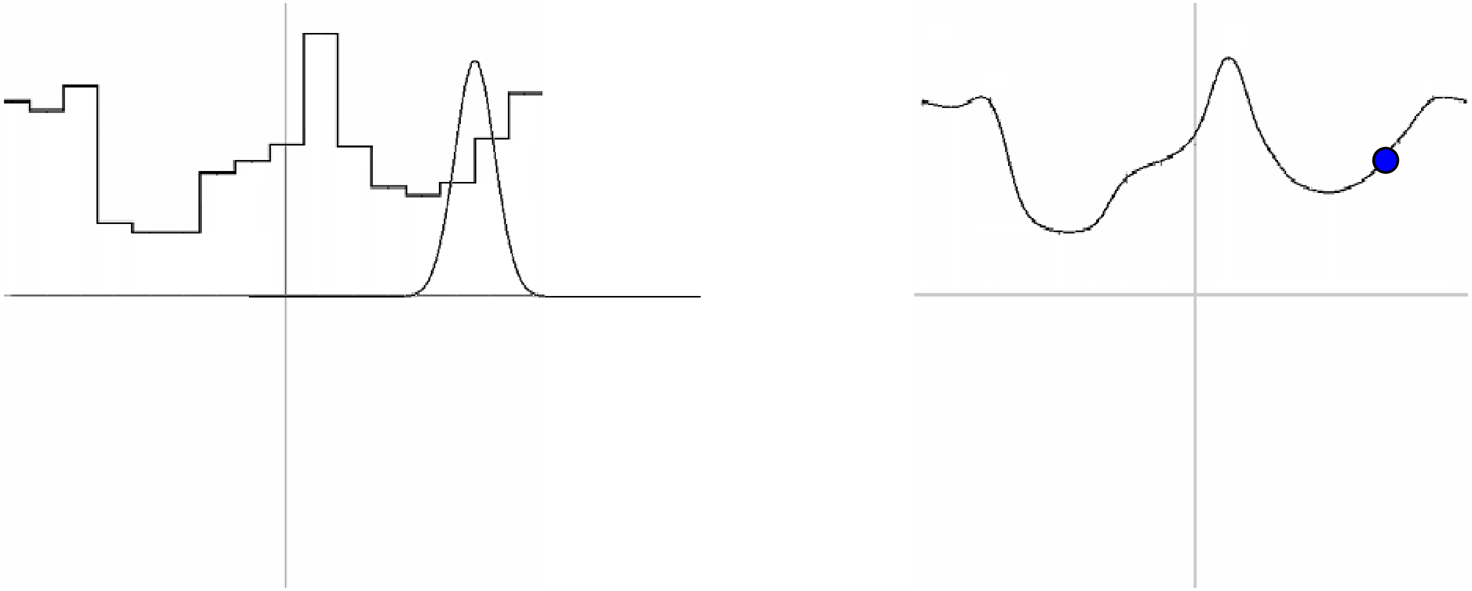




Why do we care?

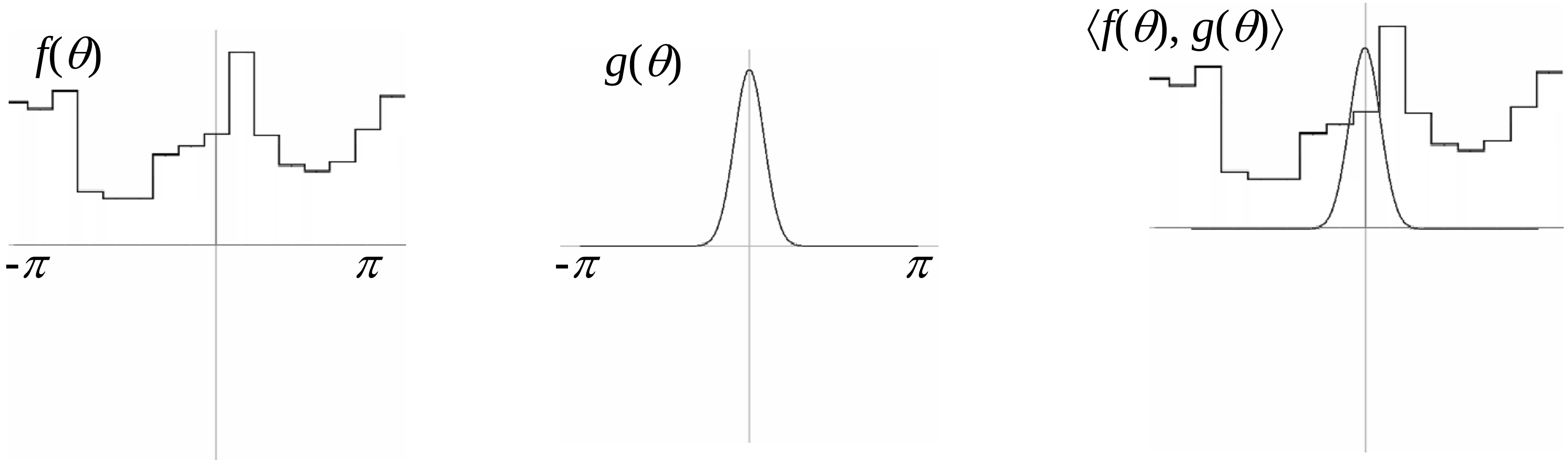
In signal/image/voxel processing, we are often interested in applying a filter to some initial data.

E.g. Smoothing:



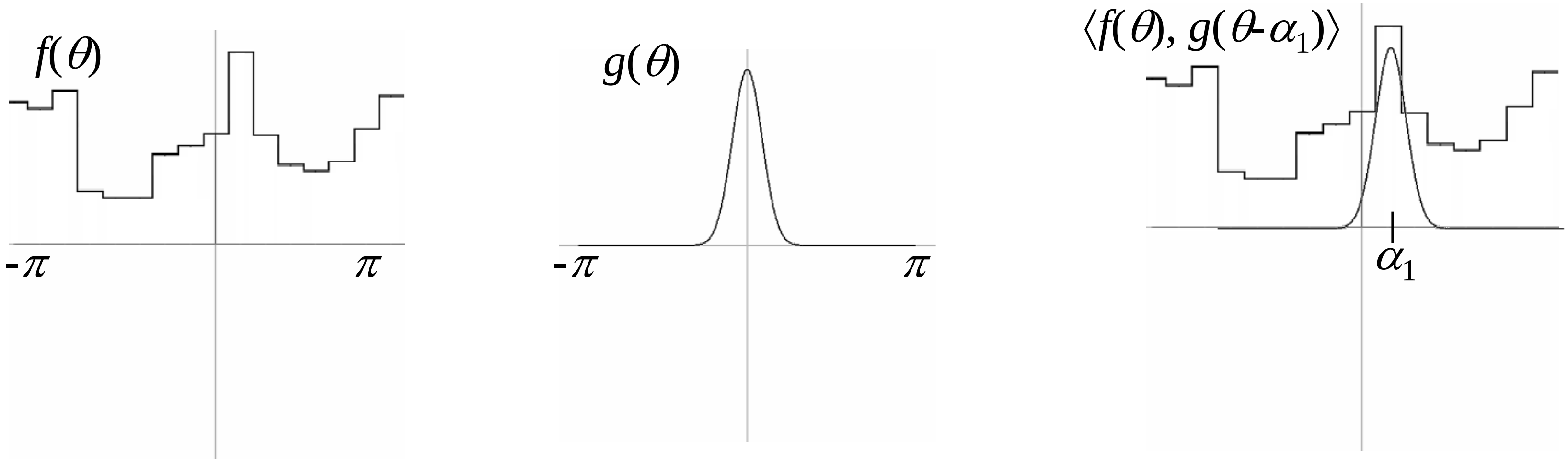
Smoothing

What we are really doing is computing a moving inner product:



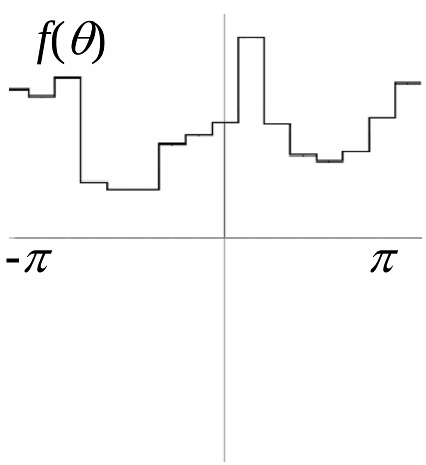
Smoothing

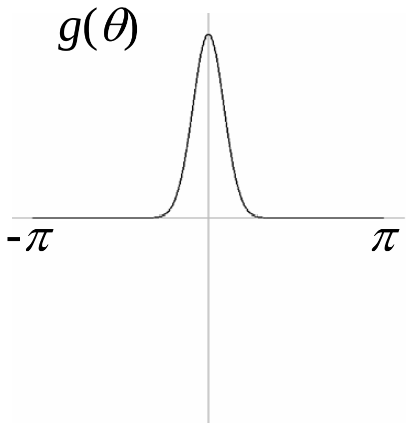
What we are really doing is computing a moving inner product:

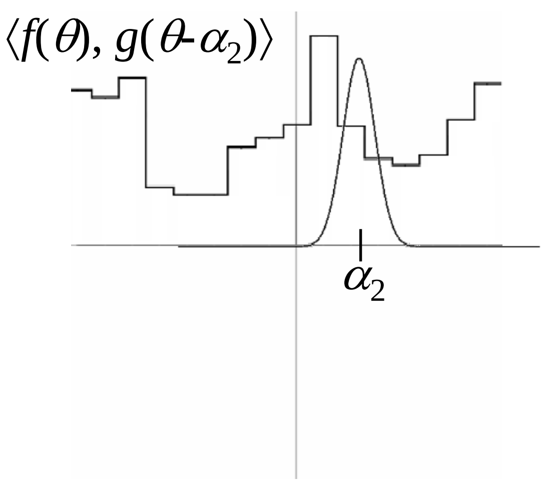


Smoothing

What we are really doing is computing a moving inner product:

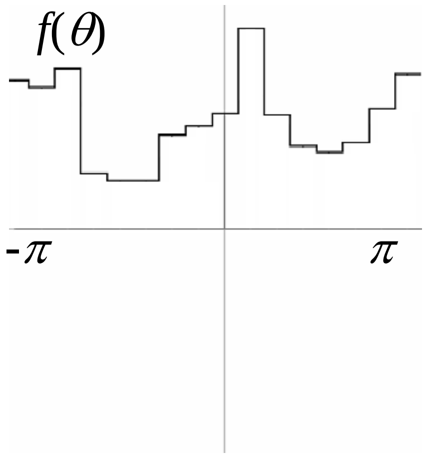


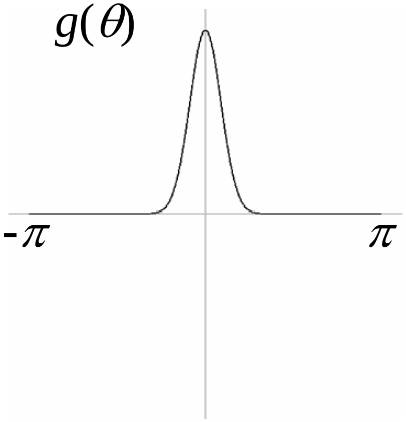


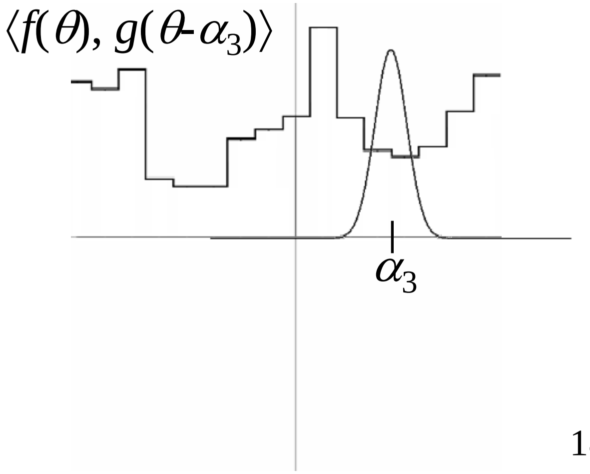


Smoothing

What we are really doing is computing a moving inner product:





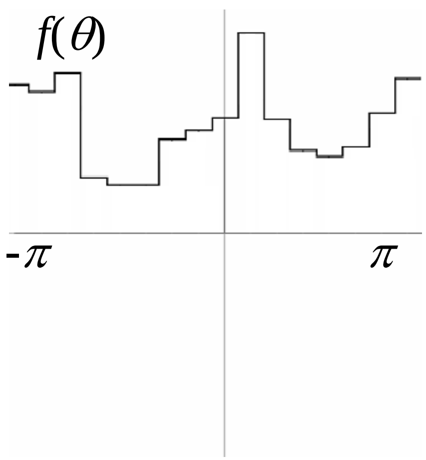


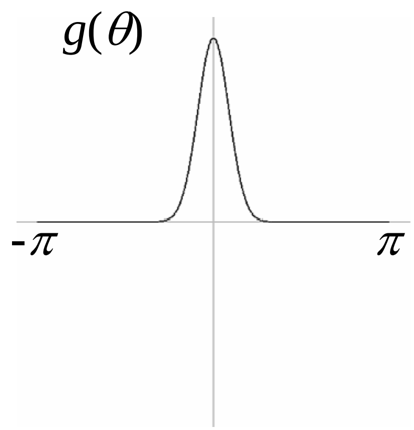
Smoothing

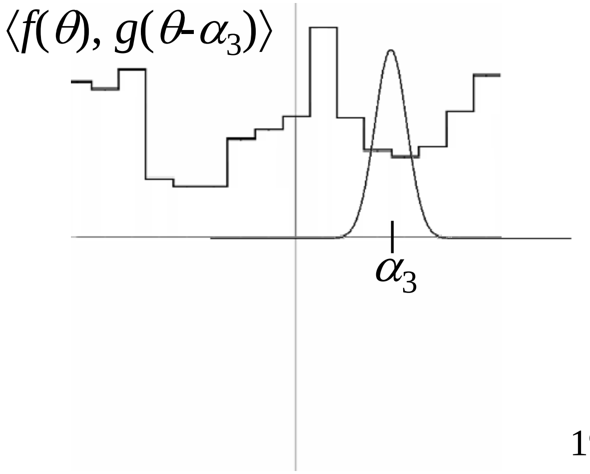
We can write out the operation of smoothing a signal *f* by a filter *g* as:

$$(f * g)(\alpha) = \langle f, \rho_\alpha(g) \rangle$$

where ρ_α is the linear transformation that translates a periodic function by α.







Moving Dot Products

We can think of this as a representation:

- V is the space of periodic functions on the line
- G is the group of real numbers in $[-\pi,\pi)$
- ρ_α is the representation translating a function by α.

Moving Dot Products

We can think of this as a representation:

- V is the space of periodic functions on the line
- G is the group of real numbers in $[-\pi,\pi)$
- ρ_α is the representation translating a function by α.

This is a representation of a commutative group…

Moving Dot Products

So there exist one-dimensional, complex, orthogonal, subspaces $V_1,\ldots,V_n \subset V$ that are the irreducible representations of V.

Moving Dot Products

So there exist one-dimensional, complex, orthogonal, subspaces $V_1,\ldots,V_n \subset V$ that are the irreducible representations of V.

In particular, we can set $\{f_i(\theta)\}$ to be an orthogonal basis for V by choosing each $f_i(\theta)$ to be some unit vector in V_i.

Moving Dot Products

So there exist one-dimensional, complex, orthogonal, subspaces $V_1,\ldots,V_n \subset V$ that are the irreducible representations of V.

In particular, we can set $\{f_i(\theta)\}$ to be an orthogonal basis for V by choosing each $f_i(\theta)$ to be some unit vector in V_i.

Since the V_i are sub-representations and they are one-dimensional, we know that:

$$\rho_\alpha\left(f_i(\theta)\right) = \lambda_i(\alpha) \cdot f_i(\theta)$$

Moving Dot Products

Since the $\{f_i(\theta)\}$ are a basis for V we can express the functions $f(\theta)$ and $g(\theta)$ in terms of this basis:

$$f(\theta) = a_1 f_1(\theta) + a_2 f_2(\theta) + \cdots + a_n f_n(\theta)$$
$$g(\theta) = b_1 f_1(\theta) + b_2 f_2(\theta) + \cdots + b_n f_n(\theta)$$

Moving Dot Products

Then the moving dot-product can be written as:

$$\left(f * g\right)(\alpha) = \left\langle f, \rho_\alpha(g) \right\rangle$$

Moving Dot Products

$$(f * g)(\alpha) = \langle f, \rho_\alpha(g) \rangle$$

Expanding *f* and *g* in terms of the basis $\{f_1,\ldots,f_n\}$:

$$(f * g)(\alpha) = \left\langle \sum_{i=1}^{n} a_i f_i, \rho_\alpha\left(\sum_{j=1}^{n} b_j f_j \right) \right\rangle$$

Moving Dot Products

$$(f * g)(\alpha) = \left\langle \sum_{i=1}^{n} a_i f_i, \rho_\alpha\left(\sum_{j=1}^{n} b_j f_j \right) \right\rangle$$

Using the fact that ρ_α is a linear transformation:

$$(f * g)(\alpha) = \left\langle \sum_{i=1}^{n} a_i f_i, \sum_{j=1}^{n} b_j \rho_\alpha(f_j) \right\rangle$$

Moving Dot Products

$$(f * g)(\alpha) = \left\langle \sum_{i=1}^{n} a_i f_i, \sum_{j=1}^{n} b_j \rho_\alpha(f_j) \right\rangle$$

Using the fact that the inner product is linear in the first term:

$$(f * g)(\alpha) = \sum_{i=1}^{n} a_i \left\langle f_i, \sum_{j=1}^{n} b_j \rho_\alpha(f_j) \right\rangle$$

Moving Dot Products

$$(f * g)(\alpha) = \sum_{i=1}^{n} a_i \left\langle f_i, \sum_{j=1}^{n} b_j \rho_\alpha(f_j) \right\rangle$$

Using the fact that the inner product is conjugate-linear in the second term:

$$(f * g)(\alpha) = \sum_{i,j=1}^{n} a_i \overline{b_j} \left\langle f_i, \rho_\alpha(f_j) \right\rangle$$

Moving Dot Products

$$(f * g)(\alpha) = \sum_{i,j=1}^{n} a_i \bar{b}_j \left\langle f_i, \rho_\alpha(f_j) \right\rangle$$

Using the fact that on V_i, the representation ρ_α is just scalar multiplication:

$$(f * g)(\alpha) = \sum_{i,j=1}^{n} a_i \bar{b}_j \left\langle f_i, \lambda_j(\alpha) f_j \right\rangle$$

Moving Dot Products

$$(f*g)(\alpha)=\sum_{i,j=1}^{n}a_i\overline{b_j}\left\langle f_i,\lambda_j(\alpha)f_j\right\rangle$$

Again, using the fact that the inner product is conjugate-linear in the second term:

$$(f*g)(\alpha)=\sum_{i,j=1}^{n}a_i\overline{b_j}\,\overline{\lambda_j}(\alpha)\left\langle f_i,f_j\right\rangle$$

Moving Dot Products

$$\left(f * g\right)(\alpha) = \sum_{i,j=1}^{n} a_i \overline{b_j}\,\overline{\lambda_j}(\alpha) \left\langle f_i, f_j \right\rangle$$

And finally, using the fact that the f_i are orthogonal unit-vectors:

$$\left(f * g\right)(\alpha) = \sum_{i=1}^{n} a_i \overline{b_i}\,\overline{\lambda_i}(\alpha)$$

Moving Dot Products

$$\left(f * g\right)(\alpha) = \sum_{i=1}^{n} a_i \overline{b_i} \overline{\lambda_i}(\alpha)$$

This implies that we can compute the moving dot-product by multiplying the coefficients of *f* and *g*.

Moving Dot Products

$$\left(f * g\right)(\alpha) = \sum_{i=1}^{n} a_i \overline{b_i} \overline{\lambda_i}(\alpha)$$

This implies that we can compute the moving dot-product by multiplying the coefficients of *f* and *g*.

Convolution in the spatial domain is multiplication in the frequency domain!

Outline

Review

Moving Dot Products:

- One-Dimensional (Continuous)
- One-Dimensional (Discrete)
- Higher-Dimensional
- Computational Complexity

Moving Dot Products (Periodic Functions)

Let’s consider the case of periodic functions in more detail:

Moving Dot Products (Periodic Functions)

Let's consider the case of periodic functions in more detail:

- *V* is the space of periodic functions on the line
- *G* is the group of real numbers in [0,2π)
- ρ_α is the representation translating a function by α:

$$\rho_\alpha\big(f(\theta)\big) = f(\theta - \alpha)$$

Moving Dot Products (Periodic Functions)

Let's consider the case of periodic functions in more detail:

- V is the space of periodic functions on the line
- G is the group of real numbers in $[0,2\pi)$
- ρ_α is the representation translating a function by α:

$$\rho_\alpha\big(f(\theta)\big) = f(\theta - \alpha)$$

What are the irreducible representations V_k?

What are the corresponding functions $\lambda_k(\alpha)$?

Moving Dot Products (Periodic Functions)

It turns out that the one-dimensional spaces V_k are the spans of the complex exponentials:

$$V_k = \text{Span}\left(e^{ik\theta}\right)$$

Moving Dot Products (Periodic Functions)

It turns out that the one-dimensional spaces V_k are the spans of the complex exponentials:

$$V_k = \mathrm{Span}\left(e^{ik\theta}\right)$$

Given any vector $v \in V_k$, applying ρ_α to v, we get:

$$\rho_\alpha(v) = \rho_\alpha\left(c \cdot e^{ik\theta}\right)$$

Moving Dot Products (Periodic Functions)

It turns out that the one-dimensional spaces V_k are the spans of the complex exponentials:

$$V_k = \mathrm{Span}\left(e^{ik\theta}\right)$$

Given any vector $v \in V_k$, applying ρ_α to v, we get:

$$\begin{aligned}\rho_\alpha(v) &= \rho_\alpha\left(c \cdot e^{ik\theta}\right)\\ &= c \cdot \rho_\alpha\left(e^{ik\theta}\right)\end{aligned}$$

Moving Dot Products (Periodic Functions)

It turns out that the one-dimensional spaces V_k are the spans of the complex exponentials:

$$V_k = \mathrm{Span}\left(e^{ik\theta}\right)$$

Given any vector $v \in V_k$, applying ρ_α to v, we get:

$$\begin{aligned}\rho_\alpha(v) &= \rho_\alpha\left(c \cdot e^{ik\theta}\right)\\ &= c \cdot \rho_\alpha\left(e^{ik\theta}\right)\\ &= c \cdot e^{ik(\theta-\alpha)}\end{aligned}$$

Moving Dot Products (Periodic Functions)

It turns out that the one-dimensional spaces V_k are the spans of the complex exponentials:

$$V_k = \mathrm{Span}\left(e^{ik\theta}\right)$$

Given any vector $v \in V_k$, applying ρ_α to v, we get:

$$\begin{aligned}\rho_\alpha(v) &= \rho_\alpha\left(c \cdot e^{ik\theta}\right)\\ &= c \cdot \rho_\alpha\left(e^{ik\theta}\right)\\ &= c \cdot e^{ik(\theta-\alpha)}\\ &= c \cdot e^{ik\theta} \cdot e^{-ik\alpha}\end{aligned}$$

Moving Dot Products (Periodic Functions)

It turns out that the one-dimensional spaces V_k are the spans of the complex exponentials:

$$V_k = \text{Span}\left(e^{ik\theta}\right)$$

Given any vector $v \in V_k$, applying ρ_α to v, we get:

$$\begin{aligned}\rho_\alpha(v) &= \rho_\alpha\left(c \cdot e^{ik\theta}\right) \\ &= c \cdot \rho_\alpha\left(e^{ik\theta}\right) \\ &= c \cdot e^{ik(\theta-\alpha)} \\ &= c \cdot e^{ik\theta} \cdot e^{-ik\alpha} \\ &= e^{-ik\alpha} \cdot v\end{aligned}$$

Moving Dot Products (Periodic Functions)

It turns out that the one-dimensional spaces V_k are the spans of the complex exponentials:

$$V_k = \mathrm{Span}\left(e^{ik\theta}\right)$$

Given any vector $v \in V_k$, applying ρ_α to v, we get:

$$\rho_\alpha(v) = \rho_\alpha\left(c \cdot e^{ik\theta}\right)$$

$$= c \cdot \rho_\alpha\left(e^{ik\theta}\right)$$

$$\overline{\lambda}_k(\alpha) = e^{ik\alpha}$$

$$= c \cdot e^{ik\theta} \cdot e^{-ik\alpha}$$

$$= e^{-ik\alpha} \cdot v$$

Moving Dot Products (Periodic Functions)

Note

The periodic functions:

$$f_k(\theta) = e^{ik\theta}$$

do not have unit norm!

Moving Dot Products (Periodic Functions)

Note

The periodic functions:

$$f_k(\theta) = e^{ik\theta}$$

do not have unit norm!

$$\| f_k(\theta) \|^2 = \langle f_k(\theta), f_k(\theta) \rangle$$

Moving Dot Products (Periodic Functions)

Note

The periodic functions:

$$f_k(\theta) = e^{ik\theta}$$

do not have unit norm!

$$\begin{aligned}\|f_k(\theta)\|^2 &= \langle f_k(\theta), f_k(\theta) \rangle \\ &= \int_0^{2\pi} f_k(\theta) \cdot \overline{f_k(\theta)}\, d\theta\end{aligned}$$

Moving Dot Products (Periodic Functions)

Note

The periodic functions:

$$f_k(\theta) = e^{ik\theta}$$

do not have unit norm!

$$\begin{aligned} \| f_k(\theta) \|^2 &= \langle f_k(\theta), f_k(\theta) \rangle \\ &= \int_0^{2\pi} f_k(\theta) \cdot \overline{f_k(\theta)} \, d\theta \\ &= \int_0^{2\pi} e^{ik\theta} \cdot e^{-ik\theta} d\theta \end{aligned}$$

Moving Dot Products (Periodic Functions)

Note

The periodic functions:

$$f_k(\theta) = e^{ik\theta}$$

do not have unit norm!

$$\begin{aligned}\|f_k(\theta)\|^2 &= \langle f_k(\theta), f_k(\theta) \rangle \\ &= \int_0^{2\pi} f_k(\theta) \cdot \overline{f_k(\theta)}\, d\theta \\ &= \int_0^{2\pi} e^{ik\theta} \cdot e^{-ik\theta} d\theta \\ &= \int_0^{2\pi} 1\, d\theta\end{aligned}$$

Moving Dot Products (Periodic Functions)

Note

The periodic functions:

$$f_k(\theta) = e^{ik\theta}$$

do not have unit norm!

$$\begin{aligned}
\left\| f_k(\theta) \right\|^2 &= \left\langle f_k(\theta), f_k(\theta) \right\rangle \\
&= \int_0^{2\pi} f_k(\theta) \cdot \overline{f_k(\theta)}\, d\theta \\
&= \int_0^{2\pi} e^{ik\theta} \cdot e^{-ik\theta} d\theta \\
&= \int_0^{2\pi} 1\, d\theta \\
&= 2\pi
\end{aligned}$$

Moving Dot Products (Periodic Functions)

Note

The periodic functions:

$$f_k(\theta) = e^{ik\theta}$$

do not have unit norm!

We need to normalize these functions to make them unit-norm:

$$f_k(\theta) = \sqrt{\frac{1}{2\pi}} e^{ik\theta}$$

Moving Dot Products (Periodic Functions)

Thus, given two periodic functions on the line, $f(\theta)$ and $g(\theta)$, we can write:

$$f(\theta)=\sum_{k=-\infty}^{\infty} a_k \sqrt{\frac{1}{2\pi}} e^{ik\theta} \quad \text{and} \quad g(\theta)=\sum_{k=-\infty}^{\infty} b_k \sqrt{\frac{1}{2\pi}} e^{ik\theta}$$

to get:

$$\left(f * g\right)(\alpha)=\sum_{k=-\infty}^{\infty} a_k \overline{b_k}\, \overline{\lambda_k}(\alpha)$$

Moving Dot Products (Periodic Functions)

Thus, given two periodic functions on the line, $f(\theta)$ and $g(\theta)$, we can write:

$$f(\theta) = \sum_{k=-\infty}^{\infty} a_k \sqrt{\frac{1}{2\pi}} e^{ik\theta} \quad \text{and} \quad g(\theta) = \sum_{k=-\infty}^{\infty} b_k \sqrt{\frac{1}{2\pi}} e^{ik\theta}$$

to get:

$$\begin{aligned}(f * g)(\alpha) &= \sum_{k=-\infty}^{\infty} a_k \overline{b_k} \overline{\lambda_k}(\alpha) \\ &= \sum_{k=-\infty}^{\infty} a_k \overline{b_k} e^{ik\alpha}\end{aligned}$$

Moving Dot Products (Periodic Functions)

Thus, given two periodic functions on the line, $f(\theta)$ and $g(\theta)$, we can write:

$$f(\theta)=\sum_{k=-\infty}^{\infty} a_k \sqrt{\frac{1}{2\pi}} e^{ik\theta} \quad \text{and} \quad g(\theta)=\sum_{k=-\infty}^{\infty} b_k \sqrt{\frac{1}{2\pi}} e^{ik\theta}$$

to get:

$$\begin{aligned}(f * g)(\alpha) &= \sum_{k=-\infty}^{\infty} a_k \overline{b}_k \overline{\lambda}_k(\alpha) \\ &= \sum_{k=-\infty}^{\infty} a_k \overline{b}_k e^{ik\alpha} \\ &= \sum_{k=-\infty}^{\infty} \sqrt{2\pi}\, a_k \overline{b}_k \sqrt{\frac{1}{2\pi}} e^{ik\alpha}\end{aligned}$$

Moving Dot Products (Periodic Functions)

What's really going on here?

Moving Dot Products (Periodic Functions)

What's really going on here?

What is a rotation by α degrees in the complex plane?

If we express a complex number in terms of radius and angle (r,θ), then rotation by α degrees corresponds to the map:

$$(r,\theta)\to(r,\theta+\alpha)$$

Moving Dot Products (Periodic Functions)

What's really going on here?

What is a rotation by α degrees in the complex plane?

If we express a complex number in terms of radius and angle (r,θ), then rotation by α degrees corresponds to the map:

$$(r,\theta) \rightarrow (r,\theta+\alpha)$$

$$\Updownarrow$$

$$re^{i\theta} \rightarrow re^{i(\theta+\alpha)} = e^{i\alpha}\left(re^{i\theta}\right)$$

Moving Dot Products (Periodic Functions)

What's really going on here?

What is a rotation by α degrees in the complex plane?

If we express a complex number in terms of radius and angle (r,θ), then rotation by α degrees corresponds to the map:

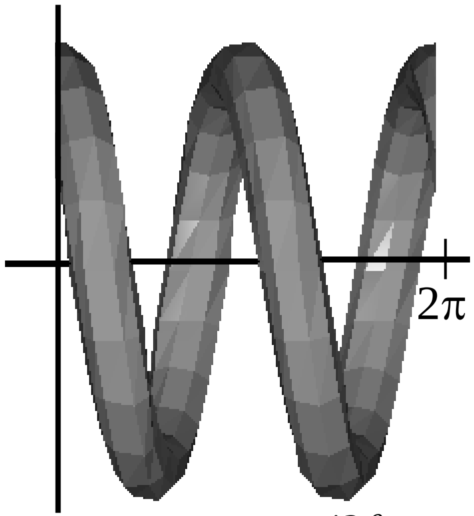
Rotating in the complex plane is the same thing as multiplying by a complex, unit-norm, number.

$$re^{i\theta} \rightarrow re^{i(\theta+\alpha)} = e^{i\alpha}\left(re^{i\theta}\right)$$

Moving Dot Products (Periodic Functions)

What's really going on here?

Let's consider the graph of a complex exponential. This is just a helix:



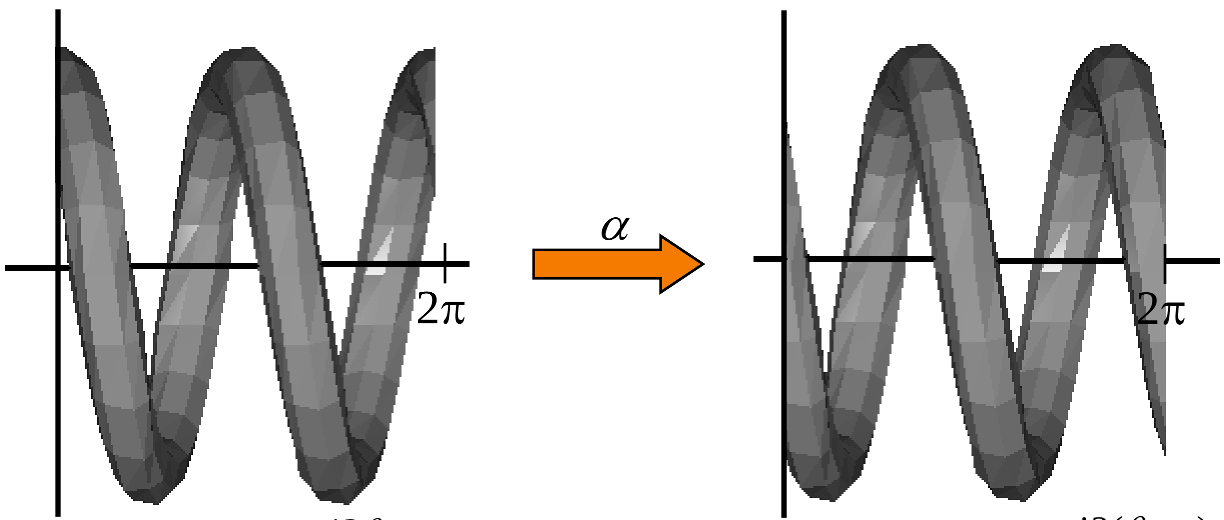
$$f(\theta) = e^{i2\theta}$$

Moving Dot Products (Periodic Functions)

What's really going on here?

Let's consider the graph of a complex exponential. This is just a helix.

If we translate the function by α, we get:



$f(\theta) = e^{i2\theta}$ $\qquad$ $f(\theta - \alpha) = e^{i2(\theta - \alpha)}$

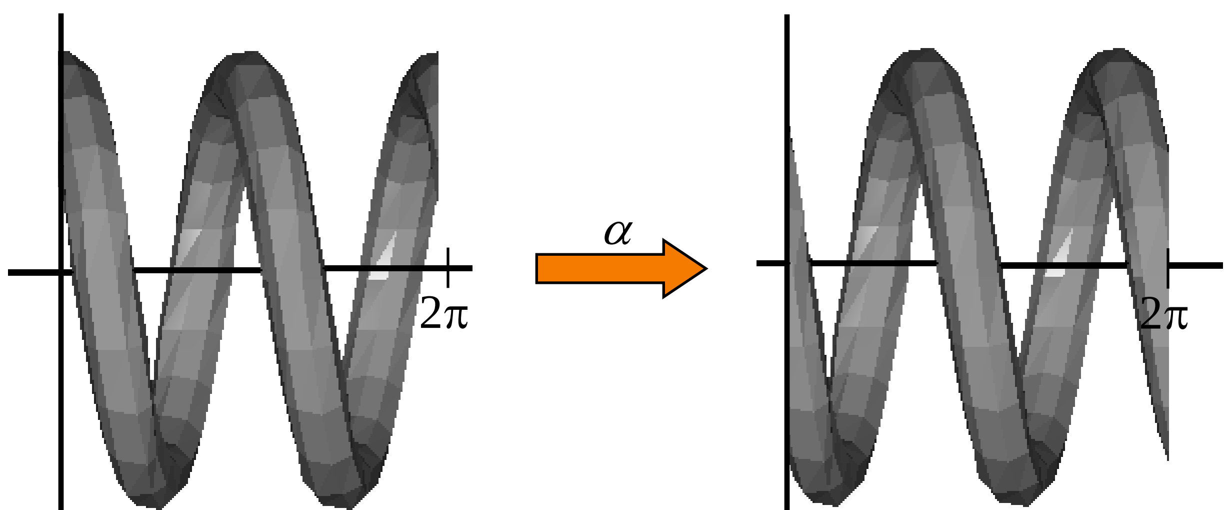
Moving Dot Products (Periodic Functions)

What's really going on here?

Let's consider the graph of a complex exponential. This is just a helix.

If we tr

Translating a periodic helix is the same thing as rotating it.



$f(\theta) = e^{i2\theta}$ $\qquad$ $f(\theta - \alpha) = e^{i2(\theta - \alpha)}$

Moving Dot Products (Periodic Functions)

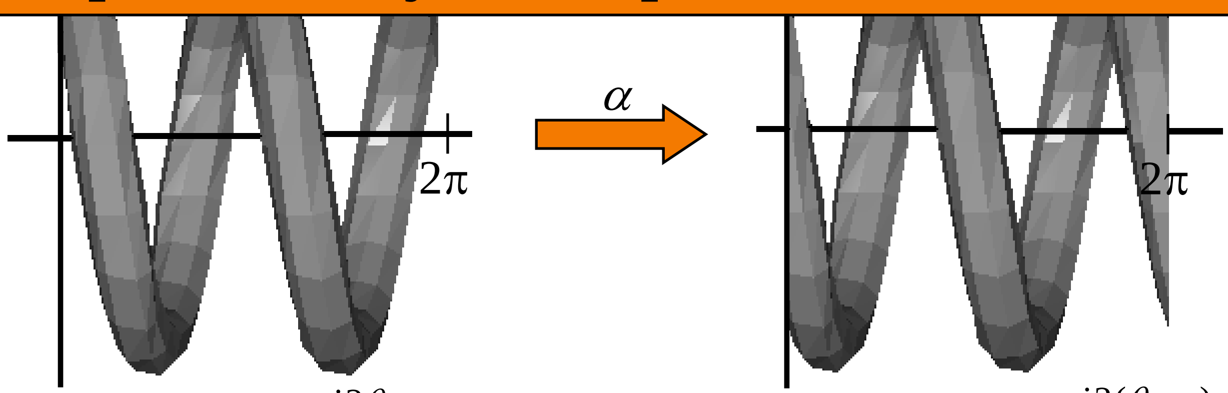
What’s really going on here?

Let’s consider the graph of a complex exponential. This is just a helix.

If we tr

Translating a periodic helix is the same thing as rotating it.

Rotation in the complex plane is the same thing as multiplication by a complex, unit-norm, number.



$f(\theta) = e^{i2\theta}$ $f(\theta-\alpha) = e^{i2(\theta-\alpha)}$

Outline

Review

Moving Dot Products:

- One-Dimensional (Continuous)
- One-Dimensional (Discrete)
- Higher-Dimensional
- Computational Complexity

Moving Dot Products (Periodic Arrays)

In practice, we don't have infinite precision, and so we discretize both the function space and the group:

- V is the space of periodic n-dimensional arrays
- G is the group of integers modulo n
- ρ_j is the representation shifting the entries in the array by j positions

Moving Dot Products (Periodic Arrays)

In practice, we don't have infinite precision, and so we discretize both the function space and the group:

- V is the space of periodic n-dimensional arrays
- G is the group of integers modulo n
- ρ_j is the representation shifting the entries in the array by j positions

What are the irreducible representations V_k?

What are the corresponding functions $\lambda_k(\alpha)$?

Moving Dot Products (Periodic Arrays)

We set V_k to be the one-dimensional spaces that are the discretized versions of the complex exponentials:

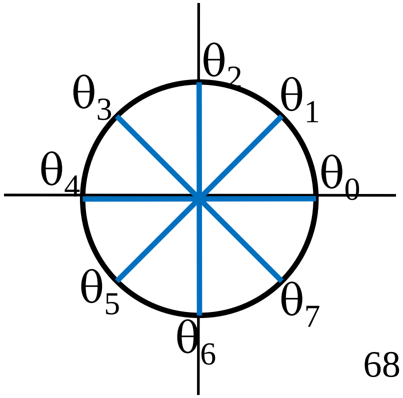
$$V_k = \mathrm{Span}(v_k)$$

where v_k is defined by sampling the *k*-th complex exponential:

$$v_k[\,] = \left(e^{ik\theta_0}, e^{ik\theta_1}, \ldots, e^{ik\theta_{n-2}}, e^{ik\theta_{n-1}}\right)$$

where:

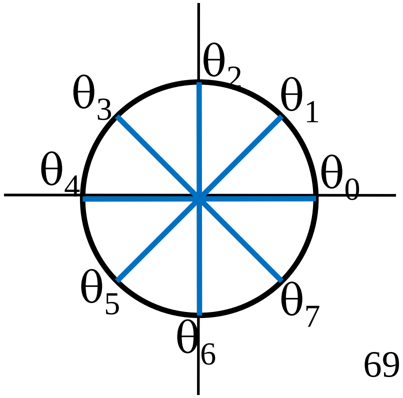
$$\theta_j = \frac{j2\pi}{n}$$



Moving Dot Products (Periodic Arrays)

Applying ρ_α to $v_k[\,]$, we get:

$$\rho_\alpha(v_k[\,]) = \left(e^{ik\theta_{0-\alpha}}, \ldots, e^{ik\theta_{n-1-\alpha}}\right)$$



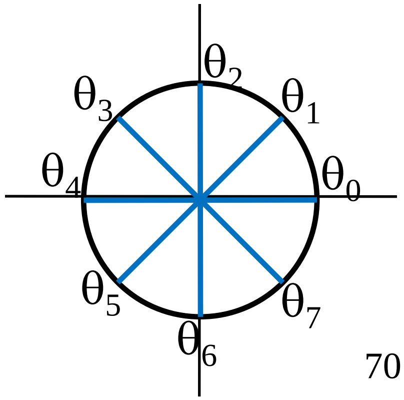
Moving Dot Products (Periodic Arrays)

Applying ρ_α to $v_k[\,]$, we get:

$$\rho_\alpha(v_k[\,]) = \left(e^{ik\theta_{0-\alpha}},\ldots,e^{ik\theta_{n-1-\alpha}}\right)$$

Now we can write out:

$$\theta_{j-\alpha} = \frac{(j-\alpha)2\pi}{n}$$



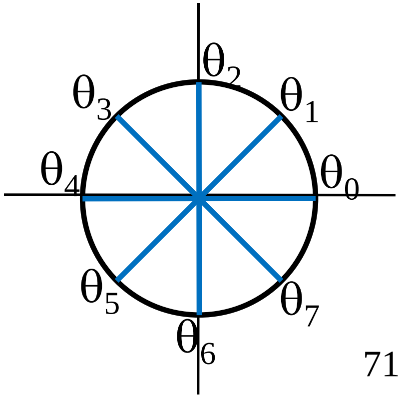
Moving Dot Products (Periodic Arrays)

Applying ρ_α to $v_k[\ ]$, we get:

$$\rho_\alpha(v_k[\ ]) = \left(e^{ik\theta_{0-\alpha}}, \ldots, e^{ik\theta_{n-1-\alpha}}\right)$$

Now we can write out:

$$\theta_{j-\alpha} = \frac{(j-\alpha)2\pi}{n}$$
$$= \frac{j2\pi}{n} + \frac{-\alpha 2\pi}{n}$$



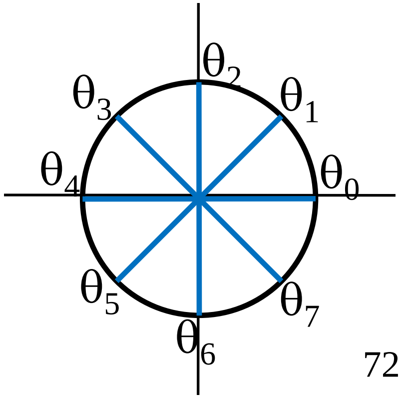
Moving Dot Products (Periodic Arrays)

Applying ρ_α to $v_k[\,]$, we get:

$$\rho_\alpha(v_k[\,]) = \left(e^{ik\theta_{0-\alpha}}, \ldots, e^{ik\theta_{n-1-\alpha}}\right)$$

Now we can write out:

$$\begin{aligned}\theta_{j-\alpha} &= \frac{(j-\alpha)2\pi}{n} \\ &= \frac{j2\pi}{n} + \frac{-\alpha 2\pi}{n} \\ &= \theta_j + \theta_{-\alpha}\end{aligned}$$



Moving Dot Products (Periodic Arrays)

Applying ρ_α to $v_k[\,]$, we get:

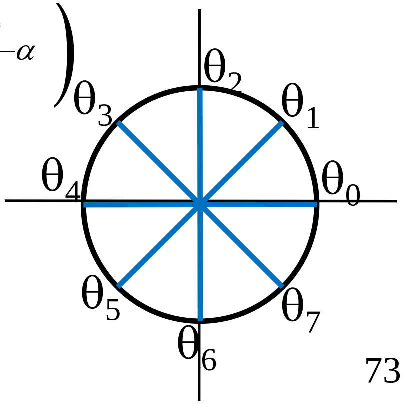
$$\rho_\alpha(v_k[\,]) = \left(e^{ik\theta_{0-\alpha}},\ldots,e^{ik\theta_{n-1-\alpha}}\right)$$

Now we can write out:

$$\theta_{j-\alpha} = \theta_j + \theta_{-\alpha}$$

So that:

$$\rho_\alpha(v_k[\,]) = \left(e^{ik\theta_0}\cdot e^{ik\theta_{-\alpha}},\ldots,e^{ik\theta_{n-1}}\cdot e^{ik\theta_{-\alpha}}\right)$$



Moving Dot Products (Periodic Arrays)

Applying ρ_α to $v_k[\,]$, we get:

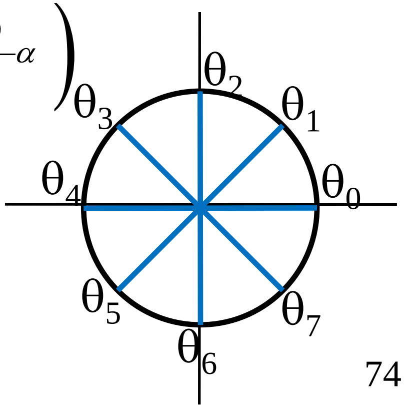
$$\rho_\alpha(v_k[\,]) = \left(e^{ik\theta_{0-\alpha}}, \ldots, e^{ik\theta_{n-1-\alpha}}\right)$$

Now we can write out:

$$\theta_{j-\alpha} = \theta_j + \theta_{-\alpha}$$

So that:

$$\begin{aligned}\rho_\alpha(v_k[\,]) &= \left(e^{ik\theta_0} \cdot e^{ik\theta_{-\alpha}}, \ldots, e^{ik\theta_{n-1}} \cdot e^{ik\theta_{-\alpha}}\right) \\ &= e^{ik\theta_{-\alpha}} \cdot v_k[\,]\end{aligned}$$



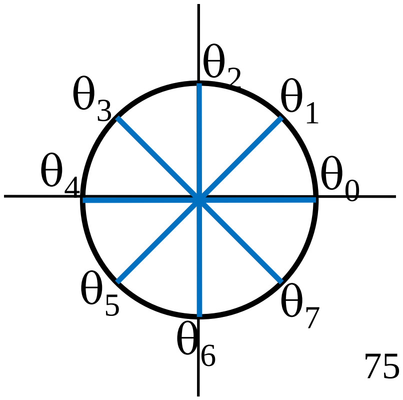
Moving Dot Products (Periodic Arrays)

Note 1

The periodic arrays:

$$v_k[\,] = \left(e^{ik\theta_0}, e^{ik\theta_1}, \ldots, e^{ik\theta_{n-2}}, e^{ik\theta_{n-1}}\right)$$

do not have unit norm!



Moving Dot Products (Periodic Arrays)

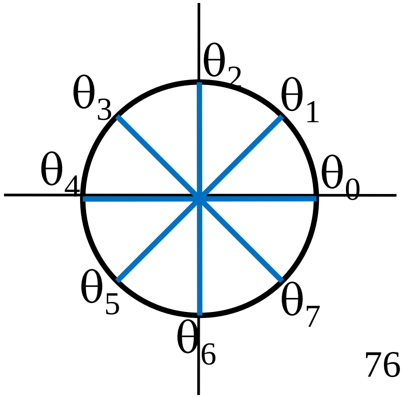
Note 1

The periodic arrays:

$$v_k[\,] = \left(e^{ik\theta_0}, e^{ik\theta_1}, \ldots, e^{ik\theta_{n-2}}, e^{ik\theta_{n-1}}\right)$$

do not have unit norm!

$$\|v_k[\,]\|^2 = \langle v_k[\,], v_k[\,] \rangle$$



Moving Dot Products (Periodic Arrays)

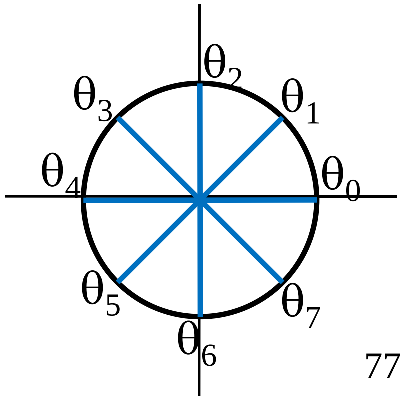
Note 1

The periodic arrays:

$$v_k[\,] = \left(e^{ik\theta_0}, e^{ik\theta_1}, \ldots, e^{ik\theta_{n-2}}, e^{ik\theta_{n-1}}\right)$$

do not have unit norm!

$$\begin{aligned}\|v_k[\,]\|^2 &= \langle v_k[\,], v_k[\,]\rangle \\ &= \sum_{j=0}^{n-1} v_k[j]\cdot\overline{v_k[j]}\end{aligned}$$



Moving Dot Products (Periodic Arrays)

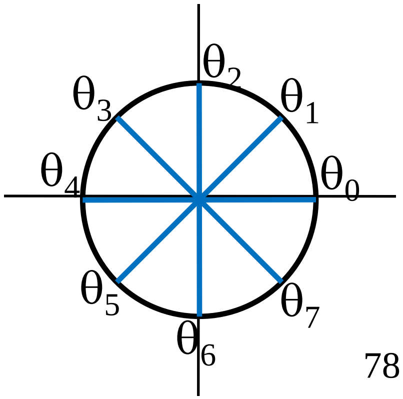
Note 1

The periodic arrays:

$$v_k[\,] = \left(e^{ik\theta_0}, e^{ik\theta_1}, \ldots, e^{ik\theta_{n-2}}, e^{ik\theta_{n-1}}\right)$$

do not have unit norm!

$$\begin{aligned} \|v_k[\,]\|^2 &= \langle v_k[\,], v_k[\,] \rangle \\ &= \sum_{j=0}^{n-1} v_k[j] \cdot \overline{v_k[j]} \\ &= \sum_{j=0}^{n-1} e^{ik\theta_j} \cdot e^{-ik\theta_j} \end{aligned}$$



Moving Dot Products (Periodic Arrays)

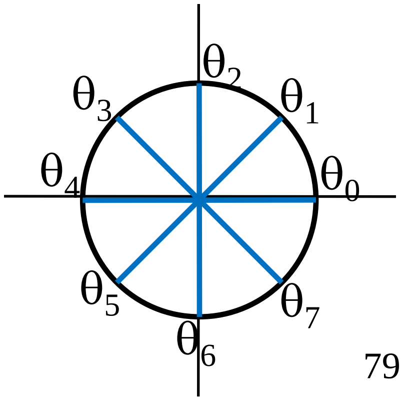
Note 1

The periodic arrays:

$$v_k[\ ] = \left(e^{ik\theta_0}, e^{ik\theta_1}, \ldots, e^{ik\theta_{n-2}}, e^{ik\theta_{n-1}}\right)$$

do not have unit norm!

$$\begin{aligned}\left\|v_k[\ ]\right\|^2 &= \left\langle v_k[\ ], v_k[\ ]\right\rangle \\ &= \sum_{j=0}^{n-1} v_k[j]\cdot\overline{v_k[j]} \\ &= \sum_{j=0}^{n-1} e^{ik\theta_j}\cdot e^{-ik\theta_j} \\ &= \sum_{j=0}^{n-1} 1\end{aligned}$$



Moving Dot Products (Periodic Arrays)

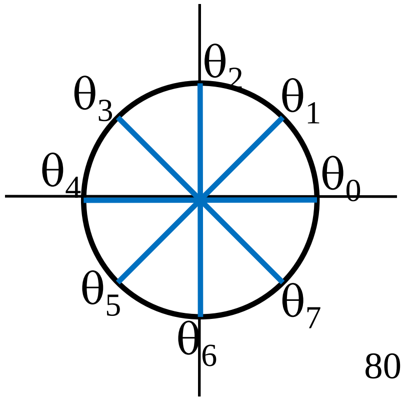
Note 1

The periodic arrays:

$$v_k[\,] = \left(e^{ik\theta_0}, e^{ik\theta_1}, \ldots, e^{ik\theta_{n-2}}, e^{ik\theta_{n-1}}\right)$$

do not have unit norm!

$$\begin{aligned}\left\|v_k[\,]\right\|^2 &= \left\langle v_k[\,], v_k[\,]\right\rangle \\ &= \sum_{j=0}^{n-1} v_k[j]\cdot\overline{v_k[j]} \\ &= \sum_{j=0}^{n-1} e^{ik\theta_j}\cdot e^{-ik\theta_j} \\ &= \sum_{j=0}^{n-1} 1 \\ &= n\end{aligned}$$



Moving Dot Products (Periodic Arrays)

Note 1

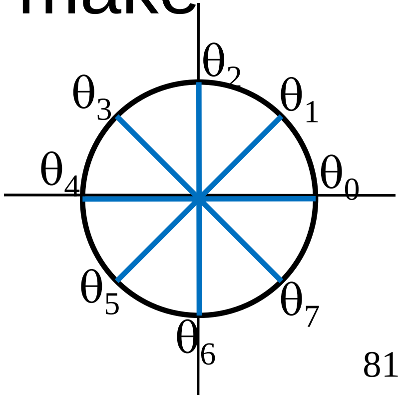
The periodic arrays:

$$v_k[\,]=\left(e^{ik\theta_0},e^{ik\theta_1},\ldots,e^{ik\theta_{n-2}},e^{ik\theta_{n-1}}\right)$$

do not have unit norm!

We need to normalize these functions to make them unit-norm:

$$v_k=\sqrt{\frac{1}{n}}\left(e^{ik\theta_0},e^{ik\theta_1},\ldots,e^{ik\theta_{n-2}},e^{ik\theta_{n-1}}\right)$$

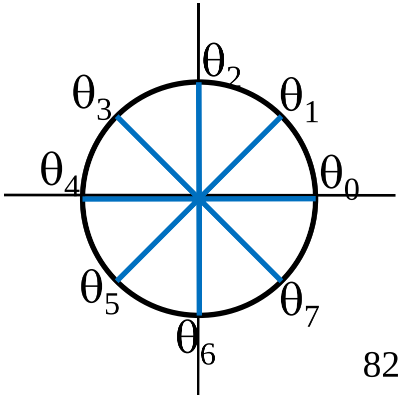


Moving Dot Products (Periodic Arrays)

Note 2

The arrays $v_k[\,]$ and $v_{k+n}[\,]$ are the same array:

$$\sqrt{n}\cdot v_{k+n}[\,]=\left(e^{i(k+n)\theta_0},\ldots,e^{i(k+n)\theta_{n-1}}\right)$$

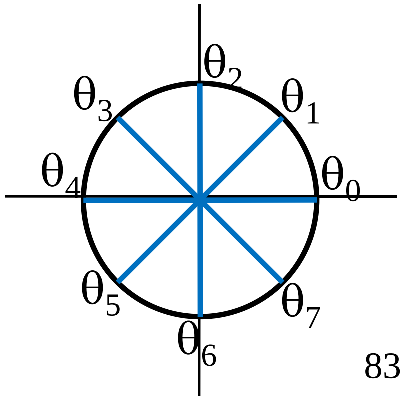


Moving Dot Products (Periodic Arrays)

Note 2

The arrays $v_k[\,]$ and $v_{k+n}[\,]$ are the same array:

$$\begin{aligned}\sqrt{n}\cdot v_{k+n}[\,] &= \left(e^{i(k+n)\theta_0},\ldots,e^{i(k+n)\theta_{n-1}}\right)\\ &= \left(e^{ik\theta_0}\cdot e^{in\theta_0},\ldots,e^{ik\theta_{n-1}}\cdot e^{in\theta_{n-1}}\right)\end{aligned}$$



Moving Dot Products (Periodic Arrays)

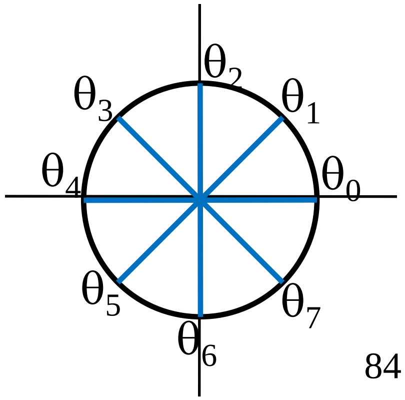
Note 2

The arrays $v_k[\,]$ and $v_{k+n}[\,]$ are the same array:

$$\sqrt{n}\cdot v_{k+n}[\,] = \left(e^{i(k+n)\theta_0},\ldots,e^{i(k+n)\theta_{n-1}}\right)$$
$$= \left(e^{ik\theta_0}\cdot e^{in\theta_0},\ldots,e^{ik\theta_{n-1}}\cdot e^{in\theta_{n-1}}\right)$$

But $n\theta_j$ is just a multiple of 2π:

$$n\theta_j = \frac{nj2\pi}{n} = j2\pi$$



Moving Dot Products (Periodic Arrays)

Note 2

The arrays $v_k[\,]$ and $v_{k+n}[\,]$ are the same array:

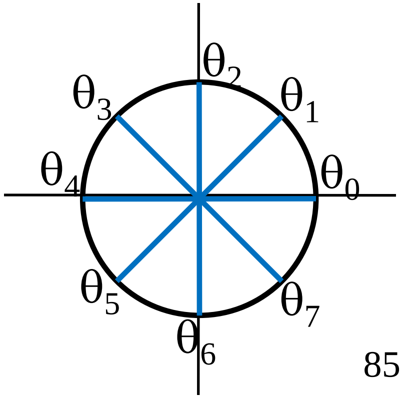
$$\sqrt{n}\cdot v_{k+n}[\,] = \left(e^{i(k+n)\theta_0},\ldots,e^{i(k+n)\theta_{n-1}}\right)$$
$$= \left(e^{ik\theta_0}\cdot e^{in\theta_0},\ldots,e^{ik\theta_{n-1}}\cdot e^{in\theta_{n-1}}\right)$$

But $n\theta_j$ is just a multiple of 2π:

$$n\theta_j = \frac{nj2\pi}{n} = j2\pi$$

$$\Updownarrow$$

$$e^{in\theta_j} = 1$$



Moving Dot Products (Periodic Arrays)

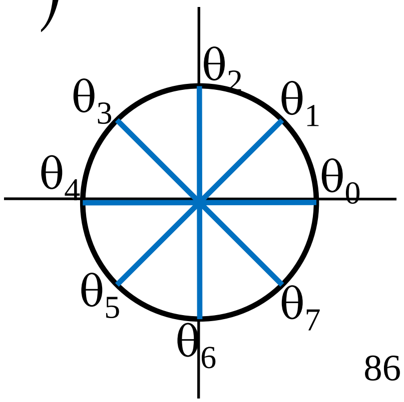
Note 2

The arrays $v_k[\,]$ and $v_{k+n}[\,]$ are the same array:

$$\sqrt{n}\cdot v_{k+n}[\,] = \left(e^{i(k+n)\theta_0},\ldots,e^{i(k+n)\theta_{n-1}}\right)$$
$$= \left(e^{ik\theta_0}\cdot e^{in\theta_0},\ldots,e^{ik\theta_{n-1}}\cdot e^{in\theta_{n-1}}\right)$$

But $n\theta_j$ is just a multiple of 2π, so

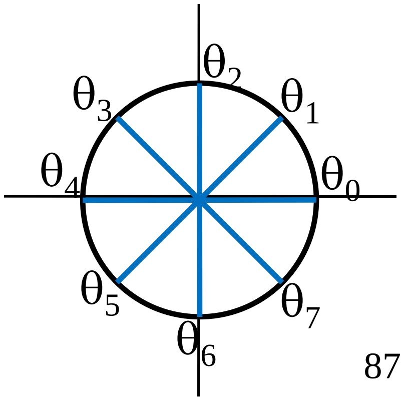
$$\sqrt{n}\cdot v_{k+n}[\,] = \left(e^{ik\theta_0}\cdot e^{in\theta_0},\ldots,e^{ik\theta_{n-1}}\cdot e^{in\theta_{n-1}}\right)$$
$$= \left(e^{ik\theta_0},\ldots,e^{ik\theta_{n-1}}\right)$$
$$= \sqrt{n}\cdot v_k[\,]$$



Moving Dot Products (Periodic Arrays)

Note 3

The arrays $\{v_0[\ ],\ldots, v_{n-1}[\ ]\}$ are linearly independent.



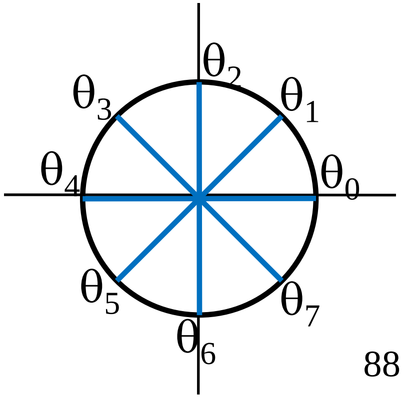
Moving Dot Products (Periodic Arrays)

Thus, given two *n*-dimensional arrays, *f*[ ] and *g*[ ], we can write:

$$f[\,] = \sum_{k=0}^{n-1} a_k v_k[\,] \quad \text{and} \quad g[\,] = \sum_{k=0}^{n-1} b_k v_k[\,]$$

to get:

$$\big(f[\,] * g[\,]\big)[\alpha] = \sum_{k=0}^{n-1} a_k \overline{b_k}\, \overline{\lambda_k}[\alpha]$$



Moving Dot Products (Periodic Arrays)

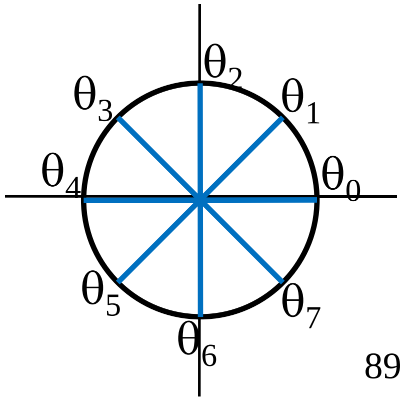
Thus, given two *n*-dimensional arrays, *f*[ ] and *g*[ ], we can write:

$$f[\,] = \sum_{k=0}^{n-1} a_k v_k[\,] \quad \text{and} \quad g[\,] = \sum_{k=0}^{n-1} b_k v_k[\,]$$

to get:

$$\left(f[\,] * g[\,]\right)[\alpha] = \sum_{k=0}^{n-1} a_k \overline{b_k}\, \overline{\lambda_k}[\alpha]$$

$$= \sqrt{n} \sum_{k=0}^{n-1} a_k \overline{b_k} v_k[\alpha]$$



Outline

Review

Moving Dot Products:

- One-Dimensional (Continuous)
- One-Dimensional (Discrete)
- Higher-Dimensional
- Computational Complexity

Moving Dot Products (Higher Dimensions)

The same kind of method can be used for higher dimensions:

- Periodic functions in 2D

$$f_{lm}(\theta,\phi)=\sqrt{\frac{1}{(2\pi)^2}}e^{il\theta}\cdot e^{im\phi}$$

$$\overline{\lambda}_{lm}(\alpha,\beta)=\sqrt{(2\pi)^2}\,f_{lm}(\alpha,\beta)$$

Moving Dot Products (Higher Dimensions)

The same kind of method can be used for higher dimensions:

- Periodic functions in 2D

$$f_{lm}(\theta,\phi)=\sqrt{\frac{1}{(2\pi)^2}}e^{il\theta}\cdot e^{im\phi}$$

$$\bar{\lambda}_{lm}(\alpha,\beta)=\sqrt{(2\pi)^2}\,f_{lm}(\alpha,\beta)$$

- Periodic functions in 3D

$$f_{lmn}(\theta,\phi,\varphi)=\sqrt{\frac{1}{(2\pi)^3}}e^{il\theta}\cdot e^{im\phi}\cdot e^{in\varphi}$$

$$\bar{\lambda}_{lmn}(\alpha,\beta,\gamma)=\sqrt{(2\pi)^3}\,f_{lmn}(\alpha,\beta,\gamma)$$

Outline

Review

Moving Dot Products:

- One-Dimensional (Continuous)
- One-Dimensional (Discrete)
- Higher-Dimensional
- Computational Complexity

Computational Complexity

What do we need to do in order to compute the moving dot-product of two periodic, n-dimensional arrays $f[\ ]$ and $g[\ ]$?

Computational Complexity

What do we need to do in order to compute the moving dot-product of two periodic, n-dimensional arrays $f[\,]$ and $g[\,]$?

1. We need to express $f[\,]$ and $g[\,]$ in terms of the basis $v_k[\,]$:

$$f[\,]=\sum_{k=0}^{n-1}a_k v_k[\,] \quad \text{and} \quad g[\,]=\sum_{k=0}^{n-1}b_k v_k[\,]$$

Computational Complexity

What do we need to do in order to compute the moving dot-product of two periodic, *n*-dimensional arrays *f*[ ] and *g*[ ]?

1. We need to express *f*[ ] and *g*[ ] in terms of the basis v_k[ ]:

$$f[\,] = \sum_{k=0}^{n-1} a_k v_k[\,] \quad \text{and} \quad g[\,] = \sum_{k=0}^{n-1} b_k v_k[\,]$$

2. We need to multiply the coefficients:

$$\left(f[\,] * g[\,]\right)[\,] = \sqrt{n} \sum_{k=0}^{n-1} a_k \overline{b_k} v_k[\,]$$

Computational Complexity

What do we need to do in order to compute the moving dot-product of two periodic, n-dimensional arrays $f[\,]$ and $g[\,]$?

1. We need to express $f[\,]$ and $g[\,]$ in terms of the basis $v_k[\,]$:

$$f[\,]=\sum_{k=0}^{n-1}a_k v_k[\,] \quad \text{and} \quad g[\,]=\sum_{k=0}^{n-1}b_k v_k[\,]$$

2. We need to multiply the coefficients:

$$\left(f[\,]*g[\,]\right)[\,]=\sqrt{n}\sum_{k=0}^{n-1}a_k\overline{b_k}v_k[\,]$$

3. We need to evaluate the moving dot-product at every index α:

$$\left(f[\,]*g[\,]\right)[\alpha]=\sqrt{n}\sum_{k=0}^{n-1}a_k\overline{b_k}v_k[\alpha]$$

Computational Complexity

What do we need to do in order to compute the moving dot-product of two periodic, *n*-dimensional arrays *f*[ ] and *g*[ ]?

The first and third steps are a change of bases.

This amounts to a matrix multiplication and can be as bad as quadratic in the dimension of the array.

Computational Complexity

What do we need to do in order to compute the moving dot-product of two periodic, *n*-dimensional arrays *f*[ ] and *g*[ ]?

1. We need to express *f*[ ] and *g*[ ] in terms of the basis v_k[ ]:

$$f[\,]=\sum_{k=0}^{n-1}a_k v_k[\,] \quad \text{and} \quad g[\,]=\sum_{k=0}^{n-1}b_k v_k[\,] \qquad \boxed{O(n^2)}$$

2. We need to multiply the coefficients:

$$\left(f[\,]*g[\,]\right)[\,]=\sqrt{n}\sum_{k=0}^{n-1}a_k\overline{b_k}v_k[\,] \qquad \boxed{O(n)}$$

3. We need to evaluate the moving dot-product at every index α:

$$\left(f[\,]*g[\,]\right)[\alpha]=\sqrt{n}\sum_{k=0}^{n-1}a_k\overline{b_k}v_k[\alpha] \qquad \boxed{O(n^2)}$$

Computational Complexity

What do we need to do in order to compute the moving dot-product of two periodic, n-dimensional arrays $f[\,]$ and $g[\,]$?

The Fast Fourier Transform (FFT) is an algorithm for expressing an array represented by samples at $\{\theta_0,\ldots,\theta_{n-1}\}$ as a linear sum of the v_k.

The Fast Inverse Fourier Transform (IFFT) is an algorithm for expressing an array represented as a linear sum of the v_k by samples at $\{\theta_0,\ldots,\theta_{n-1}\}$.

Both take $O(n \log n)$ time.

Computational Complexity

What do we need to do in order to compute the moving dot-product of two periodic, *n*-dimensional arrays *f*[ ] and *g*[ ]?

1. We need to express *f*[ ] and *g*[ ] in terms of the basis v_k[ ]:

$$f[\,] = \sum_{k=0}^{n-1} a_k v_k[\,] \quad \text{and} \quad g[\,] = \sum_{k=0}^{n-1} b_k v_k[\,] \qquad \boxed{O(n \log n)}$$

2. We need to multiply the coefficients:

$$\left(f[\,] * g[\,]\right)[\,] = \sqrt{n} \sum_{k=0}^{n-1} a_k \bar{b}_k v_k[\,] \qquad \boxed{O(n)}$$

3. We need to evaluate the moving dot-product at every index α:

$$\left(f[\,] * g[\,]\right)[\alpha] = \sqrt{n} \sum_{k=0}^{n-1} a_k \bar{b}_k v_k[\alpha] \qquad \boxed{O(n \log n)}$$

The Inverse Fourier Transform

The Fourier Transform is a change of basis transformation:

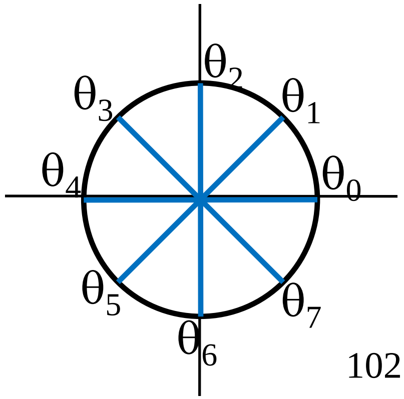
Evaluation Basis

$$(1,0,\ldots,0,0)$$
$$(0,1,\ldots,0,0)$$
$$\vdots$$
$$(0,0,\ldots,1,0)$$
$$(0,0,\ldots,0,1)$$

Fourier Transform $\longrightarrow$

Complex Exponential Basis

$$(e^{i0\theta_0},\quad e^{i0\theta_1},\quad \ldots,\quad e^{i0\theta_{n-2}},\quad e^{i0\theta_{n-1}})/\sqrt{n}$$
$$(e^{i1\theta_0},\quad e^{i1\theta_1},\quad \ldots,\quad e^{i1\theta_{n-2}},\quad e^{i1\theta_{n-1}})/\sqrt{n}$$
$$\vdots$$
$$(e^{i(n-2)\theta_0},e^{i(n-2)\theta_1},\ldots,e^{i(n-2)\theta_{n-2}},e^{i(n-2)\theta_{n-1}})/\sqrt{n}$$
$$(e^{i(n-1)\theta_0},e^{i(n-1)\theta_1},\ldots,e^{i(n-1)\theta_{n-2}},e^{i(n-1)\theta_{n-1}})/\sqrt{n}$$



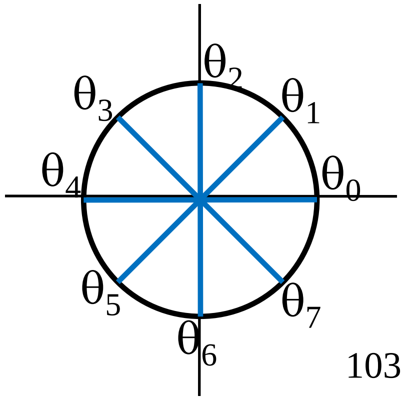
The Inverse Fourier Transform

This can be represented by the matrix:

$$F = \sqrt{\frac{1}{n}} \begin{pmatrix} 1 & 1 & \cdots & 1 & 1 \\ 1 & e^{i\theta} & \cdots & e^{i(n-2)\theta} & e^{i(n-1)\theta} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & e^{i(n-2)\theta} & \cdots & e^{i(n-2)(n-2)\theta} & e^{i(n-2)(n-1)\theta} \\ 1 & e^{i(n-1)\theta} & \cdots & e^{i(n-2)(n-1)\theta} & e^{i(n-1)(n-1)\theta} \end{pmatrix}$$

where θ is the angle:

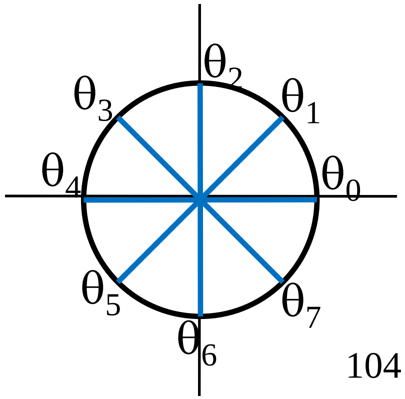
$$\theta = \frac{2\pi}{n}$$



The Inverse Fourier Transform

$$F = \sqrt{\frac{1}{n}} \begin{pmatrix} 1 & 1 & \cdots & 1 & 1 \\ 1 & e^{i\theta} & \cdots & e^{i(n-2)\theta} & e^{i(n-1)\theta} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & e^{i(n-2)\theta} & \cdots & e^{i(n-2)(n-2)\theta} & e^{i(n-2)(n-1)\theta} \\ 1 & e^{i(n-1)\theta} & \cdots & e^{i(n-2)(n-1)\theta} & e^{i(n-1)(n-1)\theta} \end{pmatrix}$$

Since both bases are orthogonal, the matrix is unitary, and the inverse Fourier transform is just the transpose conjugate of the forward Fourier transform.



The Inverse Fourier Transform

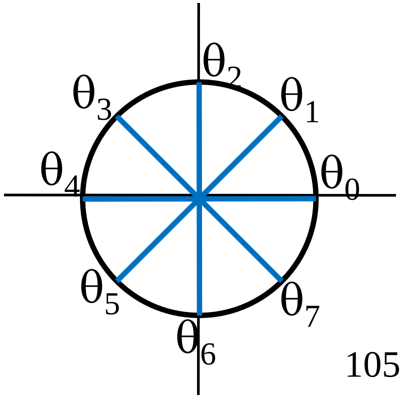
$$F = \sqrt{\frac{1}{n}} \begin{pmatrix} 1 & 1 & \cdots & 1 & 1 \\ 1 & e^{i\theta} & \cdots & e^{i(n-2)\theta} & e^{i(n-1)\theta} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & e^{i(n-2)\theta} & \cdots & e^{i(n-2)(n-2)\theta} & e^{i(n-2)(n-1)\theta} \\ 1 & e^{i(n-1)\theta} & \cdots & e^{i(n-2)(n-1)\theta} & e^{i(n-1)(n-1)\theta} \end{pmatrix}$$

In particular, given the Fourier coefficients:

$$(a_0, \ldots, a_{n-1})$$

the inverse Fourier transform gives:

$$\overline{F}^t \begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix}$$

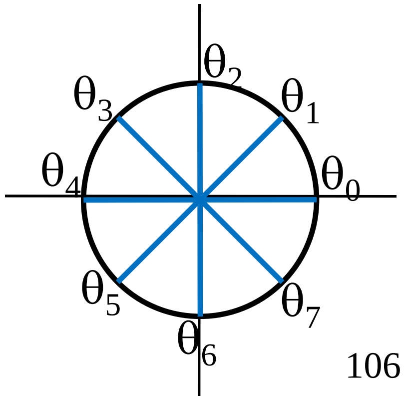


The Inverse Fourier Transform

$$F = \sqrt{\frac{1}{n}} \begin{pmatrix} 1 & 1 & \cdots & 1 & 1 \\ 1 & e^{i\theta} & \cdots & e^{i(n-2)\theta} & e^{i(n-1)\theta} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & e^{i(n-2)\theta} & \cdots & e^{i(n-2)(n-2)\theta} & e^{i(n-2)(n-1)\theta} \\ 1 & e^{i(n-1)\theta} & \cdots & e^{i(n-2)(n-1)\theta} & e^{i(n-1)(n-1)\theta} \end{pmatrix}$$

Taking the double conjugate, we get:

$$\overline{F}^t \begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix} = \overline{\overline{\overline{F}^t \begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix}}}$$

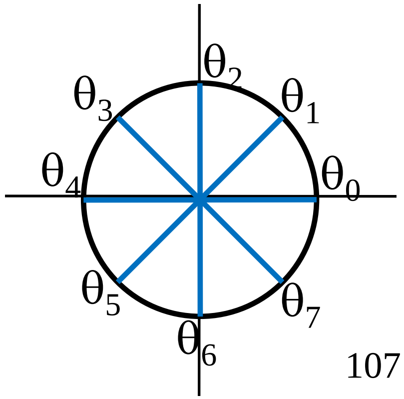


The Inverse Fourier Transform

$$F = \sqrt{\frac{1}{n}} \begin{pmatrix} 1 & 1 & \cdots & 1 & 1 \\ 1 & e^{i\theta} & \cdots & e^{i(n-2)\theta} & e^{i(n-1)\theta} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & e^{i(n-2)\theta} & \cdots & e^{i(n-2)(n-2)\theta} & e^{i(n-2)(n-1)\theta} \\ 1 & e^{i(n-1)\theta} & \cdots & e^{i(n-2)(n-1)\theta} & e^{i(n-1)(n-1)\theta} \end{pmatrix}$$

Taking the double conjugate, we get:

$$\overline{F}^{t} \begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix} = \overline{\overline{\overline{F}^{t} \begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix}}} = F^{t} \begin{pmatrix} \overline{a}_0 \\ \vdots \\ \overline{a}_{n-1} \end{pmatrix}$$

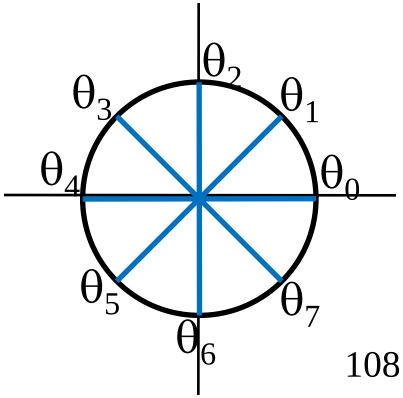


The Inverse Fourier Transform

$$F = \sqrt{\frac{1}{n}} \begin{pmatrix} 1 & 1 & \cdots & 1 & 1 \\ 1 & e^{i\theta} & \cdots & e^{i(n-2)\theta} & e^{i(n-1)\theta} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & e^{i(n-2)\theta} & \cdots & e^{i(n-2)(n-2)\theta} & e^{i(n-2)(n-1)\theta} \\ 1 & e^{i(n-1)\theta} & \cdots & e^{i(n-2)(n-1)\theta} & e^{i(n-1)(n-1)\theta} \end{pmatrix}$$

Since $F = F^t$, this implies that:

$$\overline{F}^{\,t} \begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix} = \overline{F \begin{pmatrix} \overline{a}_0 \\ \vdots \\ \overline{a}_{n-1} \end{pmatrix}}$$



The Inverse Fourier Transform

$$\overline{F}^{\,t}\begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix} = \overline{F\begin{pmatrix} \overline{a}_0 \\ \vdots \\ \overline{a}_{n-1} \end{pmatrix}}$$

Thus, we can compute the inverse Fourier transform by:

1. Taking the conjugate of the Fourier coefficients
2. Computing the forward Fourier transform
3. Taking the conjugate of the resultant coefficients.