

# Towards Automatic Music Transcription using Nonnegative Independent Component Analysis

An audio signal processing example  
of optimization on manifolds

Mark Plumbley  
Queen Mary, University of London



# Outline

- Example task:  
Automatic Polyphonic Music Transcription
- Independent Component Analysis (ICA)
- Nonnegative ICA
- Optimization with orthogonality constraint
- Fourier expansion update
- Future possibilities

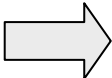
# Part 1: Problem Introduction

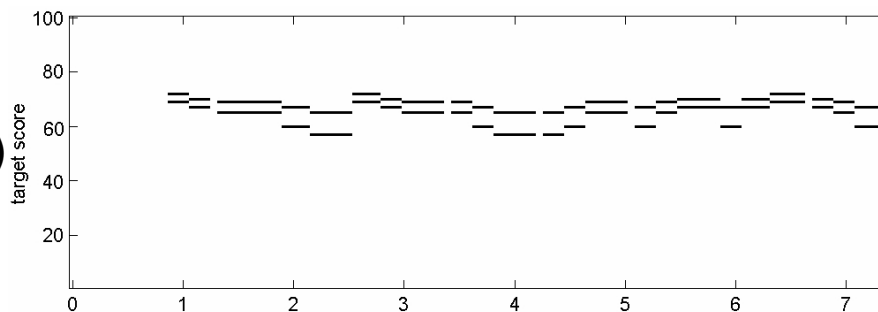
# The Task:

## Polyphonic Music Transcription


- Q: What notes are being played in a piece of musical audio?
- Range of difficulty
- “Easy”: Monophonic - one note at once
  - » Example: 
  - » Can typically solve with autocorrelation etc
- “Tricky”: Polyphonic – many notes at once
  - » Examples: 
  - » Simple solutions no longer work ...


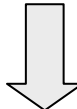
# Example: Lizst Etude

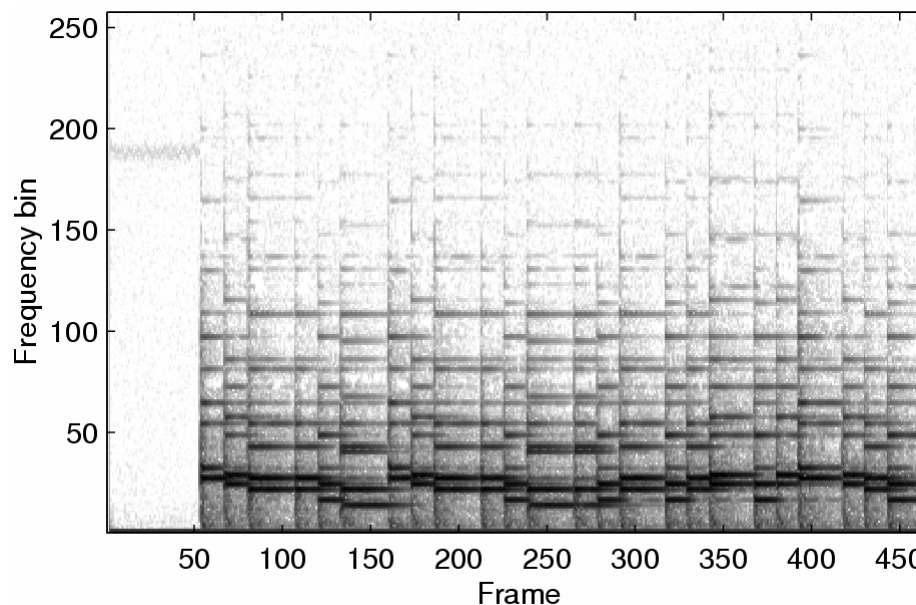
Notes  MIDI  
("Piano Roll")



(Liszt: Etude No. 5 aus Grandes Etudes de Paganini. MIDI from Classical Piano Midi Page <http://www.piano-midi.de>, copyright Bernd Krueger)

Task: Extract notes from e.g. this 

  Synth/sample/FT



# “Knowledge Engineering” Approaches

Design-in knowledge about the problem domain:

- Spectral peak detection (Sterian, 1999)
- Iterative spectral subtraction (Klapuri et al, 2000; Lepain, 1999)
- Networks of adaptive oscillators (Marolt, 2000)
- Spectral dictionaries (Rossi et al, 1997)
- Time-domain waveform dictionaries (Bello, 2003)
- Blackboard models (Martin, 1996; Kashino et al, 1998)

Instead: let's take a data-driven approach

# Features of the Problem

Identify some general features we can use:

- We'll work in the frequency domain
  - » Familiar for signal processing researchers
- Spectra of individual notes add to give observed spectrum (approximately)
  - » Linear generative model
- Notes “more independent” than spectra
  - » Factorial model
  - » Independent Component Analysis (ICA)

# Features of the Problem (cont)

- Positive amount of each note
  - » “Positive/Nonnegative” approaches
- Positive spectrum of each note
  - » “Positive/Nonnegative” approaches

Many of these features also true for other real-world problems (e.g. chemical factor analysis)

(Can also use “sparsity”: On piano, 88 notes, but typ. <11 played at once. Not considered in this talk)

# Linear Generative Model

Observation is weighted sum of sources plus noise

$$\mathbf{x} = \mathbf{A}\mathbf{s} + \mathbf{e}$$

Sometimes called a *Multiple-Cause Model* (Saund, 1995)  
i.e. a component of  $\mathbf{x}$  is “caused” by many components of  $\mathbf{s}$ .  
(C.f. “Winner-takes-all” neural network)

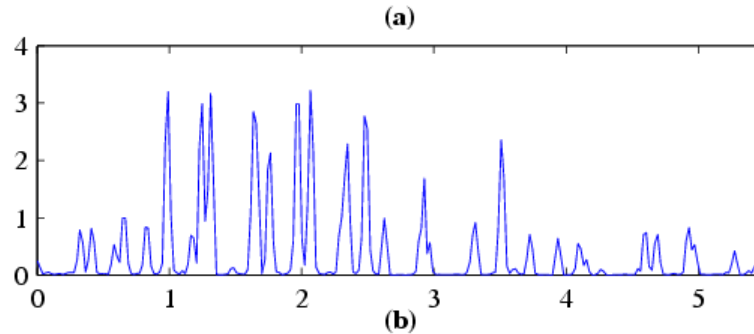
Problem is now, given sequence of observations  $\mathbf{x}$ ,  
to estimate sequence of  $\mathbf{s}$  (and mixing matrix  $\mathbf{A}$ ).

Often breaks into two related problems:

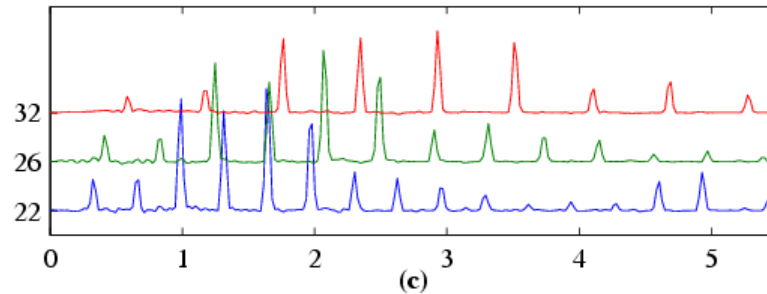
- (1) Inference: If we knew  $\mathbf{A}$ , can we estimate  $\mathbf{s}$  from  $\mathbf{x}$ ?
- (2) Learning: Can we learn  $\mathbf{A}$ ?

# Spectrum decomposition

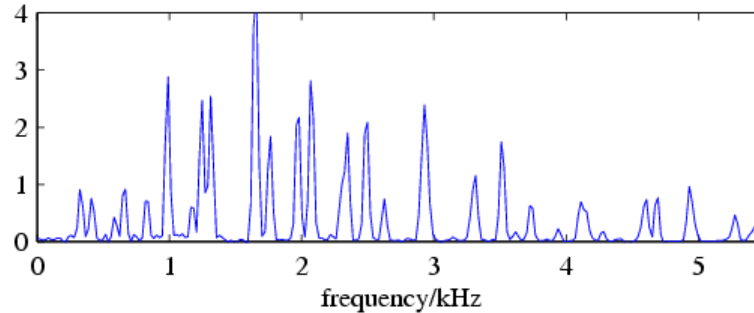
Visualization:  
Original spectrum



Components



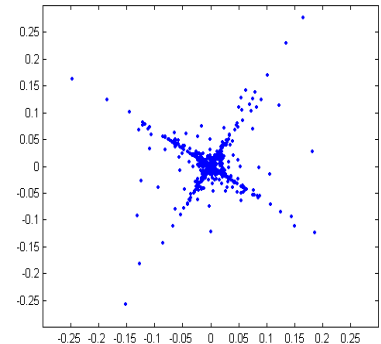
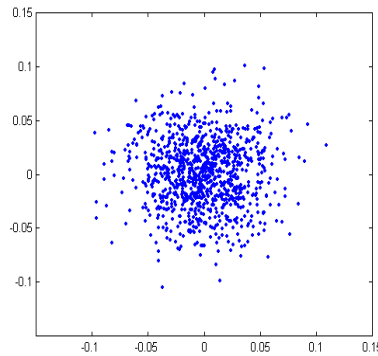
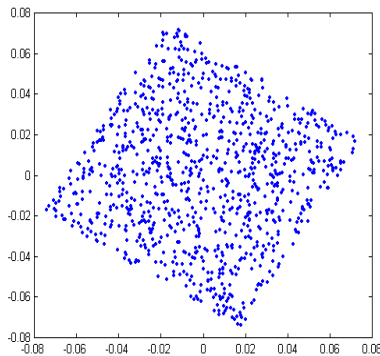
Reconstruction



# Independent Component Analysis (ICA)

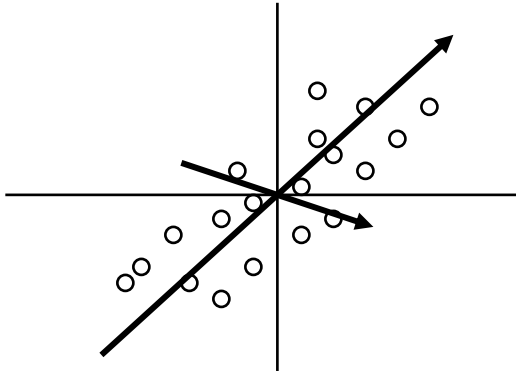
Assume independent  $s$ , e.g. pdf factorises  $p(\mathbf{s}) = \prod_i p(s_i)$

If true, we can find the original sources,  
using e.g. higher-order statistics (if sources non-Gaussian)  
More than just decorrelation...

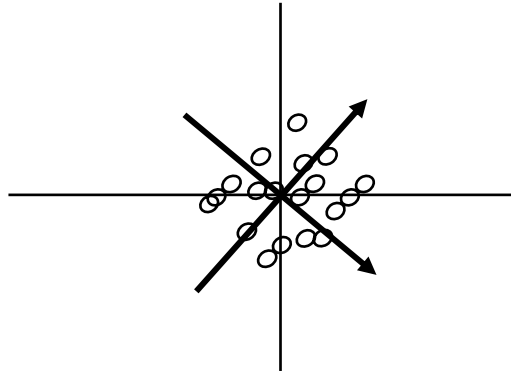


Can “see” the independent source directions, (except for Gaussian case)

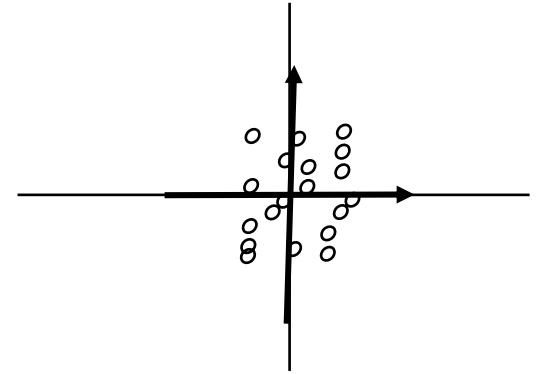
# Typical ICA Procedure



(1) Initial  
Distribution



(2) Decorrelate:  
Form equal variance  
outputs



(3) Rotate:  
Make outputs  
independent

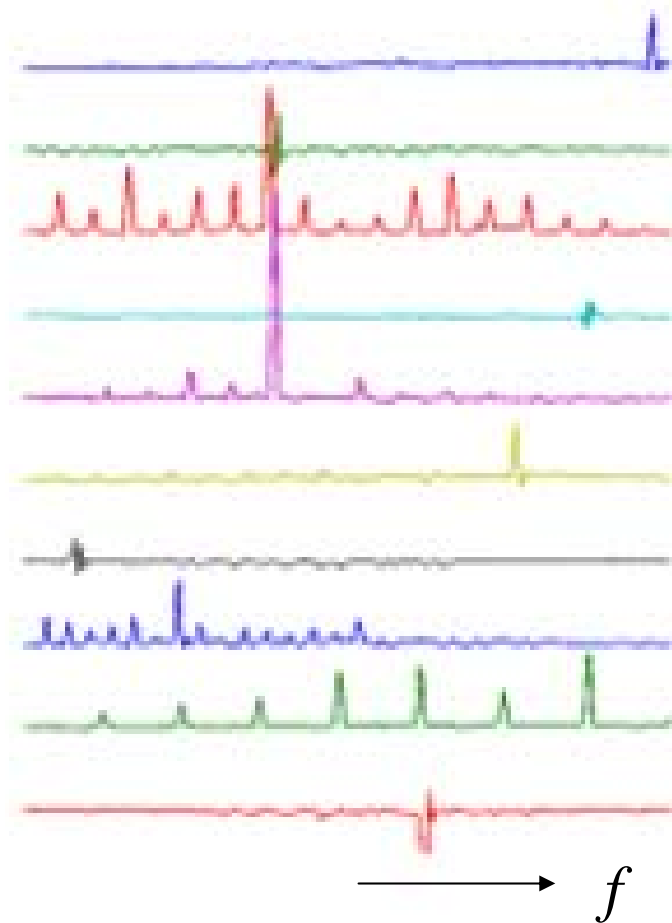
# Q: Does ICA work for this?

A: Partially

For  $\mathbf{A}$  matrix, finds some note spectra.

But some missed,  
and some have undesired  
“negative power”.

So: Try Non-negative ICA  
(NNICA)



# Non-Negative ICA

- Observations of mixed data - generative model

$$\mathbf{X} = \mathbf{A}\mathbf{S}$$

with sources  $\mathbf{S} \in \mathbb{R}^{n \times p}$  and mixing matrix  $\mathbf{A} \in \mathbb{R}^{m \times n}$ .

- Task - to discover the source samples  $\mathbf{S}$  and mixing matrix  $\mathbf{A}$  given only the observations  $\mathbf{X}$ .
- An Underdetermined problem: if  $(\mathbf{A}^*, \mathbf{S}^*)$  is a solution, so is  $(\mathbf{A}^* \mathbf{M}, \mathbf{M}^{-1} \mathbf{S}^*)$  (for invertible  $\mathbf{M}$ )

# Constraints

1. Independence of sources:  $s_{jk}$  sampled from independent random variables  $S_j$ .
2. Non-negativity of sources:  $s_{jk} \geq 0$  for all  $1 \leq j \leq n, 1 \leq k \leq p$ .

Independence alone

→ classical noiseless ICA.

Non-negativity alone (of  $\mathbf{S}$  and  $\mathbf{A}$ )

→ *non-negative matrix factorization* [Lee & Seung, 1999]

Both constraints

→ *non-negative independent component analysis*.

# Pre-whitening

ICA often simplified by *pre-whitening* - transform

$$\mathbf{z} = \mathbf{Q}\mathbf{x}$$

to get identity covariance  $\mathbf{C}_{\mathbf{z}} = E((\mathbf{z} - \bar{\mathbf{z}})(\mathbf{z} - \bar{\mathbf{z}})^T) = \mathbf{I}$ .

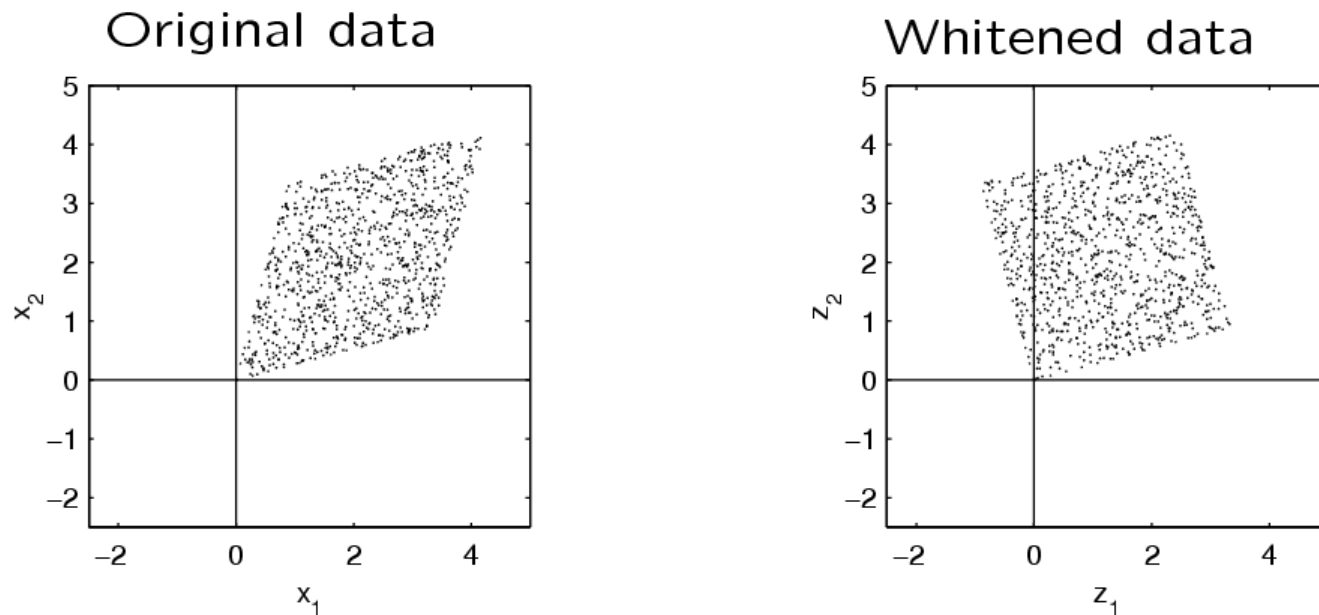
Problem now:

find orthonormal weight matrix  $\mathbf{W}$ , (i.e.  $\mathbf{W}^T\mathbf{W} = \mathbf{W}\mathbf{W}^T = \mathbf{I}_n$ ),  
such that the outputs  $\mathbf{y} = \mathbf{W}\mathbf{z} = \mathbf{W}\mathbf{Q}\mathbf{A}\mathbf{s}$  are independent.

Typical ICA algorithms do this by searching for extremum of contrast function (e.g. kurtosis).

# Whitening non-negative data

To retain non-negativity: whiten without subtracting mean.  
I.e. use  $z = Qx$  not  $z = Q(x - \bar{x})$ .



Suggests: just try to fit the data into +ve quadrant.

# Non-negative condition

Let  $y = Us$  be an orthogonal rotation of the sources (e.g.  $y = Wz = WQAs = Us$ , due to orthonormal rotation  $W$  of observations  $Qx$ ).

Suppose that the sources  $s_i$  are

(i) independent,

(ii) *non-negative*, and

(iii) *well-grounded* (i.e.  $\Pr(s_i < \delta) > 0$  for any  $\delta > 0$ ),

**then**

$U$  is a permutation matrix (i.e. sources are separated)

**if and only if**

all components of  $y$  are non-negative w.p.1.

# Minimize Mean Squared Error

Can show: conditions

“all  $y$  non-negative w.p.1” and

“orthogonal  $\mathbf{W}$ ”

are satisfied by finding the zero of a suitable cost function.

E.g. mean squared reconstruction error

$$J = \frac{1}{2}E(\|\mathbf{z} - \mathbf{W}^T g_+(\mathbf{y})\|)$$

where  $g_+(\mathbf{y})$  is rectified version of  $\mathbf{y} = \mathbf{W}\mathbf{z}$ .

Suggests: construct algorithm to adapt  $\mathbf{W}$  to find min of  $J$ .

# Non-Negative PCA

Based on earlier "nonlinear PCA" algorithms for ICA :

$$\Delta \mathbf{W} = \eta \mathbf{g}(\mathbf{y}) [\mathbf{z} - \mathbf{W} \mathbf{g}(\mathbf{y})]^T$$

We can use "Nonnegative PCA" algorithm :

$$\Delta \mathbf{W} = \eta \mathbf{g}_+(\mathbf{y}) [\mathbf{z} - \mathbf{W}^T \mathbf{g}_+(\mathbf{y})]^T$$

where  $\mathbf{g}_+(\mathbf{y}) = \max(0, \mathbf{y})$ .

But this can be a bit slow.

Can we use Optimization on Manifolds?

# Part 2: Approach using Optimization on Manifolds

# Recap: ICA

Independent Component Analysis (ICA):

- Observations of mixed data - generative model

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) \quad \text{or} \quad \mathbf{X} = \mathbf{A}\mathbf{S}$$

with sources  $\mathbf{S} \in \mathbb{R}^{n \times p}$  and mixing matrix  $\mathbf{A} \in \mathbb{R}^{m \times n}$ .

- Task - to discover the source samples  $\mathbf{S}$  and mixing matrix  $\mathbf{A}$  given only the observations  $\mathbf{X}$ .
- An Underdetermined problem: if  $(\mathbf{A}^*, \mathbf{S}^*)$  is a solution, so is  $(\mathbf{A}^* \mathbf{M}, \mathbf{M}^{-1} \mathbf{S}^*)$  (for invertible  $\mathbf{M}$ )
- To resolve this, use *independence* of sources  $s_j$

# ICA by Pre-whitening

Typical ICA algorithm has two stages:

1. Pre-whitening
2. Orthonormal rotation to maximize non-Gaussianity

(1) In pre-whitening, we transform

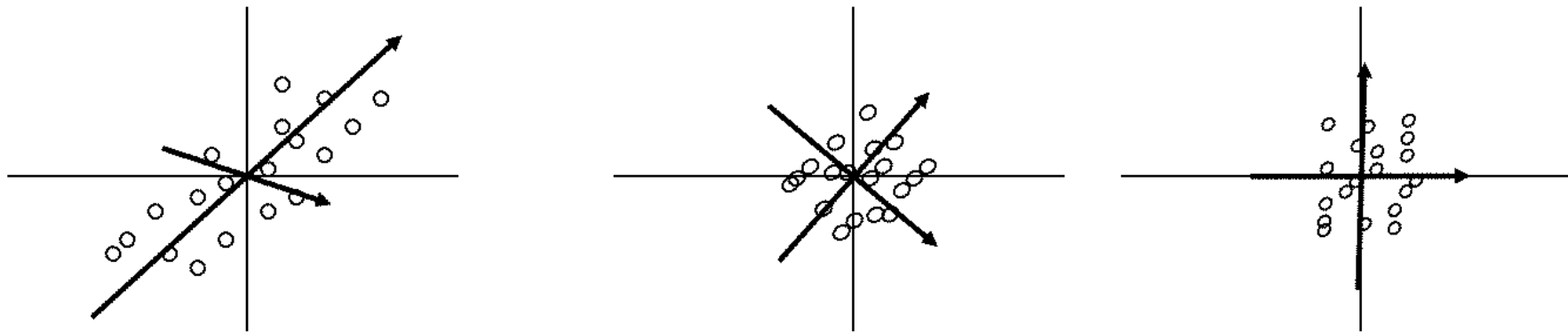
$$\mathbf{z} = \mathbf{Q}\mathbf{x}$$

to get identity covariance  $\Sigma_{\mathbf{z}} = E((\mathbf{z} - \bar{\mathbf{z}})(\mathbf{z} - \bar{\mathbf{z}})^T) = \mathbf{I}$ .

(2) Now want orthonormal weight matrix  $\mathbf{W}$ , (i.e.  $\mathbf{W}^T\mathbf{W} = \mathbf{W}\mathbf{W}^T = \mathbf{I}_n$ ), such that the outputs  $\mathbf{y} = \mathbf{W}\mathbf{z} = \mathbf{W}\mathbf{Q}\mathbf{A}$ s are independent.

Typ. search for e.g. extremum of kurtosis.

# Two-stage ICA Process



1. Decorrelation stage ensures the independent sources are mutually orthogonal
2. Orthogonal rotation stage aligns sources with the output axes

# Recap: Non-negative ICA

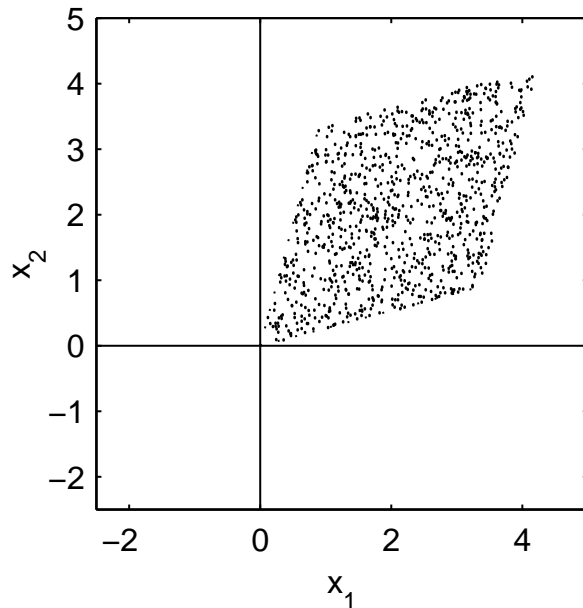
I will use *non-negative ICA*. Here we have:

1. Independence of sources:  $s_{jk}$  sampled from independent random variables  $S_j$ .
  2. Non-negativity of sources:  $s_{jk} \geq 0$  for all  $1 \leq j \leq n$ ,  $1 \leq k \leq p$ .
- Independence alone  $\rightarrow$  classical noiseless ICA.
  - Non-negativity (of  $S$  and  $A$ )  $\rightarrow$  *non-negative matrix factorization* [Lee & Seung, 1999]
  - Both constraints  $\rightarrow$  *non-negative independent component analysis*.

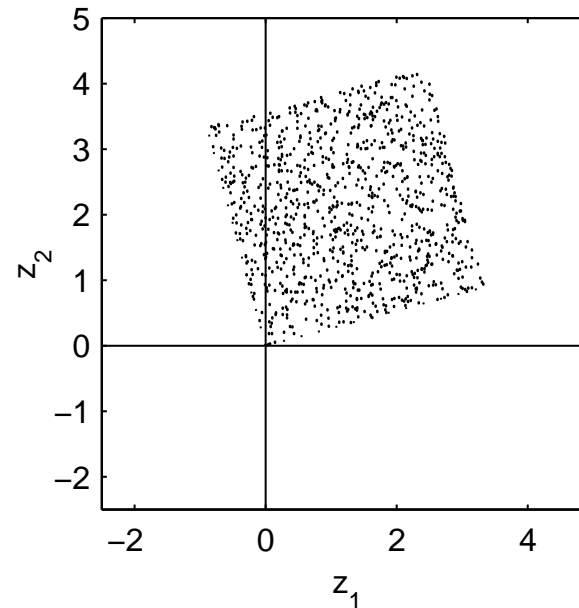
# Whitening of Non-Negative Data

To retain non-negativity: whiten without subtracting mean.  
I.e. use  $z = Qx$  not  $z = Q(x - \bar{x})$ .

Original data



Whitened data



Suggests: just try to fit the data into +ve quadrant.

# Constrained Minimum MSE

- For non-negative ICA, sufficient to find min mean squared reconstruction error

$$J = \frac{1}{2} E(\|\mathbf{z} - \mathbf{W}^T g_+(\mathbf{y})\|)$$

where  $g_+(\mathbf{y})$  is rectified version of  $\mathbf{y} = \mathbf{W}\mathbf{z}$  and  $\mathbf{W}$  is orthonormal.

- So, construct algorithm to adapt  $\mathbf{W}$  to find min of  $J$  for some orthonormal  $\mathbf{W}$ .
- Problem of *Constrained Optimization*
- (Similar for many other ICA methods)

How do we keep  $\mathbf{W}$  to be an *orthogonal* matrix?

# First Attempt: Steepest Descent

- Calculate gradient

$$\nabla_{\mathbf{W}} J \equiv \frac{\partial J}{\partial \mathbf{W}}$$

where  $[\partial J / \partial \mathbf{W}]_{ij} = \partial J / \partial w_{ij}$ ,

- Update  $\mathbf{W}$  by

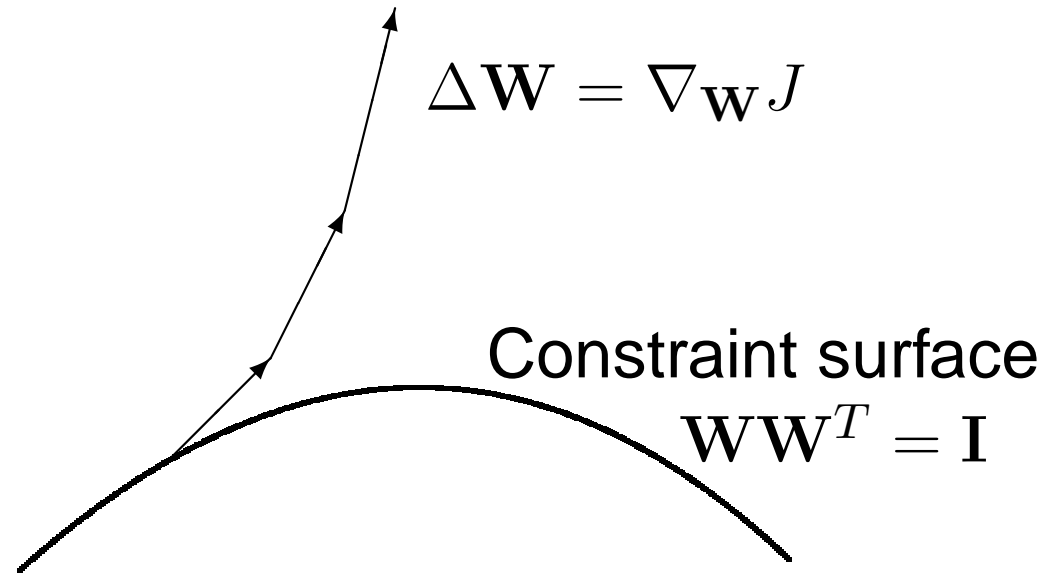
$$\mathbf{W}_{k+1} = \mathbf{W}_k - \eta \nabla_{\mathbf{W}} J$$

for small  $\eta > 0$ . Also write as  $\Delta \mathbf{W} = -\eta \nabla_{\mathbf{W}} J$

E.g. for nonnegative ICA:

$$\nabla_{\mathbf{W}} J \equiv \frac{\partial J}{\partial \mathbf{W}} = E(\mathbf{y} - \mathbf{z}^T) \equiv E(g(\mathbf{y})\mathbf{z}^T)$$

# Steepest Descent



- No attempt to impose orthogonality of  $\mathbf{W}$ .
- How do we do this?

# Simple method: Penalty function

Construct penalty function  $J_{\mathbf{W}}$ ,

- $J_{\mathbf{W}} = 0$  whenever constraint  $\mathbf{W}\mathbf{W}^T = \mathbf{I}_n$  satisfied
- $J_{\mathbf{W}} > 0$  otherwise

New target function  $J_1 = J + \mu J_{\mathbf{W}}$ , new updated

$$\Delta \mathbf{W} = -\eta(\nabla_{\mathbf{W}} J + \mu \nabla_{\mathbf{W}} J_{\mathbf{W}})$$

E.g. for our NNICA task, can use

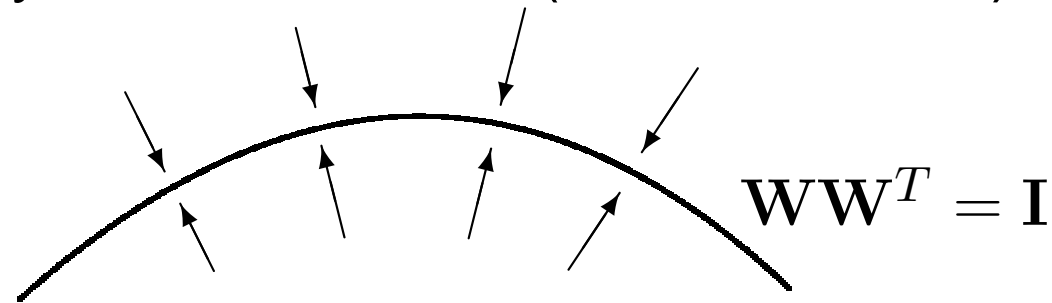
$$J_{\mathbf{W}} = \frac{1}{2} \|\mathbf{W}\mathbf{W}^T - \mathbf{I}_n\|_F^2 = \frac{1}{2} \text{trace}((\mathbf{W}^T \mathbf{W} - \mathbf{I}_n)(\mathbf{W}^T \mathbf{W} - \mathbf{I}_n))$$

$$\Delta \mathbf{W} = -\eta(E(\mathbf{y} - \mathbf{z}^T) + 2\mu \mathbf{W}(\mathbf{W}^T \mathbf{W} - \mathbf{I}_n))$$

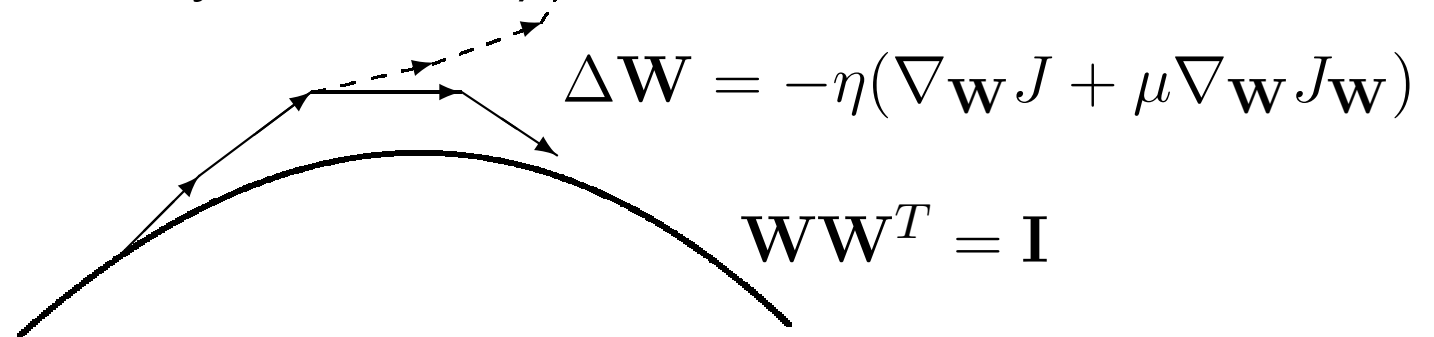
# Penalty Function

Penalty function is supposed to force  $\mathbf{W}$  onto the constraint...

Penalty function “force” (–ve derivative)

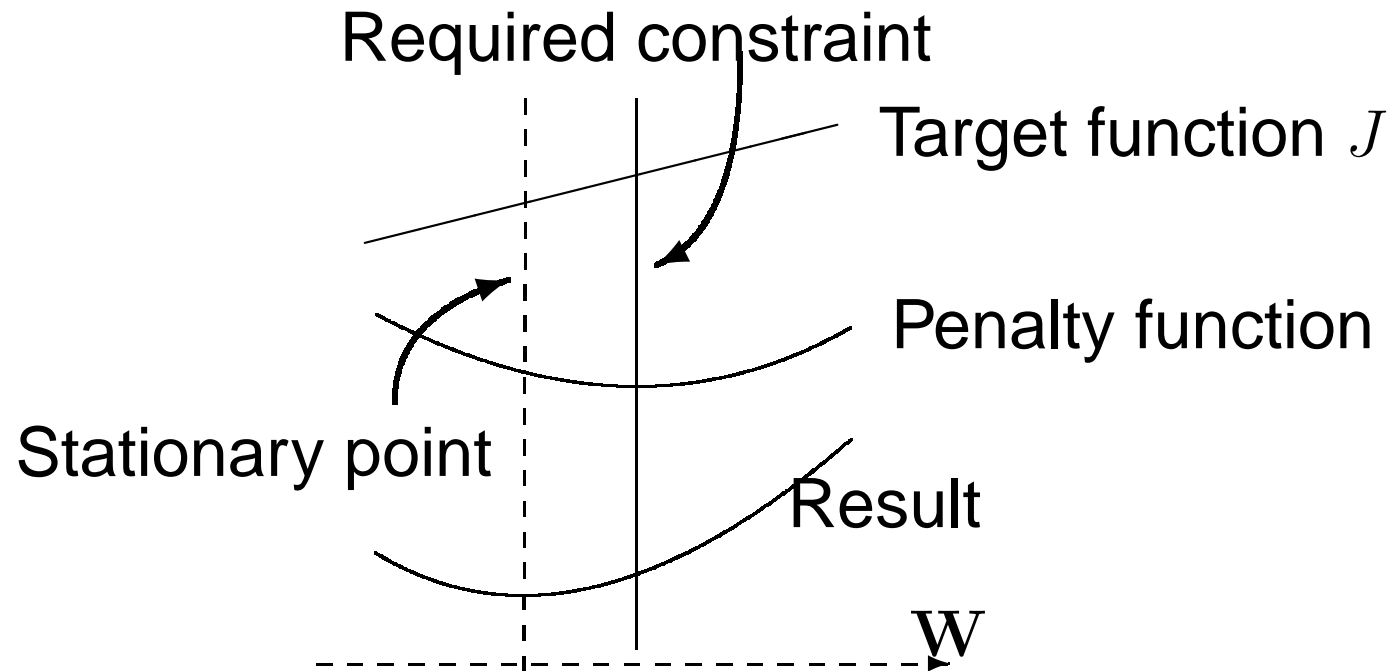


(Penalty too weak)



# Penalty Function (cont)

... but can easily miss ...



Can we *constrain*  $W$  to remain orthogonal?

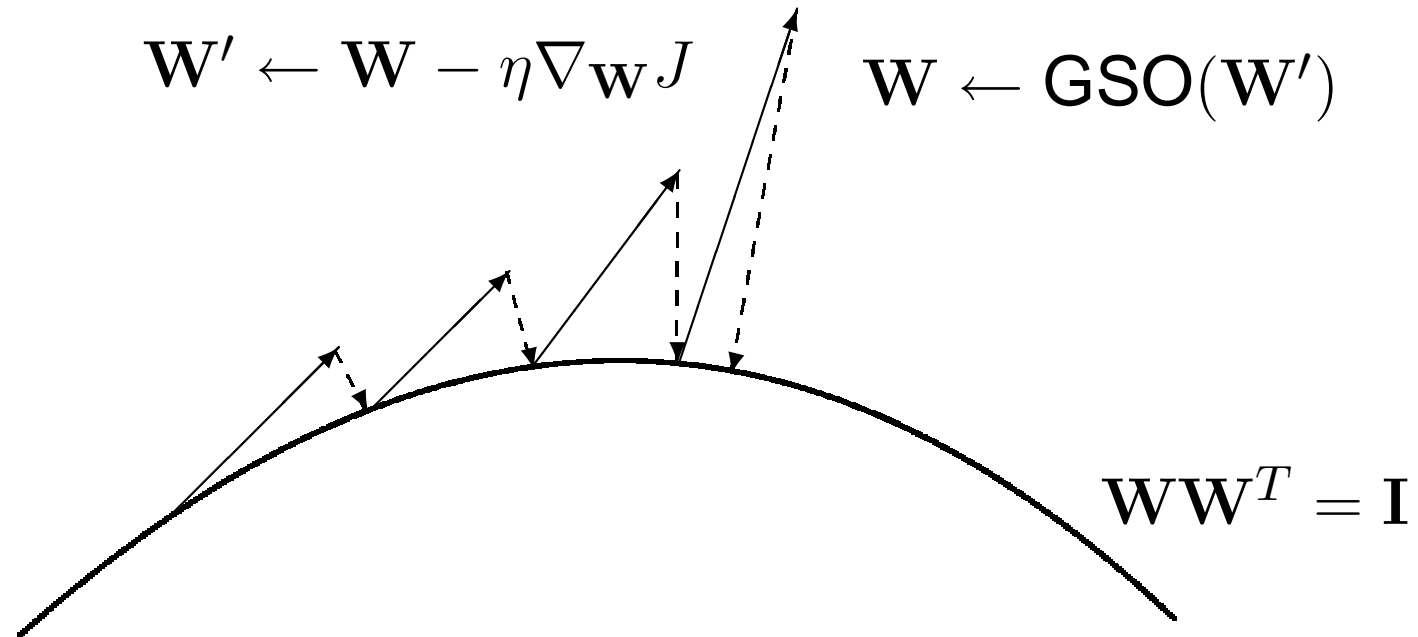
# Repeated Constraint Application

Another approach - make each update a 2-step process:

1. Update according to e.g. (unconstrained) steepest descent
2. Modify  $W$  to fit the constraint

E.g. can repeatedly use Gram-Schmidt Orthogonalization (GSO) after each update.

# Repeated Orthogonalization



Note: The GSO operation is not an “additive” movement

But: Wasted effort (and accuracy) moving away from constraint surface.

# Tangent direction updates

- Try to move “along” (tangent to) the surface.
- Require the constraint to remain unchanged, i.e.  
 $\mathbf{0} = d/dt(\mathbf{W}^T \mathbf{W} - \mathbf{I}_n) = (d\mathbf{W}/dt)^T \mathbf{W} + \mathbf{W}^T (d\mathbf{W}/dt)$
- Leads to modified update equation

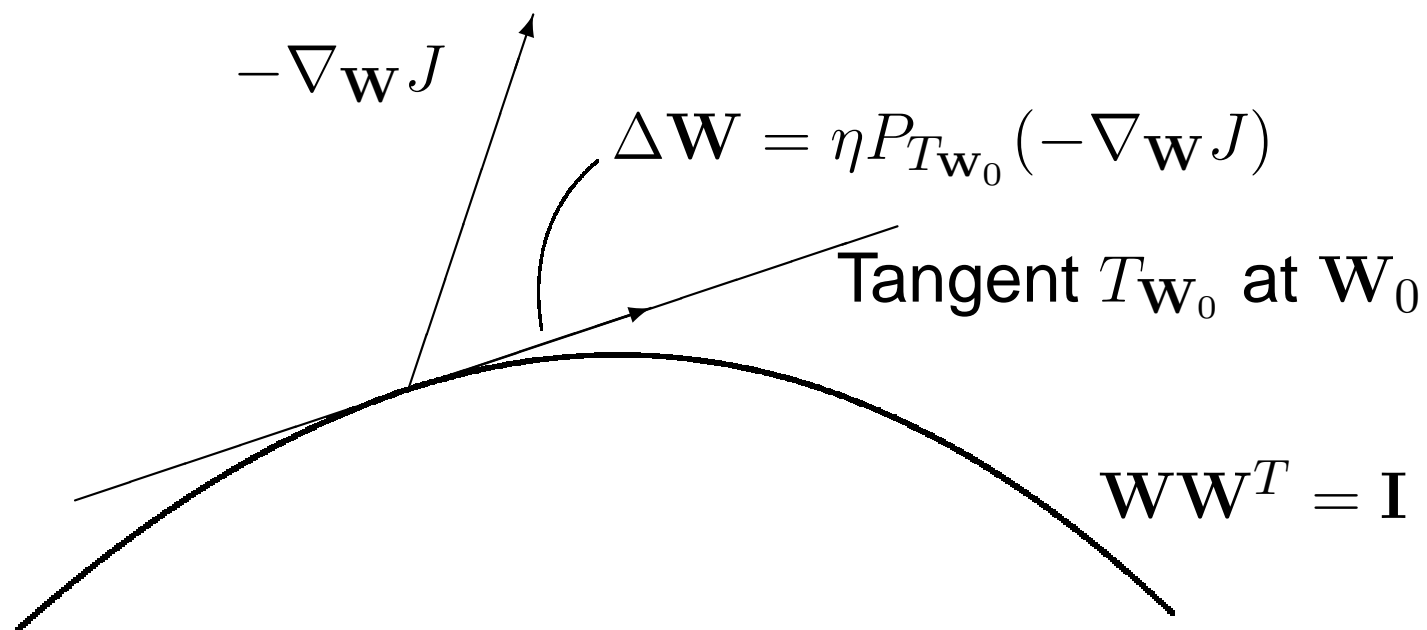
$$d\mathbf{W}/dt|_{\text{Orth}} = \frac{1}{2}((d\mathbf{W}/dt)\mathbf{W}^T \mathbf{W} - \mathbf{W}(d\mathbf{W}/dt)^T \mathbf{W})$$
$$\Delta \mathbf{W} = -\frac{1}{2}\eta((\nabla_{\mathbf{W}} J)\mathbf{W}^T \mathbf{W} - \mathbf{W}(\nabla_{\mathbf{W}} J)^T \mathbf{W})$$

For non-negative ICA [Oja & Plumbley 2004]:

$$\Delta \mathbf{W} = -\eta(E(\mathbf{y}_- \mathbf{y}_-^T) - E(\mathbf{y} \mathbf{y}_-^T)) \mathbf{W}$$

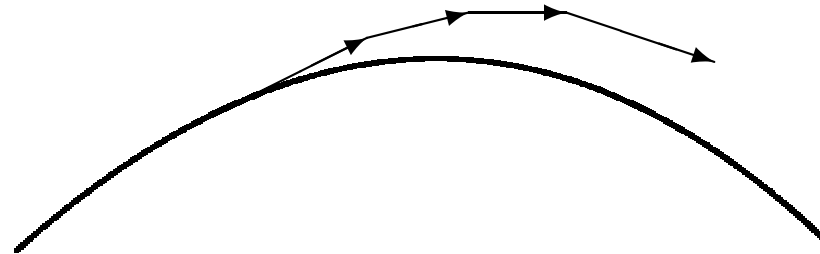
# Tangent Update

Using the tangent removes the component of update away from the surface ...



# Tangent drift

... but, we can still drift away from the surface.



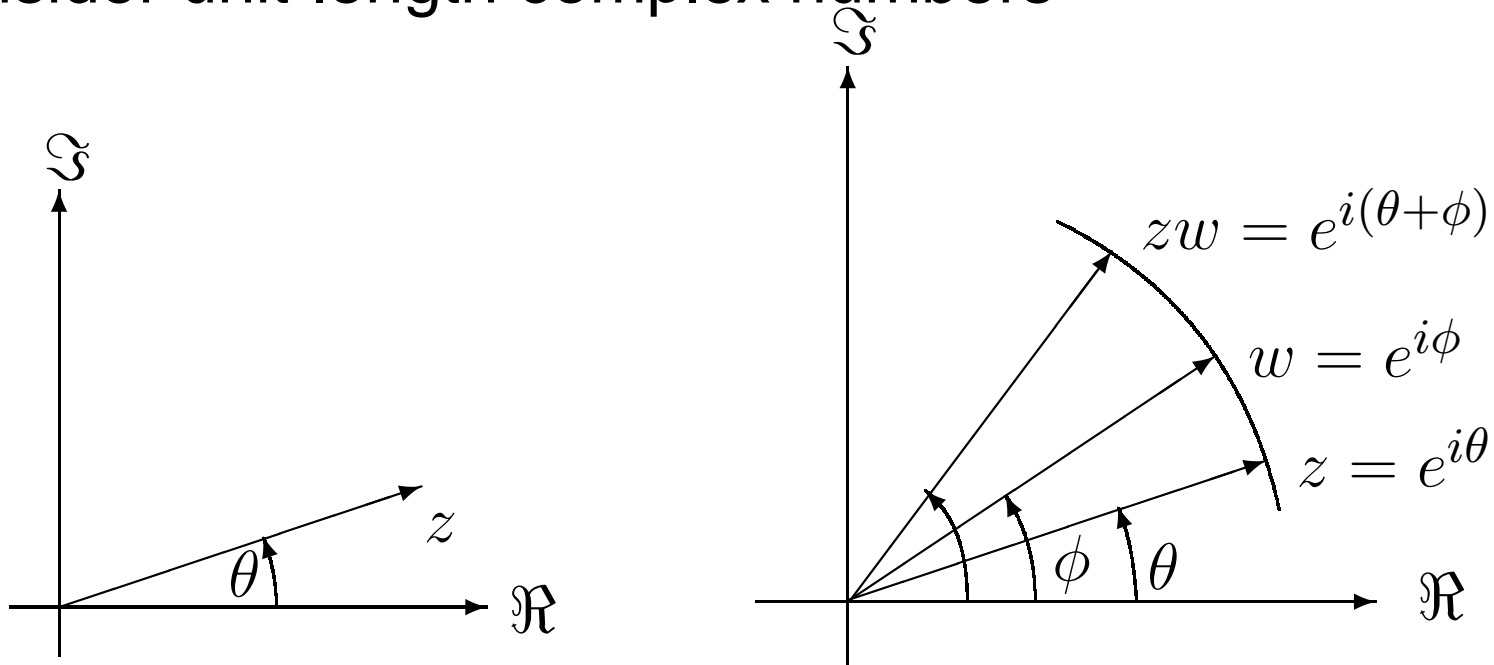
There are ways of dealing with these various problems, e.g.

- Lagrange multiplier methods
- Tangent updates with occasional GSO
- Self-stabilising algorithms (Douglas, 2000)

Instead: use *Lie groups* to avoid violating constraint.

# Lie groups

Consider unit-length complex numbers



- Multiply  $z = e^{i\theta}$  by  $w = e^{i\phi}$  we get  $zw = e^{i(\theta+\phi)}$
- I.e. product is also a unit-length complex number

# Reminder: Group operations

1. Closed under operation: if  $z, w \in G$ , then  $zw = y \in G$ ;
  2. Associativity:  $z(wy) = (zw)y$  for  $z, w, y \in G$ ;
  3. Identity element:  $I \in G$ , such that  $Iz = zI = z$ ;
  4. Each element has inverse:  $z^{-1}$  such that  $z^{-1}z = zz^{-1} = I$ ;
- Also: Unit-length complex numbers are Abelian (commutative)
  - Multiplication of unit-length complex corresponds to addition of angles (modulo  $2\pi$ )
  - This group is *smooth*: locally it looks like the real line  $\mathbb{R}$ .

This smoothness means it is a *Lie group*.

# Manifolds

Recap from earlier in the course ...

A *Manifold* is really a set where

- we can put a local coordinate system  $\mathbb{R}^m$  for some  $m$  on any small neighbourhood, and
- we can join these local coordinate systems together, with overlaps, but without anything “nasty” happening.

The number  $m$  gives the dimensionality of the manifold.

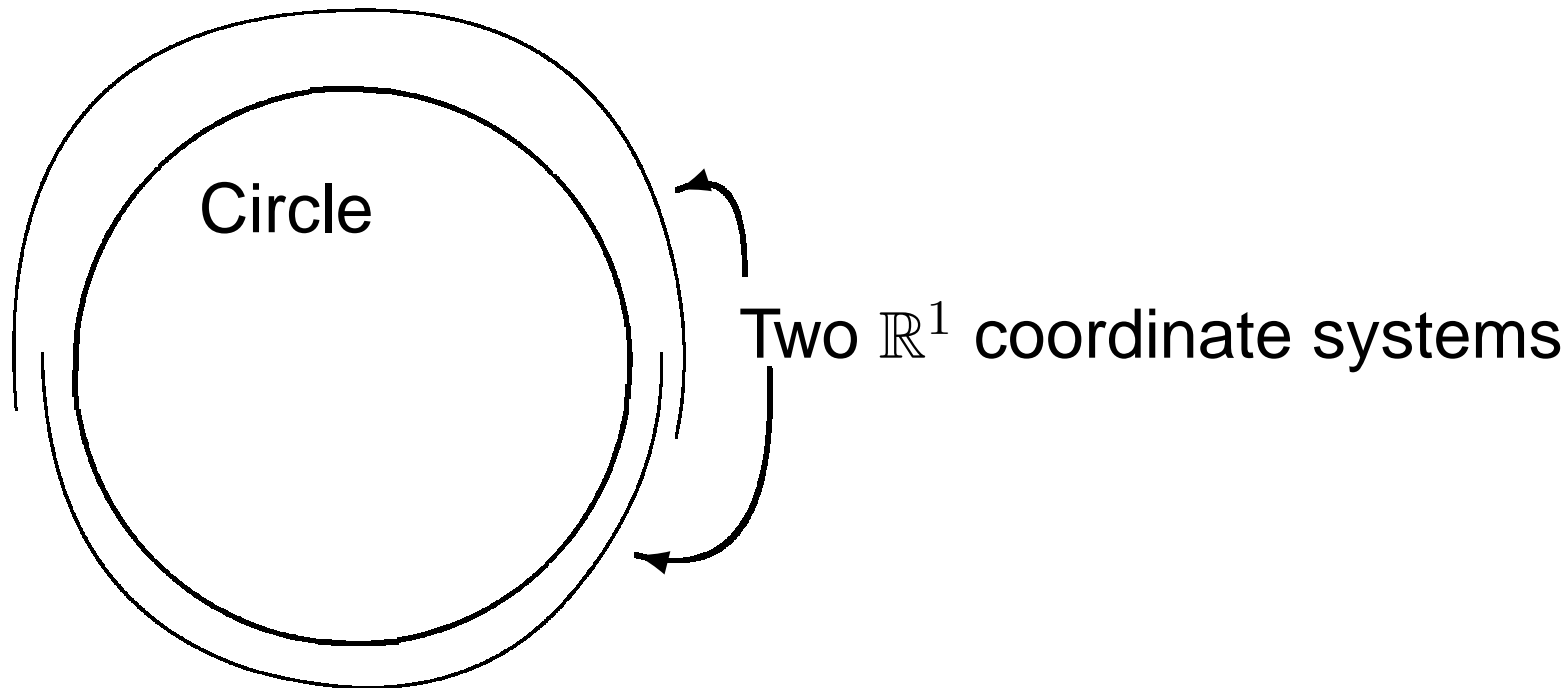
Since we can move about in  $\mathbb{R}^m$ , these local coordinates mean we could (in theory) move about locally on our manifold.

(This is what this Summer School is all about!)

# Manifold Example 1: Circle

Consider circle (and unit-length complex numbers).

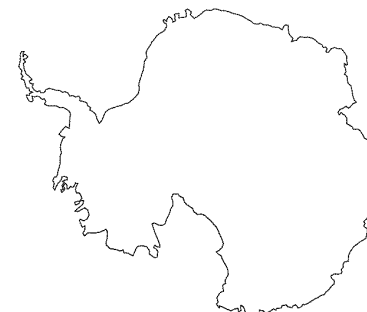
Locally looks like  $\mathbb{R}^1$ . Can cover with e.g. two segments of  $\mathbb{R}^1$  with smooth overlaps.



# Manifold Example 2: Sphere (surface)

Consider the Earth. Locally looks like  $\mathbb{R}^2$ .

Series of 2-D *charts* covering the globe form an *atlas*.



Maps (“charts”) from WorldAtlas.com

# Manifold of Orthonormal Matrices

Set of orthogonal matrices  $W$  with  $WW^T = I_n$  forms a manifold with  $n(n - 1)/2 < n^2$  dimensions (degrees of freedom).

(Special case of *Stiefel manifold*)

- Advantage: Smaller space to search over.
- Disadvantage: Oddly shaped space.

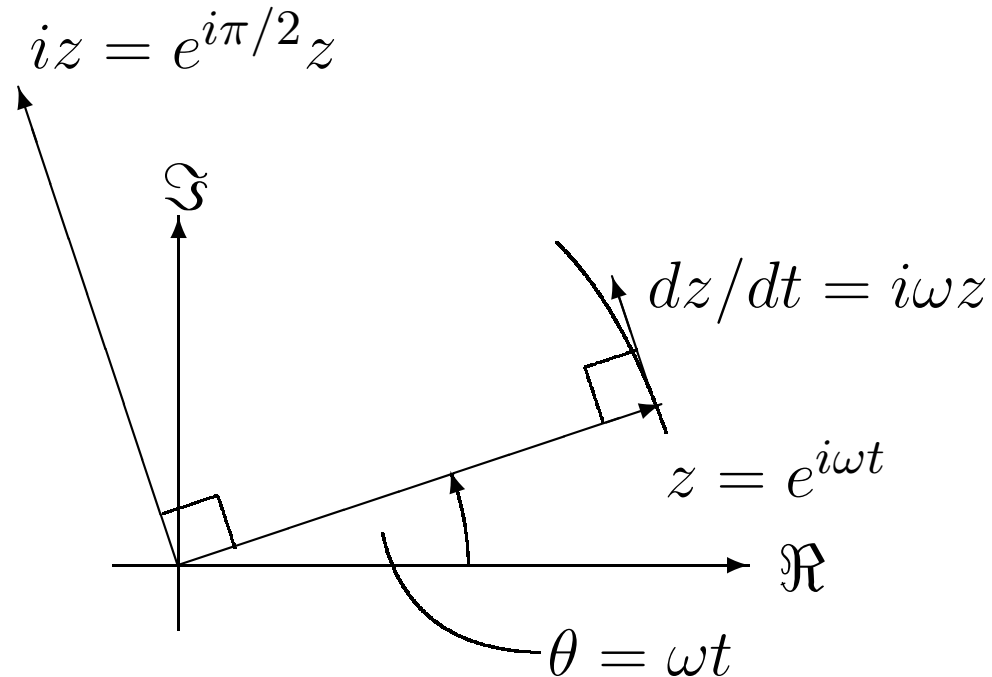
Q: How do we “move about” on our manifold? It is not “flat”!

A: Don't use *addition* to move about, use *Lie group operation*.

For small movements, investigate differentiation...

# Differentiating on Lie Group

Since we have a local smooth coordinate system, we can differentiate functions on it.



Example: Derivative of  $z = \exp(i\omega t)$ .

Notice the derivative is *tangent to the manifold surface*.

# Lie group of orthogonal matrices

- Real orthogonal matrices form a group called  $O(n)$ .
- Can check group axioms: e.g.
  1. if  $W$  and  $Z$  are orthogonal ( $W, Z \in O(n)$ ), then for  $V = WZ$  we get  $V^T V = Z^T W^T W Z = Z^T Z = I_n$  so  $V \in O(n)$ .
  2. Have an  $I_n \in O(n)$ , etc. etc.
- For  $n = 2$ , matrices are 1 of two types:

$$(a) \quad W = \begin{pmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{pmatrix} \quad (b) \quad W = \begin{pmatrix} \cos \theta & -\sin \theta \\ -\sin \theta & -\cos \theta \end{pmatrix}$$

- Type (a) have determinant 1
- These form the “special” orthogonal matrices,  $SO(n)$ .

# Aside: Givens Rotations

Some algorithms use *Givens rotations* to retain the orthogonality constraint (see e.g. Comon, 1994):

$$\begin{pmatrix} u_{i_1}(k+1) \\ u_{i_2}(k+1) \end{pmatrix} = \begin{pmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{pmatrix} \begin{pmatrix} u_{i_1}(k) \\ u_{i_2}(k) \end{pmatrix}$$

where  $(i_1, i_2)$  is an axis pair to rotate over.

This is one type of movement over  $SO(n)$  (equivalent to moving parallel to an axis in  $\mathbb{R}^n$ ).

# Moving about on $SO(2)$

- For  $SO(2)$  we can simply add angles:

$$\begin{pmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{pmatrix} \begin{pmatrix} \cos \phi & \sin \phi \\ -\sin \phi & \cos \phi \end{pmatrix} = \begin{pmatrix} \cos(\theta + \phi) & \sin(\theta + \phi) \\ -\sin(\theta + \phi) & \cos(\theta + \phi) \end{pmatrix}$$

- For  $n = 2$  this is an Abelian (commutative) group.
- Behaves like (*isomorphic to*) unit-length complex nos.

So, to move about on  $SO(2)$ :

1. Given some  $\mathbf{W} = \begin{pmatrix} c & s \\ -s & c \end{pmatrix} \in SO(2)$ , calculate  $\theta = \arctan(s, c)$
2. Move from angle  $\theta$  to a new angle  $\theta' = \theta + \Delta\theta$
3. Transform to new matrix  $\mathbf{W}' = \begin{pmatrix} \cos \theta' & \sin \theta' \\ -\sin \theta' & \cos \theta' \end{pmatrix} \in SO(2)$ .

# Derivatives of $\mathbf{W} \in \text{SO}(2)$

Let  $\theta = t\phi$ . Differentiate  $\mathbf{W} = \begin{pmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{pmatrix}$  wrt  $t$ , we get

$$\frac{d}{dt}\mathbf{W} = \begin{pmatrix} -\sin \theta & \cos \theta \\ -\cos \theta & -\sin \theta \end{pmatrix} \cdot \phi = \begin{pmatrix} 0 & \phi \\ -\phi & 0 \end{pmatrix} \mathbf{W}$$

Can check  $\frac{d}{dt}(\mathbf{W}^T \mathbf{W} - \mathbf{I}_n) = 0$  (i.e. tangent) since

$$\begin{aligned} \mathbf{W}^T (d\mathbf{W}/dt) + (d\mathbf{W}/dt)^T \mathbf{W} &= \mathbf{W}^T \begin{pmatrix} 0 & \phi \\ -\phi & 0 \end{pmatrix} \mathbf{W}^T + \mathbf{W}^T \begin{pmatrix} 0 & -\phi \\ \phi & 0 \end{pmatrix} \mathbf{W}^T \\ &= \mathbf{0} \end{aligned}$$

- For scalars, if  $dz/dt = bz$  then  $z = \exp(tb)$ .
- Same for these matrices?

# Matrix exponential

Define matrix exponential in similar way to scalar:

$$\exp(t\mathbf{B}) = \mathbf{I} + t\mathbf{B} + \frac{t^2\mathbf{B}^2}{2!} + \dots + \frac{t^k\mathbf{B}^k}{k!} + \dots$$

$$\frac{d}{dt} \exp(t\mathbf{B}) = \mathbf{0} + \mathbf{B} + \mathbf{B} \frac{t\mathbf{B}}{1!} + \dots + \mathbf{B} \frac{t^{k-1}\mathbf{B}^{k-1}}{(k-1)!} + \dots = \mathbf{B} \exp(t\mathbf{B})$$

$$\mathbf{W} = \begin{pmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{pmatrix} = \exp \Theta \quad \text{where} \quad \Theta = \begin{pmatrix} 0 & \theta \\ -\theta & 0 \end{pmatrix}$$

- Tangent space  $T_{\mathbf{W}}$  (space of derivatives) on  $SO(2)$  at  $\mathbf{W}$  is set of matrices  $\Psi\mathbf{W}$  where  $\Psi^T = -\Psi$  is skew-symmetric.

# Optimization over $SO(2)$

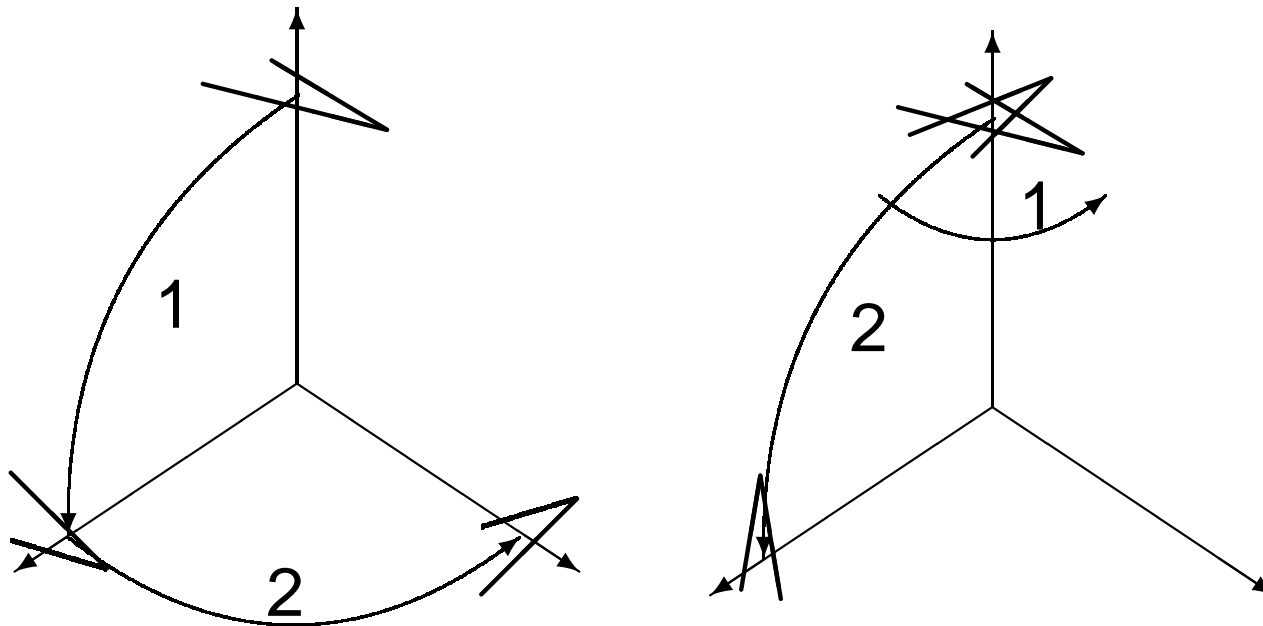
- Matrices  $W \in SO(2)$  can be parameterized by one parameter  $u$  in  $W = \exp(u\Phi)$  for constant skew-symmetric matrix  $\Phi$ .
- So, calculate  $dJ/du$  and change  $u$  to reduce  $J$ .
- Steepest descent in  $u$ :  
 $u(t+1) = u(t) + \Delta u$  where  $\Delta u = -\eta dJ/du$ .
- Corresponding update for  $W$ :  
 $W(t+1) = RW(t)$  where  $R = \exp(\Delta u \cdot \Phi)$ .

Can also use 2nd-order methods - e.g. local Fourier expansion.

(More on this later)

# $n > 3$ : The commutation problem

- Multiplication of matrices  $A, B$  is commutative if and only if they share all eigenvectors.
- Illustration for multiplication by orthogonal rotations in  $SO(3)$ .



# Lie Bracket

- Non-commutation true in general for small rotations:

$$\exp(\epsilon \mathbf{A}) \exp(\epsilon \mathbf{B}) - \exp(\epsilon \mathbf{B}) \exp(\epsilon \mathbf{A}) = \epsilon^2 [\mathbf{A}, \mathbf{B}] + O(\epsilon^3)$$

with commutator  $[\mathbf{A}, \mathbf{B}] = \mathbf{A}\mathbf{B} - \mathbf{B}\mathbf{A}$ .

- For  $\mathbf{A}, \mathbf{B}$  skew-symmetric, have

$$[\mathbf{A}, \mathbf{B}]^T = \mathbf{B}^T \mathbf{A}^T - \mathbf{A}^T \mathbf{B}^T = \mathbf{B}\mathbf{A} - \mathbf{A}\mathbf{B} = -[\mathbf{A}, \mathbf{B}]$$

so the commutator (“bracket”) is also skew-symmetric.

- *Lie algebra*: vector space (can add, and can multiply by scalars) with Lie bracket operator satisfying:

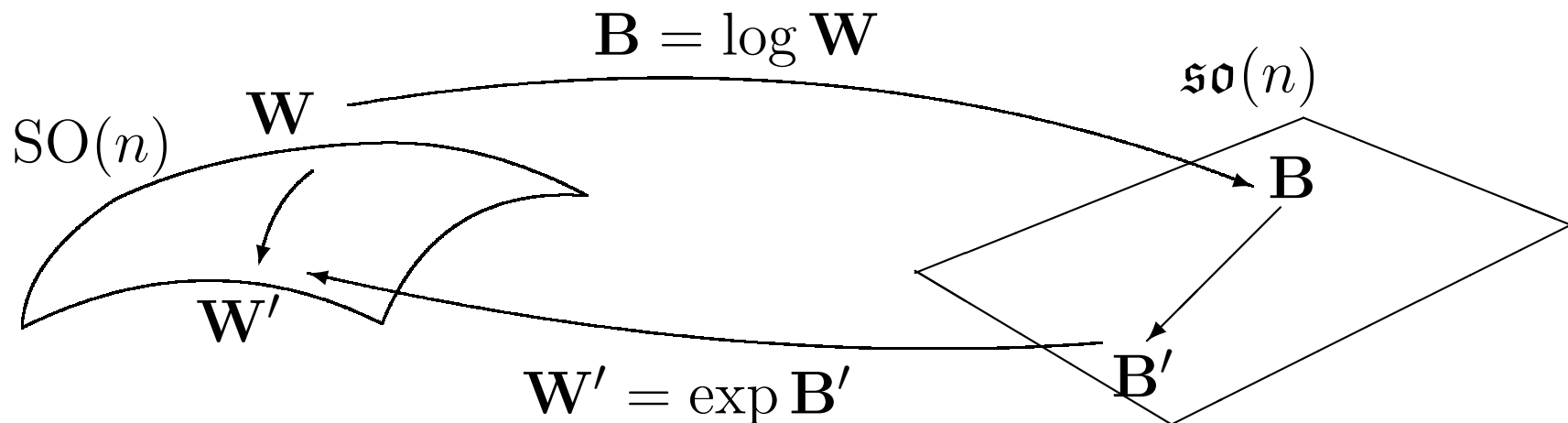
$$[\mathbf{A}, \mathbf{A}] = \mathbf{0}$$

$$[\mathbf{A} + \mathbf{B}, \mathbf{C}] = [\mathbf{A}, \mathbf{C}] + [\mathbf{B}, \mathbf{C}]$$

$$[\mathbf{A}, [\mathbf{B}, \mathbf{C}]] + [\mathbf{B}, [\mathbf{C}, \mathbf{A}]] + [\mathbf{C}, [\mathbf{A}, \mathbf{B}]] = \mathbf{0}$$

# Lie algebras of Lie groups

- Skew-symmetric matrices in the Lie algebra  $\mathfrak{so}(n)$  are related to those in the Lie group  $SO(n)$ : *exponential* of a matrix  $\mathbf{B} \in \mathfrak{so}(n)$  is a matrix in  $SO(n)$ , i.e.  $\mathbf{B} \in \mathfrak{so}(n) \mapsto \exp(\mathbf{B}) \in SO(n)$ .
- Key to *Lie group methods* to solve differential equations (Iserles, Munthe-Kaas & Zanna; 2000).



Easy to stay in  $\mathfrak{so}(n)$ , since addition works.

# Modified Lie group update method

- Realize that  $W' = RW$  for some  $R \in SO(n)$ .
- Moving from  $W$  to  $W'$  equiv. to moving from  $I_n$  to  $R$ .

Modified update method:

1. Start at  $0_n \in \mathfrak{so}(n)$ , equivalent to  $I_n \in SO(n) = \exp(0_n)$
2. Move about in  $\mathfrak{so}(n)$  from  $0_n$  to  $B \in \mathfrak{so}(n)$
3. Use  $\exp(\cdot)$  to map back into  $SO(n)$ , giving  $R = \exp(B)$
4. Calculate  $W' = RW = \exp(B)W \in SO(n)$ .

Avoids logarithm.

# Steepest descent in $\mathfrak{so}(n)$

- For steepest descent, need to know what “steepest” means.
- Simplest to define inner product and length (norm) as:  
 $\langle \mathbf{B}, \mathbf{H} \rangle = \sum_{ij} b_{ij} h_{ij} / 2$  with  $l_{\mathbf{B}}^2 = \langle \mathbf{B}, \mathbf{B} \rangle$ .  
Factor of  $1/2$  avoids double-counting dimensions.
- Space with this type of norm is a *Hilbert space*.
- So for gradient in  $\mathbf{B}$ -space we get (eventually)

$$\nabla_{\mathbf{B}} J = (\nabla_{\mathbf{W}} J) \mathbf{W}^T - \mathbf{W} (\nabla_{\mathbf{W}} J)^T$$

For non-negative ICA (using  $\nabla_{\mathbf{W}} J = E(\mathbf{y}_- \mathbf{z}^T)$ ) we get

$$\nabla_{\mathbf{B}} J = 2 \text{skew}(E(\mathbf{y}_- \mathbf{z}^T) \mathbf{W}^T) = E(\mathbf{y}_- \mathbf{y}^T) - E(\mathbf{y} \mathbf{y}_-^T)$$

# Geodesic flow

- Simple update method: take small step in  $\mathfrak{so}(n)$ .
- Equiv to moving in  $SO(n)$  from  $\mathbf{W}_k$  to

$$\mathbf{W}_{k+1} = \exp(-\eta \mathbf{G}) \mathbf{W}_k$$

where

$$\mathbf{G} = \nabla_{\mathbf{B}} J|_{\mathbf{B}=\mathbf{0}} = (\nabla_{\mathbf{W}} J) \mathbf{W}^T - \mathbf{W} (\nabla_{\mathbf{W}} J)^T.$$

- This is the *geodesic flow* method introduced to ICA by Nishimori [1999].
- Possible issues:
  1. Cost of calculation of matrix exponential
  2. Numerical issues for small  $\eta$ .

# Searching over one-parameter subgroups

- We can move in any direction  $B = tH$  in our Lie algebra
- All steps along this “line” will commute.
- The set of matrices  $\mathfrak{g}_H = \{tH | t \in \mathbb{R}\}$  is a Lie algebra itself, called a *one-parameter subalgebra* of  $\mathfrak{so}(n)$ .
- So: perform a “line search” along this one-parameter subalgebra.
- This has been called *geodesic search*

# One-parameter subalgebra search

Outline of method:

1. Start at  $\mathbf{W}_k(0) = \mathbf{R}(0)\mathbf{W}(0)$  where  $\mathbf{R}(0) = \mathbf{I}_n \in \text{SO}(n)$
2. Choose a search direction  $\mathbf{H}$  in our Lie algebra  $\mathfrak{so}(n)$
3. Search along the points in the one parameter subalgebra  $t\mathbf{H}$ ,  
corresponding to points in one-parameter subgroup  $\mathbf{W}_k(t) = \mathbf{R}(t)\mathbf{W}_k(0)$  where  $\mathbf{R}(t) = \exp(t\mathbf{H})$ ,  
to reduce  $J(\mathbf{R}(t)\mathbf{W}_k(0))$
4. Update  $\mathbf{W}_{k+1}(t) = \mathbf{R}(t)\mathbf{W}_k(0)$  and start again with a new “line” search.

# One-parameter subalgebra search (2)

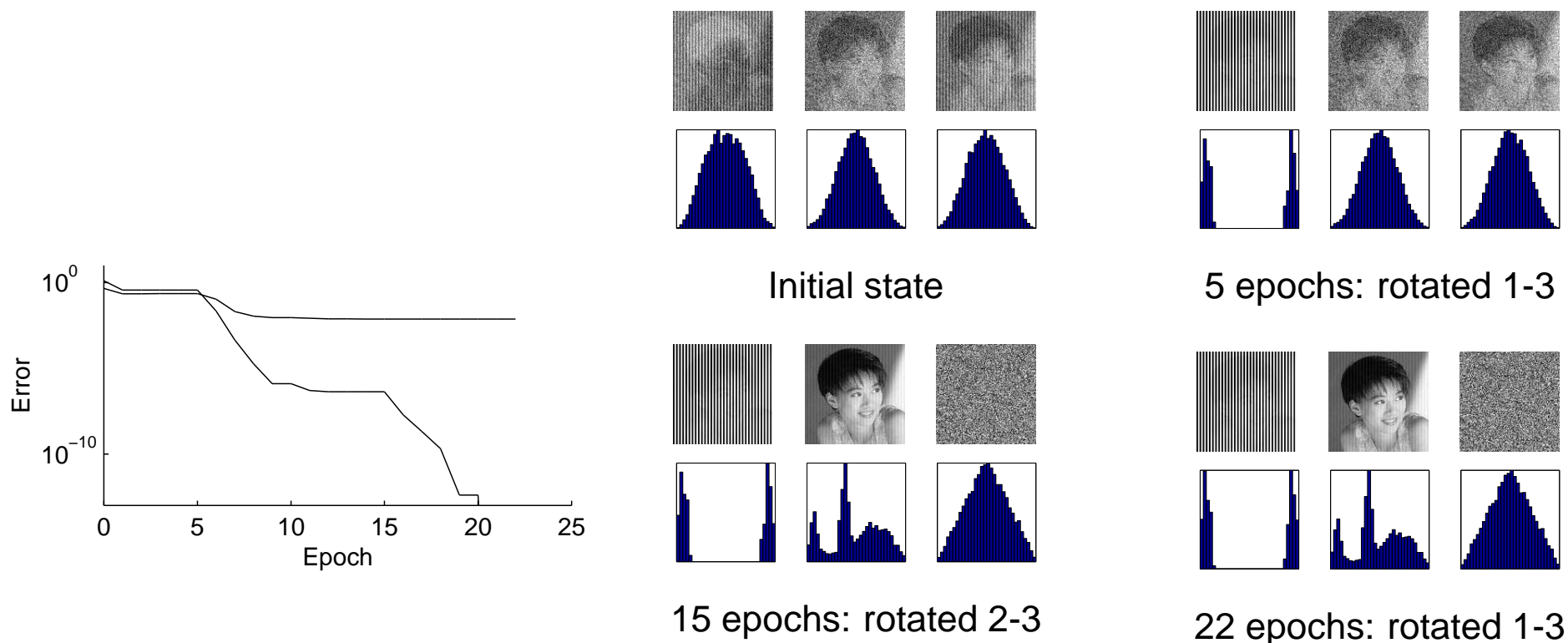
Turning this into an algorithm we get:

1. Calculate  $\mathbf{G} = \nabla_{\mathbf{B}} J$  and  $\theta = |\mathbf{G}|$ . If  $\theta$  is small or zero exit, otherwise calculate  $\mathbf{H} = -\mathbf{G}/\theta$ .
2. Search along  $\mathbf{R}(t) = \exp(t\mathbf{H})$  using  $\mathbf{y} = \mathbf{R}\mathbf{W}\mathbf{z}$  to find a minimum (or near-minimum) of  $J$  at  $t = t^*$ .
3. Update  $\mathbf{W}_{k+1} = \mathbf{R}(t^*)\mathbf{W}_k$
4. Repeat from step 1 until  $\theta$  is small enough to exit.

As well as line search methods, can also construct conjugate gradient methods (Edelman, Arias, Smith, 1998).

# Image separation example

One-parameter (geodesic) search along axis pair rotations:

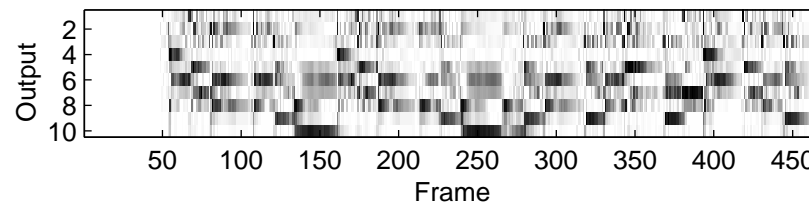
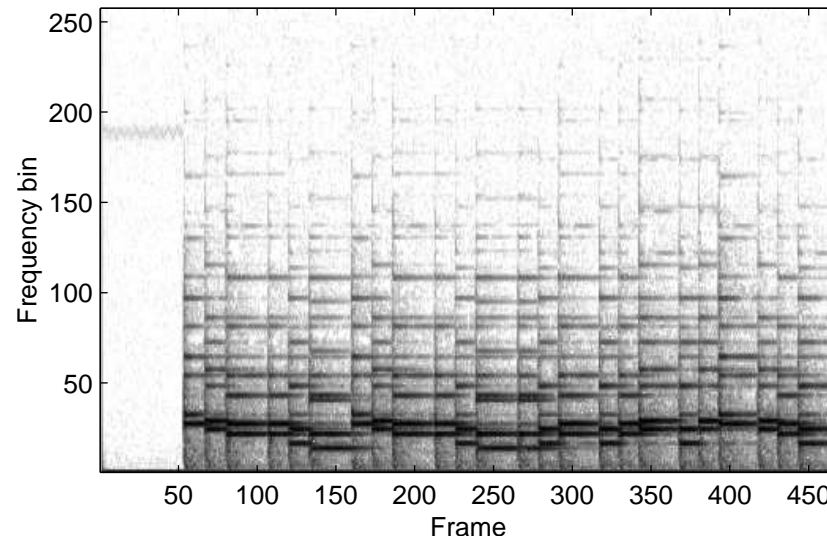
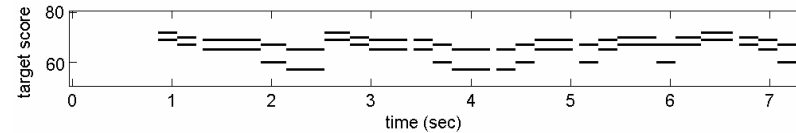


Images from (Cichocki, Kasprzak & Amari 1996).

# 'Music transcription' example

Liszt Etude No 5 (extract)

- PCA to reduce 256 dimensions to 10.
- Geodesic flow: 350 iterations to converge.



Nonnegativity and independence constraints produce approximation to underlying notes.

# Fourier Expansion Updates

Replacing the Taylor expansion in Newton's method

# Recap: Non-negative ICA

Task of non-negative ICA:

- To estimate sources  $\mathbf{s} = (s_1, \dots, s_n)$  and mixing matrix  $\mathbf{A}$  in  $\mathbf{x} = \mathbf{A}\mathbf{s}$  or  $\mathbf{X} = \mathbf{A}\mathbf{S}$
- given observations  $\mathbf{x} = (x_1, \dots, x_n)$
- where the sources are
  1. *non-negative*, i.e.  $\Pr(s_i < 0) = 0$ , and
  2. *independent*, i.e.  $p(s_i s_j) = p(s_i)p(s_j)$  if  $i \neq j$ .

# Previous Work

- Previously we showed that if  $\Pr(s < \delta) > 0$  for any  $\delta > 0$  (*well-grounded*), need only find a rotation of pre-whitened observations which is non-negative.
- Geodesic search, on manifold of orthogonal matrices
- Tangent gradient for search direction

Here we explore use of second-order information, using Fourier expansion instead of Taylor expansion in normal Newton method.

# Non-negative ICA system

- Pre-whiten  $\mathbf{z} = \mathbf{V}\mathbf{x}$
- Calculate  $\mathbf{y} = \mathbf{W}\mathbf{z}$  where  $\mathbf{W}$  orthonormal, i.e.  
 $\mathbf{W}^T\mathbf{W} = \mathbf{W}\mathbf{W}^T = \mathbf{I}_n$
- Calculate rectified output  $\mathbf{y}_+ = (y_1^+, \dots, y_n^+)$  where  
 $y_i^+ = g_+(y_i) \equiv \max(y_i, 0)$
- At each update step, update  $\mathbf{W}_{new} \leftarrow \mathbf{R}\mathbf{W}_{old}$  for orthonormal “rotation” matrix  $\mathbf{R} \in SO(n)$ . ( $\mathbf{W}$  remains orthonormal.)

Can write  $\mathbf{Y}_{new} = \mathbf{R}\mathbf{W}_{old}\mathbf{Z}$ , or  $\mathbf{Y} = \mathbf{R}\mathbf{W}\mathbf{Z}$ .

# Rotation as an exponential

- Orthonormal rotation matrix  $\mathbf{R} \in SO(n)$  is exponential of a skew-symmetric matrix, i.e.  $\mathbf{R} = e^{\mathbf{B}}$  where  $\mathbf{B}^T = -\mathbf{B}$  is skew-symmetric
- non-zero elements  $\{\phi_{ij} \mid i < j\}$  of  $\Phi = \text{UT}_+(\mathbf{B})$  are coords of an  $(n(n-1)/2)$ -dimensional parameter space.

E.g. for  $n = 2$ , we have

$$\mathbf{B} = \begin{pmatrix} 0 & \phi \\ -\phi & 0 \end{pmatrix} \quad \text{giving} \quad \mathbf{R} = \begin{pmatrix} \cos \phi & \sin \phi \\ -\sin \phi & \cos \phi \end{pmatrix}.$$

# Constrained optimization

- For NNICA, minimize  $J = \frac{1}{2} \|\mathbf{Y}_-\|_F^2 = \frac{1}{2} \sqrt{\sum_{ij} y_{ij}^-}$
- This has derivative in  $\Phi$ -space  
 $\nabla_{\Phi} J = \mathbf{U} \mathbf{T}_+(\mathbf{Y}_- \mathbf{Y}^T - \mathbf{Y} \mathbf{Y}_-^T)$
- Defining  $\Phi$ -space inner product as  $\langle \Phi, \Theta \rangle = \sum_{ij} \phi_{ij} \theta_{ij}$   
with norm  $|\Phi| = \sqrt{\langle \Phi, \Phi \rangle} = \|\Phi\|_F$

Then:

- $-\nabla_{\Phi} J$  is the *steepest descent* gradient for  $J$  in  $\Phi$ -space
- $\theta = |\nabla_{\Phi} J| = \frac{1}{2} \|\mathbf{Y}_- \mathbf{Y}^T - \mathbf{Y} \mathbf{Y}_-^T\|_F$  is the gradient norm,  
and
- $\mathbf{H}_{\Phi} = -\nabla_{\Phi} J / \theta = -\mathbf{U} \mathbf{T}_+(\mathbf{Y}_- \mathbf{Y}^T - \mathbf{Y} \mathbf{Y}_-^T) / \theta$  is the  
unit-norm steepest descent direction

# Steepest descent “line” (geodesic) search

- Start from  $\Phi(0) = \mathbf{0}$ , i.e.  $\mathbf{R}(0) = \mathbf{I}_n$
- $\Phi(t) = t\mathbf{H}_\Phi$  gives a geodesic  $\mathbf{R}(t) = e^{\mathbf{B}}(t)$ , where  
 $\mathbf{B}(t) = \Phi(t) - \Phi(t)^T$   
(Nishimori, 1999)
- Defining  $\mathbf{H} = \mathbf{H}_\Phi - \mathbf{H}_\Phi^T = -(\mathbf{Y}_- \mathbf{Y}^T - \mathbf{Y} \mathbf{Y}_-^T) / \theta$  as  
B-space direction, we get  $\mathbf{B}(t) = t\mathbf{H}$

So reduce  $J$  by performing “line search” to minimize  $J(t)$  in this direction

# Rotational Geometry of $\mathbf{R}$ for $n \leq 3$

- For  $n = 2$  we get Givens rotations (Comon, 1994)

$$\mathbf{H} = \pm \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \quad \mathbf{B}(t) = \pm \begin{pmatrix} 0 & t \\ -t & 0 \end{pmatrix}$$

$$\mathbf{R} = \begin{pmatrix} \cos t & \pm \sin t \\ \mp \sin t & \cos t \end{pmatrix}$$

- For  $n = 3$  get normalized Rodrigues formula (Fiori & Rossi 2003)

$$\mathbf{R}(t) = \mathbf{I}_n + \sin t \mathbf{H} + (1 - \cos(t)) \mathbf{H}^2$$

- For both of these, we see  $\mathbf{R}(t) = \mathbf{R}(t + 2k\pi)$ .

# Taylor expansion of $J$

If close to min of  $J(t)$  on a line, use Taylor expansion about minimum  $t^*$ :

- $J(t) \approx a_0 + a_1(t - t^*) + a_2(t - t^*)^2$
- thus  $J'(t) \approx a_1 + 2a_2(t - t^*)$  and  $J''(t) \approx 2a_2$ .
- Since  $J'(t) = 0$  at  $t = t^*$ , we must have  $a_1 = 0$ ,
- So estimate of distance from  $t^*$  of  $(t - t^*) \approx J'(t)/J''(t)$ .
- So guess min is at  $\hat{t} = t - J'(t)/J''(t)$ ,

I.e. this is a Newton update step.

# Fourier expansion

$J(t)$  repeats every  $t = 2k\pi$ : suggests *Fourier expansion*:

$$\begin{aligned} J(t) &= -a_0/2 - \sum_i (a_i \cos(t-t^*) + b_i \sin(t-t^*)) \\ &\approx -a_0/2 - a_1 \cos(t - t^*) \end{aligned}$$

(Ignores higher order terms, and term in  $\sin(t - t^*)$  is zero since derivative must be zero at  $t = t^*$ ).

- Differentiating we get  $J'(t) \approx a_1 \sin(t - t^*)$  and  $J''(t) \approx a_1 \cos(t - t^*)$ ,
- Leads to estimate for the minimum of  $\hat{t} = t - \arctan(J'(t), J''(t))$  where  $\arctan(\cdot, \cdot)$  is a four-quadrant arc tan function

Notice  $\arctan(J'(t), J''(t)) \approx J'(t)/J''(t)$  for small  $t - t^*$ , i.e. Newton as  $t \rightarrow t^*$ .

# Line derivatives etc.

For the Non-negative ICA system, we get:

$$\mathbf{H} = -(\mathbf{Y}_- \mathbf{Y}^T - \mathbf{Y} \mathbf{Y}_-^T) / \theta$$

$$J'(t) = -2\theta = -\|\mathbf{Y}_- \mathbf{Y}^T - \mathbf{Y} \mathbf{Y}_-^T\|_F.$$

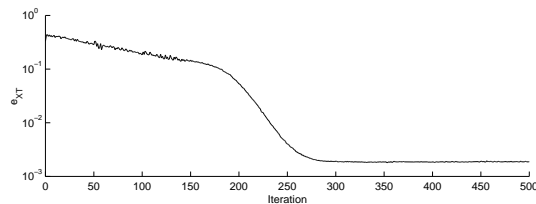
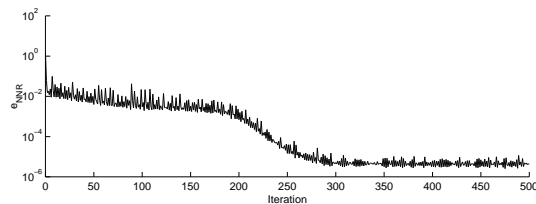
$$\begin{aligned} J''(t) &= \langle \mathbf{K}_- \circ \mathbf{Y}'(t), \mathbf{K}_- \circ (\mathbf{H} \mathbf{Y}) \rangle + \langle \mathbf{Y}_-, \mathbf{H} \mathbf{Y}'(t) \rangle \\ &= \|\mathbf{K}_- \circ (\mathbf{H} \mathbf{Y})\|_F^2 + \langle \mathbf{Y}_-, \mathbf{H}^2 \mathbf{Y} \rangle. \end{aligned}$$

where  $\mathbf{K}_- = [k_{ij}^-]$  be an indicator matrix for  $\mathbf{Y}_-$ , and  $\circ$  represents element-wise multiplication.

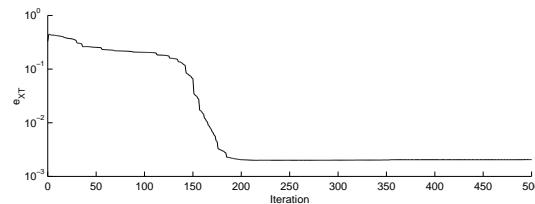
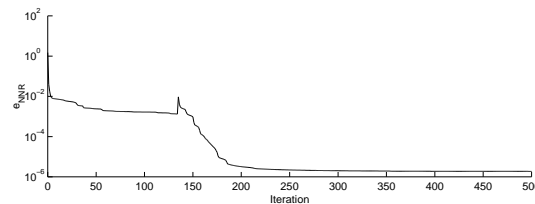
# Summary of algorithm

1. Initialize  $\mathbf{W} = \mathbf{I}_n$
2. Calculate  $\mathbf{Y} = \mathbf{W}\mathbf{Z}$ ,  $\mathbf{Y}_- = g_-(\mathbf{Y})$  and  $\theta$  as in (??).
3. If  $\theta = 0$ , finish.
4. Calculate  $\mathbf{H}$ ,  $J'(t)$  and  $J''(t)$  as required, and set  $t_1 = -\arctan(J'(t), J''(t))$ .
5. Calculate  $\mathbf{B} = t_1\mathbf{H}$  and  $\mathbf{R} = e^{\mathbf{B}}$  (using e.g. the Rodrigues formula for  $n \leq 3$ ).
6. Update  $\mathbf{W} \leftarrow \mathbf{R}\mathbf{W}$ .
7. Repeat from step 2 until  $\theta = 0$ .

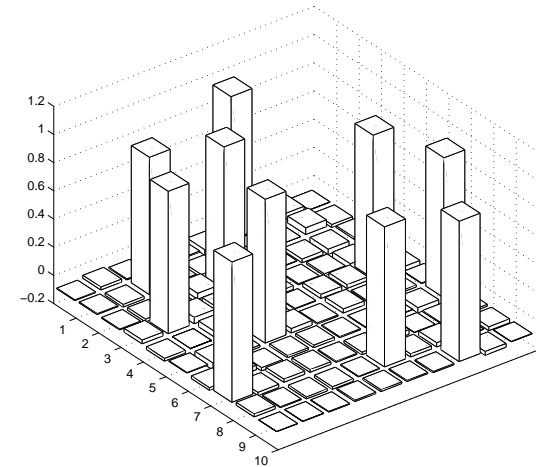
# Example: Artificial data, $n = 10$



(a)



(b)



(c)

(a) Fastest single-step “line” search ( $J$  and permutation error)

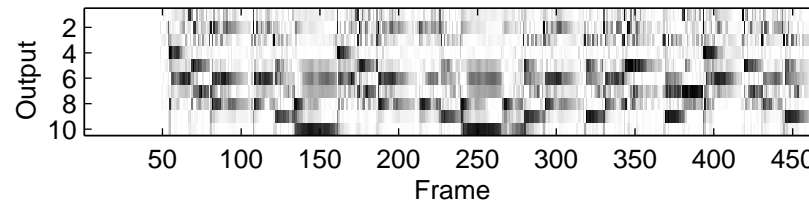
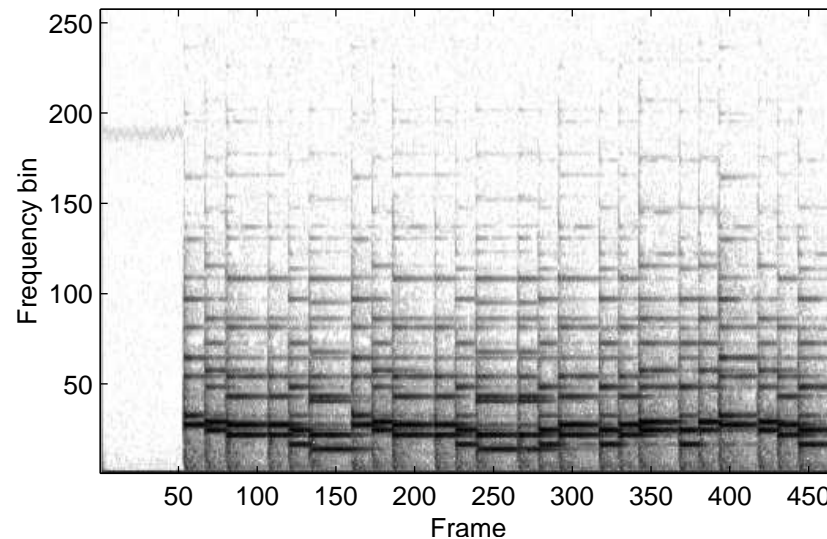
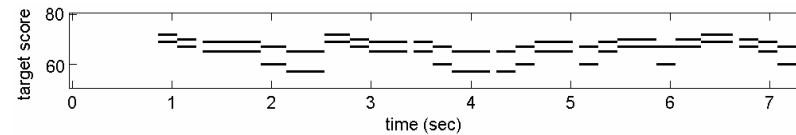
(b) Fourier expansion step (c) Resulting permutation matrix

# Music Transcription example

Previous result: Geodesic flow

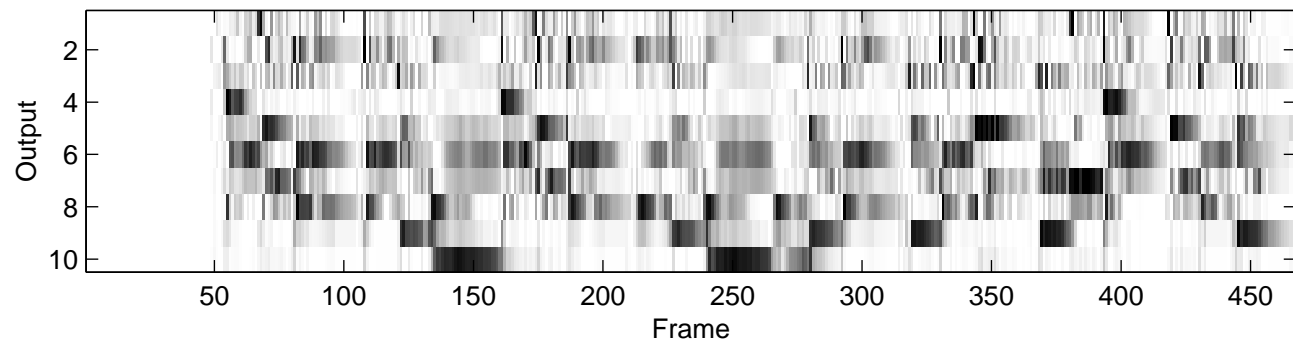
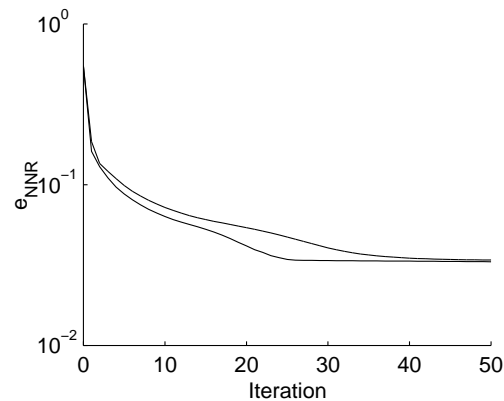
Liszt Etude No 5 (extract)

- PCA to reduce 256 dimensions to 10.
- Geodesic flow: 350 iterations to converge.



# Music example: Fourier expansion step

Fourier expansion step speeds up a bit  
(c.f. 350 iterations for Geodesic Flow)



Also: No adjustable parameters required.

# Other possible directions

- Conjugate Direction updates
- Visualizing and toral subgroups

# Conjugate Gradients in $SO(n)$

Standard conjugate gradients algorithm

1. Set  $k = 0$  and choose an initial search direction  $\mathbf{h}_0 = -\mathbf{g}_0$  where  $\mathbf{g}_0 = \nabla J$
2. Choose  $t_k = t_k^*$  to min  $J(\mathbf{x})$  along  $\mathbf{x} = \mathbf{x}_k + t_k \mathbf{h}_k$
3. At the minimum, find the new gradient  $\mathbf{g}_{k+1} = \nabla_{\mathbf{x}} J$  at  $t_k^*$
4. Choose a new search direction  $\mathbf{h}_{k+1} = -\mathbf{g}_{k+1} + \gamma_k \mathbf{h}_k$  where

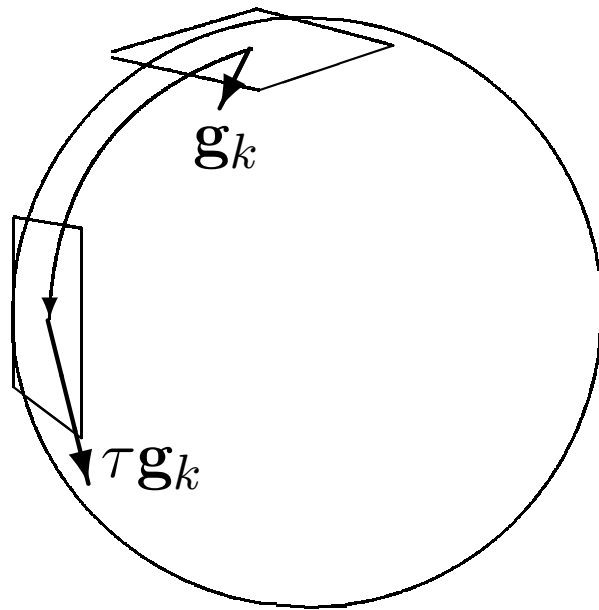
$$\gamma_k^{\text{FR}} = \frac{\mathbf{g}_{k+1}^T \mathbf{g}_{k+1}}{\mathbf{g}_k^T \mathbf{g}_k} \quad \text{or} \quad \gamma_k^{\text{PR}} = \frac{\mathbf{g}_{k+1}^T (\mathbf{g}_{k+1} - \mathbf{g}_k)}{\mathbf{g}_k^T \mathbf{g}_k}$$

5. Repeat from 2 with  $k = k + 1$  until  $\mathbf{g}_k$  small

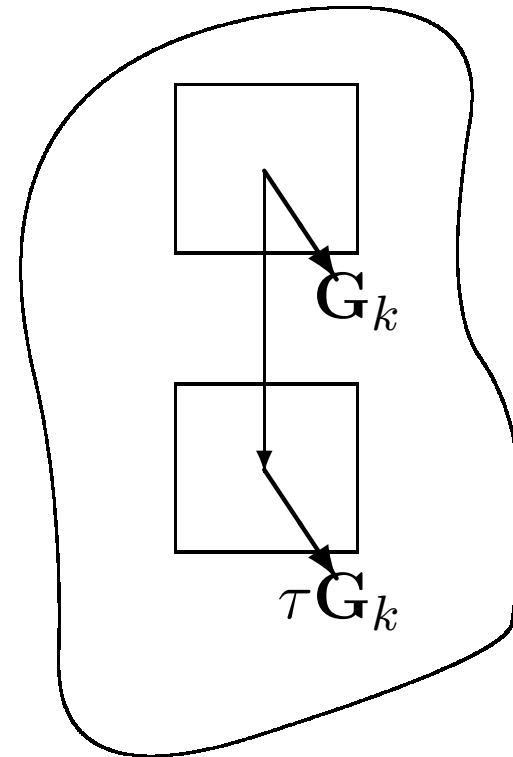
But:  $\mathbf{g}_k$ s live in tangent spaces – these change!

# Parallel Transport

Need to *transport* gradients (Edelman, Arias, Smith 1998).



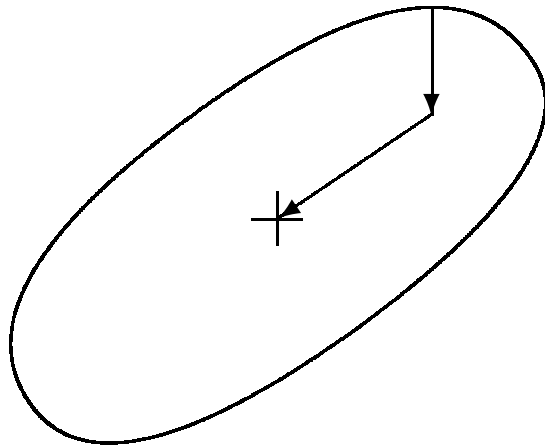
Parallel transport  
in Lie Group



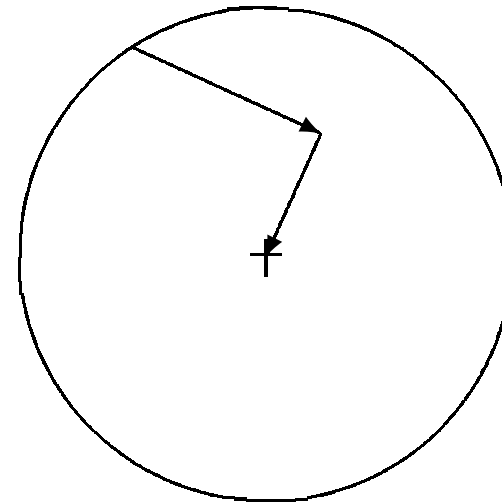
Parallel transport  
in Lie Algebra

# Direct Conjugate Direction Updates

Standard CG is approximation to underlying principle:



Conjugate Gradients  
(original space)



Orthogonal gradients  
(dual space)

- CG corresp to Orthogonal gradients in dual space (with identity Hessian).

# Conjugate Updates

- Want new search direction  $\mathbf{h}_{k+1}$  to be conjugate to previous, i.e.  $\mathbf{h}_{k+1}^T (\nabla^2 J) \mathbf{h}_k = 0$
- Notice that  $d/dt_k (\nabla J) = (\nabla^2 J) \mathbf{h}_k$  is a vector only
- For direction we have “just come along”:

$$\mathbf{V}_k = \frac{d}{dt} \nabla_{\mathbf{B}} J = 2 \text{skew} \left( \left( \frac{d}{dt} \nabla_{\mathbf{W}} J \right) \mathbf{W}^T \right)$$

- Using e.g.  $\mathbf{H}_{k+1} = \tilde{\mathbf{H}}_{k+1} / |\tilde{\mathbf{H}}_{k+1}|$  where  
 $\tilde{\mathbf{H}}_{k+1} = \mathbf{G}_{k+1} - \tilde{\mathbf{G}}_{k+1}$  and  
 $\tilde{\mathbf{G}}_{k+1} = \mathbf{V}_k \langle \mathbf{G}_{k+1}, \mathbf{V}_k \rangle / \langle \mathbf{V}_k, \mathbf{V}_k \rangle$  ensures  $\langle \mathbf{H}_{k+1}, \mathbf{V}_k \rangle = 0$ ,  
i.e. conjugate.
- Avoids calculating complete Hessian

# Visualizing $SO(n)$

- $SO(2)$  is like a circle (1-sphere), but
- $SO(n)$  is *not* like a hypersphere ( $m$ -sphere).
- For  $n = 2$  and  $n = 3$ ,  $SO(n)$  equiv to rotations about axis.
- What about  $SO(n)$  for  $n \geq 4$ ?

Jordan canonical form for skew-symmetric matrices:

$$\mathbf{B} = \mathbf{U} \begin{pmatrix} \Theta_1 & & & \\ & \ddots & & \\ & & \ddots & \\ & & & \Theta_m \end{pmatrix} \mathbf{U}^T$$

where  $\mathbf{U} \in SO(n)$  is orthogonal and  $\Theta_i = \begin{pmatrix} 0 & \theta_i \\ -\theta_i & 0 \end{pmatrix}$  (or 0).

# Exponential form

Jordan canonical form for rotation matrices:

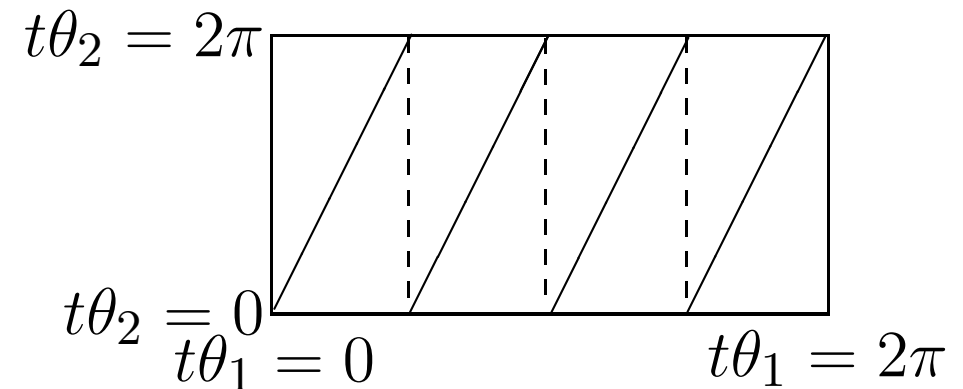
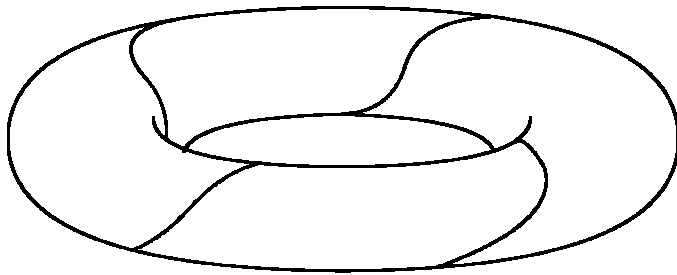
$$\mathbf{R} = \exp \mathbf{B} = \mathbf{U} \begin{pmatrix} \mathbf{M}_1 & & \\ & \ddots & \\ & & \mathbf{M}_m \end{pmatrix} \mathbf{U}^T$$

where  $\mathbf{M}_i = \begin{pmatrix} \cos \theta_i & \sin \theta_i \\ -\sin \theta_i & \cos \theta_i \end{pmatrix}$  (or 1).

- The elementary rotations  $\mathbf{M}_i$  are independent from each other: i.e. commutative.
- Once we choose  $\mathbf{U}$ , can make several moves without a new decomposition.

# Toral subgroup

- Choosing decomposition  $\mathbf{U}$  gives an  $m$ -dimensional subgroup.
- This is *toral*, since each is a separate rotation.
- A given line search may define a search direction on this torus.



# Possible uses...

Possible uses for toral subgroup:

- Simplify exponential: calculate  $U$  once for many updates
- Conjugate tori (instead of conjugate gradients)
- Component-aligned tori (like line search along axes): easy to calculate exponential

# Conclusions

- Problem: Automatic Music Transcription
- Approach: Nonnegative Independent Component Analysis (NNICA)
- Optimization with orthonormal constraint is common in ICA
- Keeping to the constraint is difficult
- Optimization on manifolds offer a possible approach
- Can use steepest descent, line search or conjugate gradient
- Automatic Music Transcription Example
- More possibilities for the future...