Group theoretical methods in Machine Learning

Risi Kondor
Columbia University

Tutorial at ICML 2007
Part 2

The symmetric group
$S_n$ is the group of bijections

$$\sigma: \{1, 2, \ldots, n\} \rightarrow \{1, 2, \ldots, n\}$$

under composition of maps.

Clearly, $|S_n| = n!$. 
\[ \sigma(1) = 3 \]
\[ \sigma(2) = 2 \]
\[ \sigma(3) = 6 \]
\[ \sigma(4) = 5 \]
\[ \sigma(5) = 4 \]
\[ \sigma(6) = 1 \]

\[
\sigma = \begin{pmatrix}
1 & 2 & 3 & 4 & 5 & 6 \\
3 & 2 & 6 & 5 & 4 & 1 \\
\end{pmatrix}
\]
Cycle notation

Example:

\[ \sigma = (163)(45)(2) \]

Cycle type:

\[ \sigma = (3, 2, 1) \]
Generators

Transpositions \((i, j)\) generate the whole group.

In fact, adjacent transpositions are sufficient, since (assuming \(i < j\))

\[(i, j) = (i, i+1) \ldots (j-2, -1)(j-1, j) \ldots (i+1, i+2)(i, i+1)\]
Subgroups

Cayley’s theorem:

Any finite group $G$ is a subgroup of $S_{|G|}$. 
Subgroups

\( S_k < S_n \) permutes \( \{1, 2, \ldots, k\} \)

\( S_\lambda = S_{\lambda_1} \times S_{\lambda_2} \times \ldots \times S_{\lambda_k} < S_n \) permutes \( \{1, 2, \ldots, \lambda_1\}, \{\lambda_1 + 1, \ldots, \lambda_1 + \lambda_2\}, \ldots, \{n - \lambda_k, \ldots, n\} \)
Normal subgroups

\[ A_n = \{ \sigma \in S_n \mid \text{sgn} (\sigma) = 1 \} \]

For \( n \geq 5 \) this is the only one!
Representations
1. The **trivial** representation $\rho_{\text{triv}}(\sigma) = (1)$

2. The **alternating** representation $\rho_{\text{triv}}(\sigma) = \text{sgn}(\sigma)$

3. The **defining** representation $[\rho_{\text{def}}(\sigma)]_{i,j} = \delta_{\sigma(i),j}$

reducible!!!
Young diagrams

\[ \lambda \vdash n \leftrightarrow \text{integer partitions} \]

\[ \leftrightarrow \longrightarrow \text{irreducible representations} \quad \rho \in \mathcal{R} \]
Young Tableaux

A standard tableau if numbers increase left to right and top to bottom.

Standard tableaux of shape \( \lambda \) ↔ dimensions of \( \rho \lambda \)
Young's Orthogonal Representation (YOR)

\[
\begin{align*}
[\rho \lambda (\tau_k)]_{t,t} &= \frac{1}{d_t(k, k+1)} \\
[\rho \lambda (\tau_k)]_{\tau_k(t),t} &= \sqrt{1 - \frac{1}{d_t(k, k+1)^2}} \\
\text{if } \tau_k(t) \text{ standard}
\end{align*}
\]

All \( \rho \lambda (\sigma) \) are real!
$S_4$

\[ d = 1 \quad \rho_{(4)}(\sigma) = (1) \]

\[ d = 3 \]

\[ d = 2 \]

\[ d = 3 \]

\[ d = 1 \quad \rho_{(1,1,1,1)}(\sigma) = (\text{sgn}(\sigma)) \]
\[ S_n \]

\[
d = 1
\]

\[
d = n - 1
\]

\[
d = n(n - 3)/2
\]

\[
d = (n - 1)(n - 2)/2
\]

\[
d = n(n - 1)(n - 5)/6
\]
Hook rule

\[ d_{\lambda} = \frac{n!}{\prod_{i=1}^{n} l_i} \]

\[ l_i = \text{length of hook } i \]
Restricted representations

If $\rho$ is a representation of $S_n$, $\rho_{\downarrow S_k}$ is a representation of $S_k$ given by

$$\rho_{\downarrow S_k}(\sigma) = \rho(\sigma) \quad \sigma \in S_k.$$
Young’s Rule

In YOR

\[ \rho \lambda \downarrow_{S_{n-1}} (\sigma) = \bigoplus \rho \lambda^- (\sigma) \]

\[ \lambda^- \vdash n - 1 \]
\[ \lambda^- < \lambda \]
Bratelli diagram
Clausen’s FFT for $S_n$
Define the **contiguous cycle**

\[ [i, j] = (i, i + 1, i + 2, \ldots, j) \]

The cosets \([1, n]S_{n-1}, [2, n]S_{n-1}, \ldots, [n, n]S_{n-1}\)
form a partition of \(S_n\)

Idea: decompose FT over \(S_n\) into \(n\) smaller FTs over the \([i, n]S_{n-1}\) cosets.
\[
\hat{f}(\lambda) = \sum_{\sigma \in S_n} \rho_\lambda(\sigma) f(\sigma)
\]

\[
= \sum_{i=1}^{n} \sum_{\sigma' \in S_{n-1}} \rho_\lambda([i, n] \sigma') f([i, n] \sigma')
\]

\[
= \sum_{i=1}^{n} \rho_\lambda([i, n]) \sum_{\sigma' \in S_{n-1}} \rho_\lambda(\sigma') f_{[i,n]S_{n-1}}(\sigma')
\]

\[
= \sum_{i=1}^{n} \rho_\lambda([i, n]) \bigoplus_{\lambda^- \vdash n-1, \lambda^- < \lambda} \hat{f}_{[i,n]S_{n-1}}(\lambda^-)
\]
Op count

- $[i, k]$ is a product of $k - i$ adjacent transpositions.
- Computing $\rho_\lambda([i, j]) \cdot M$ takes $2 \lambda^2$ time.
- For $\lambda \vdash k$, $\lambda^2 = k!$.
- Layer $k$ has $n!/k!$ Fourier transforms.

\[ \sum_{k=1}^{n} \sum_{i=1}^{k} 2(k - i) n! = n! \frac{(n + 1) n (n - 1)}{3} \]
Snob
$S_{n}\text{ob}$

A C++ library for fast Fourier transforms on the symmetric group.

author: Risi Kondor, Columbia University (risi@cs.columbia.edu)

Development version as of August 23, 2006 (unstable!)

Documentation: [ps][pdf]
C++ source code: [directory]
BibTeX entry: [bib]
Entire package: [tar.gz]

ALL SOFTWARE ON THIS PAGE IS DISTRIBUTED UNDER THE TERMS OF THE GNU GENERAL PUBLIC LICENSE [site]

References:

3. K. L. Kwek, T. Olson, D. Rockmore and K. S. Tan: Nonlinear approximation theory on finite...
#include <vector>
#include "base.h"
#include "Matrix.hpp"
#include "<iostream>
#include "Sn.hpp"
#include "SnFunction.hpp"
#include "StandardTableau.hpp"

using namespace std;

class Sn::FourierTransform: FiniteGroup::FourierTransform{

public:

friend class Sn::Function;
friend class Sn::Ftree;

FourierTransform(const Sn& _group);
FourierTransform(const Sn& _group, int dummy):group(_group),n(_group.n){};
FourierTransform(const Sn& _group, const vector<Matrix<FIELD>>* matrices);
FourierTransform(const Function& f);
~FourierTransform();

Function* iFFT() const;

FIELD operator()(const StandardTableau& t1, const StandardTableau& t2) const;

double norm2() const {double result; for(int i=0; i<matrix.size(); i++) result+=1; return result;}

string str() const;

vector<Matrix<FIELD>>* matrix;

private:

void fft(const Sn::Function& f, const int offset);
void ifft(Sn::Function* target, const int _offset) const;

const int n;
const Sn* group;

};
Sn::Irreducible

 Represents an irreducible representation $\rho_\lambda$ of $S_n$.

 Parent class: FiniteGroup::Irreducible

 CONSTRUCTORS

 Irreducible(Sn* G, Partition* lambda)
 Construct the irreducible representation of the symmetric group G corresponding to the partition lambda.

 MEMBER FUNCTIONS

 Matrix<FIELD> & rho(const Sn::Element* sigma)
 Returns $\rho(\sigma)$, the representation matrix of permutation sigma in Young's orthogonal representation.

 FIELD character(const Partition* nu)
 Returns $\chi(\mu)$, the character of this representation at permutations of cycle type $\mu$.

 void computeTableaux()
 Compute the standard tableaux of this irreducible if they have not already been computed. Because this is an expensive operation, it is postponed until some function is called (such as rho or character) which requires the tableaux of this particular irreducible. computeTableaux() is called automatically by these functions, and once the tableaux have been computed they are stored for the lifetime of the Irreducible.

 StandardTableau* tableau(const int t)
 Return a new standard tableau of index t. This works even if tableauW has not been computed.

 void computeYUR()
 Compute and store the coefficients (2.5) and (2.6) in Young's orthogonal representation for all adjacent transpositions $\tau_\lambda$ and all tableau $t$ of shape $\lambda$. Because this is an expensive operation, these coefficients are not normally computed until they are demanded by functions such as rho or character. computeYUR() is called automatically by these functions, and once the tableaux have been computed they are stored for the lifetime of the Irreducible. computeYUR() also requires the tableaux, so it calls computeTableaux() if those have not been computed yet.

 void applyCycle(const int j, Matrix<FIELD>* M[ int m])
 void applyCycle(const int j, Matrix<FIELD>* M[ int m])
Sparse transforms
Twisted transforms

Restrict to two sided cosets $\sigma_L S_k \sigma_R$
The isotypal components of $S_n$
Persi Diaconis

“Group Representations in Probability and Statistics”
IMS, 1988
Interpretation of $\sigma$: rank of $j$ is $i = \sigma^{-1}(j)$

Partial rankings: $\sigma(\mathbb{S}_{\lambda_1} \times \ldots \times \mathbb{S}_{\lambda_k})$ cosets

Permutation representations:

$$\left[ \rho^k(\sigma) \right]_{(j_1,\ldots,j_k),(i_1,\ldots,i_k)} = \begin{cases} 1 & \text{if } \sigma(i_a) = j_a, \ a = 1, 2, \ldots, k \\ 0 & \text{otherwise} \end{cases}$$
\[ T^{-1} \rho^k T = \bigoplus \rho \lambda \]

\( \lambda \vdash n - k \)  \( \lambda \geq (n - k) \)
Application: multi-object tracking

R. Kondor, A. Howard, T. Jebara (AISTATS 2007)
Band limited approximation to $p(\sigma)$

<table>
<thead>
<tr>
<th>$n$</th>
<th>$n!$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>24</td>
</tr>
<tr>
<td>5</td>
<td>120</td>
</tr>
<tr>
<td>6</td>
<td>720</td>
</tr>
<tr>
<td>7</td>
<td>5040</td>
</tr>
<tr>
<td>8</td>
<td>40320</td>
</tr>
<tr>
<td>9</td>
<td>362880</td>
</tr>
<tr>
<td>10</td>
<td>3628800</td>
</tr>
<tr>
<td>11</td>
<td>39916800</td>
</tr>
<tr>
<td>12</td>
<td>$4.8 \cdot 10^8$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>15</td>
<td>$1.3 \cdot 10^{12}$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>20</td>
<td>$2.4 \cdot 10^{18}$</td>
</tr>
</tbody>
</table>

$\Delta_{\sigma_1, \sigma_2} = \begin{cases} 1 & \text{if } \sigma_1 = (i, j) \cdot \sigma_2 \\ -n(n-1)/2 & \text{if } \sigma_1 = \sigma_2, \\ 0 & \text{otherwise.} \end{cases}$

$\alpha_\lambda = -\binom{n}{2} \left( 1 - \frac{\text{tr} \left[ \rho_\lambda((1, 2)) \right]}{d_\lambda} \right)$
\[ d = 1 \]
\[ d = n - 1 \]
\[ d = n(n - 3)/2 \]
\[ d = (n - 1)(n - 2)/2 \]
\[ d = n(n - 1)(n - 5)/6 \]
\[ d = n(n - 2)(n - 4)/3 \]
\[ d = (n - 1)(n - 2)(n - 3)/6 \]
1. Noise: diffusion over transpositions

2. Observations: $O_{i \rightarrow j}$ with probability $\pi$

3. Inference: $p[\sigma(i) = j]$
1. Noise model

\[ p_{t'} = \left( I + \frac{\beta(t' - t)}{m} \Delta \right)^m p_t = e^{\beta(t' - t)\Delta} \quad m \to \infty \]

\[ \hat{p}_{t'}(\rho\lambda) = e^{\beta\alpha\lambda(t' - t)} \hat{p}_t(\rho\lambda) \]

**cost:** \( O(d_{\text{max}}^2) \)
2. Observations

\[ p(O_{i\to j} \mid \sigma) = \begin{cases} 
\pi & \text{if } \sigma(i) = j, \\
(1 - \pi)/(n - 1) & \text{if } \sigma(i) \neq j.
\end{cases} \]

Compute the FT of \[ f_{i\to j}(\sigma') = f([j, n] \sigma' [i, n]^{-1}) \]

\[ \hat{f}(\lambda) = \sum_{j=1}^{n} \rho([j, n]) \left( \bigoplus_{\lambda^{-}} \hat{f}_{i\to j}(\rho_{\lambda^{-}}) \right) \rho([i, n]^{-1}) \]

\text{cost: } O(d_{\text{max}}^2 n)
3. Inference

\[ p[\sigma(i) = j] = \hat{\rho}_{i \rightarrow j}((n - 1)) \]

cost: \( O(n^3) \)
Results

\( n = 6 \)  
\( n! = 720 \)

\( n = 10 \)  
\( n! = 6.6 \cdot 10^6 \)

\( n = 15 \)  
\( n! = 1.3 \cdot 10^{12} \)

using just the components  
\( \rho(4), \rho(n-1,1), \rho(n-2,2), \rho(n-2,1,1) \)

\( n = 15 \)  
\( t \approx 59\text{ms} \)

\( n = 30 \)  
\( t \approx 3\text{s} \)