Haarpy: a Python library for integration over the Haar measure

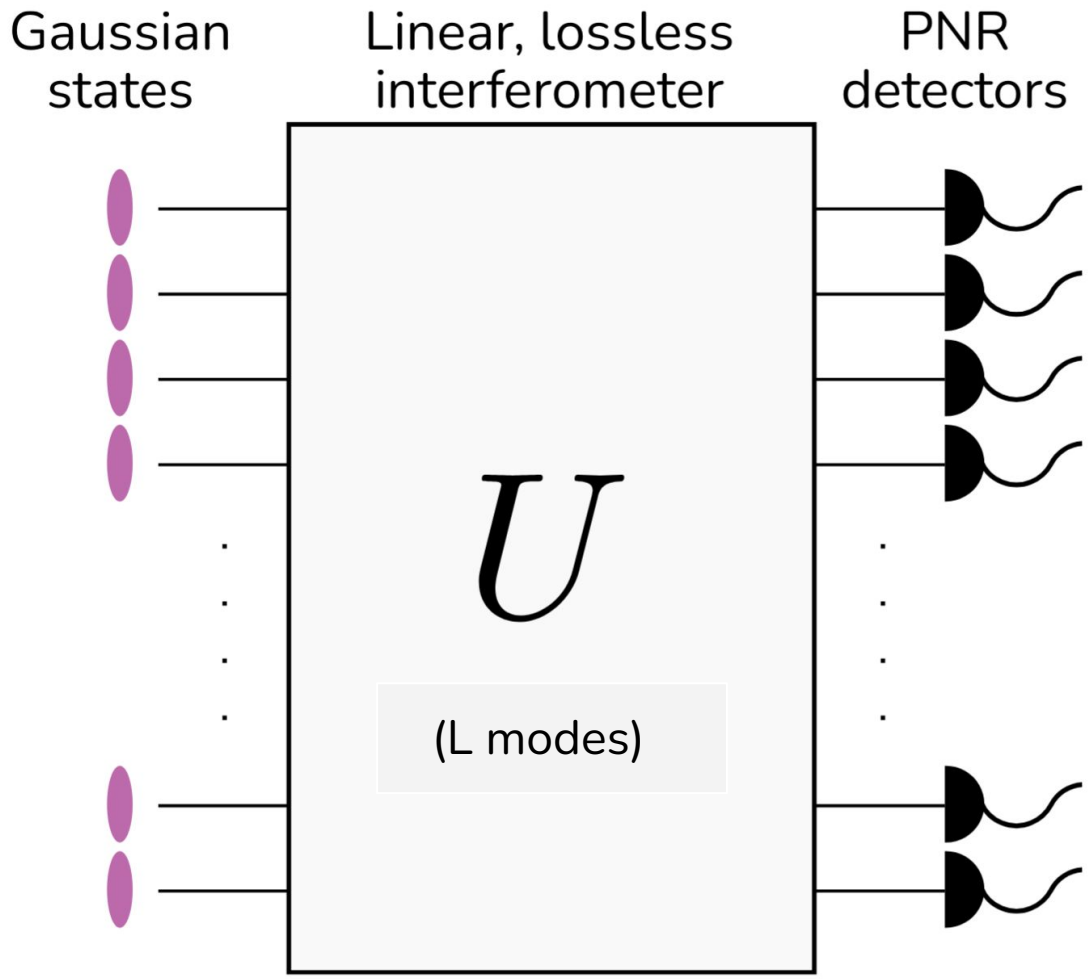
Yanic Cardin

Département de Génie Physique
Polytechnique Montréal

June 19th 2026

POLYTECHNIQUE MONTRÉAL

Gaussian boson sampling (GBS)



A set of Gaussian states is sent into a lossless interferometer, mathematically represented by a **Haar random unitary matrix**. After propagation, the output light is measured using **photon-number-resolving measurement**.

The result of such measurement is a sequence of photon count

$$\boldsymbol{n} = (n_1, n_2, \ldots n_L)\,, \quad n_j \in \mathbb{N}$$

Martínez-Cifuentes et al. PRX Quantum 5, 040312 (2024)

The Haar measure

Haar Measure - The Haar measure **μ** on the unitary group **U(N)** is the unique probability measure that is both left and right invariant over the group, i.e., for all integrable functions **f** and for all **V ∈ U(N)**, we have

$$\int_{U(N)} f(U)d\mu(U) = \int_{U(N)} f(VU)d\mu(U) = \int_{U(N)} f(UV)d\mu(U)$$

- The notion of Haar measure formalizes the fundamental concept of drawing unitary matrices uniformly at random.
- All locally compact topological group admit a Haar measure.

The Weingarten calculus

The Weingarten calculus answers the following question: given a compact group **G** equipped with its Haar measure, how can one explicitly evaluate moments of the matrix coefficients of a random element of **G**?

For instance

Unitary group : $\int_{U(N)} U_{i_1 j_1} \dots U_{i_k j_k} U^*_{i'_1 j'_1} \dots U^*_{i'_k j'_k} \, dU$

Orthogonal group : $\int_{O(N)} O_{i_1 j_1} \dots O_{i_{2k} j_{2k}} \, dO$

The Weingarten calculus

Timeline

- **1978** - Weingarten computes asymptotic expressions for integrals over the unitary group in the large-dimension limit $N \to \infty$.

The Weingarten calculus

Timeline

- **1978** - Weingarten computes asymptotic expressions for integrals over the unitary group in the large-dimension limit $N \to \infty$.
- **2003** - Collins et al. find exact formulas, for the unitary group, valid for arbitrary group dimension.

The Weingarten calculus

Timeline

- **1978** - Weingarten computes asymptotic expressions for integrals over the unitary group in the large-dimension limit $N \to \infty$.
- **2003** - Collins et al. find exact formulas, for the unitary group, valid for arbitrary group dimension.
- **Today** - Full collection of closed-form Weingarten-type formulas for integrals over many classical groups and random matrix ensembles equipped with the Haar measure.

Applications

- Quantum information theory
- Statistical physics
- Random matrix theory
- Free probability

The Weingarten function

Theorem Let $U = (U_{ij})$ be an $N \times N$ Haar random unitary matrix. For four sequences $\boldsymbol{i} = (i_1, \ldots, i_k)$, $\boldsymbol{j} = (j_1, \ldots, j_k)$, $\boldsymbol{i'} = (i'_1, \ldots, i'_k)$ and $\boldsymbol{j'} = (j'_1, \ldots, j'_k)$ of positive integers in $[N]$, we have

$$\int_{U(N)} U_{i_1 j_1} \ldots U_{i_k j_k} U^*_{i'_1 j'_1} \ldots U^*_{i'_k j'_k} \, dU = \sum_{\sigma,\tau \in S_k} \delta_\sigma(\boldsymbol{i}, \boldsymbol{i'}) \delta_\tau(\boldsymbol{j}, \boldsymbol{j'}) \mathrm{Wg}^U(\sigma^{-1}\tau;\, N)$$

with

$$\mathrm{Wg}^U(\sigma;\, N) = \frac{1}{k!^2} \sum_{\lambda \vdash k} \frac{\chi^\lambda(1)^2 \chi^\lambda(\sigma)}{s_{\lambda,N}(1)}, \qquad \delta_\sigma(\boldsymbol{i}, \boldsymbol{i'}) = \prod_{s=1}^{k} \delta_{i_{\sigma(s)}, i'_s}$$

where S_k is the symmetric group and Wg^U is the unitary Weingarten function.

- $\chi^\lambda(1)$ is the dimension of the irrep λ of S_k.
- $\chi^\lambda(\sigma)$ is the character of element σ in the irrep λ.
- $s_{\lambda,L}(1)$ is the dimension of irrep λ of $U(N)$.
- The sum is taken over all partitions λ of k.

haarpy

A Python library for Weingarten calculus and integration of classical compact groups and ensembles

- Open-source
- Easy to install
- Thoroughly tested
- Documented
- Symbolic

```
pip install haarpy
```

haarpy.readthedocs.io

Quickstart

- Haarpy requires a group dimension variable for symbolic integration

```
import haarpy as ap
from sympy import Symbol
dimension = Symbol('N')
```

✓ 0.0s

Quickstart

- Haarpy requires a group dimension variable for symbolic integration

- Orthogonal group

$$\int_{O(N)} O_{i_1 j_1} \dots O_{i_{2k} j_{2k}} \, dO$$

```
import haarpy as ap
from sympy import Symbol
dimension = Symbol('N')
```

✓ 0.0s

```
i, j = (1,1,2,2), (2,2,1,1)
ap.haar_integral_orthogonal((i, j), dimension)
```

✓ 0.0s

$$\frac{N+1}{N(N-1)(N+2)}$$

Quickstart

- Haarpy requires a group dimension variable for symbolic integration

```
import haarpy as ap
from sympy import Symbol
dimension = Symbol('N')
```

✓ 0.0s

- Orthogonal group

$$\int_{O(N)} O_{i_1 j_1} \dots O_{i_{2k} j_{2k}} \, dO$$

```
i, j = (1,1,2,2), (2,2,1,1)
ap.haar_integral_orthogonal((i, j), dimension)
```

✓ 0.0s

$$\frac{N+1}{N\,(N-1)\,(N+2)}$$

- Unitary group

$$\int_{U(N)} U_{i_1 j_1} \dots U_{i_k j_k} U^*_{i'_1 j'_1} \dots U^*_{i'_k j'_k} \, dU$$

```
i, j, ip, jp = (1,2), (1,2), (1,2), (2,1)
ap.haar_integral_unitary((i, j, ip, jp), dimension)
```

✓ 0.0s

$$-\frac{1}{N\,(N-1)\,(N+1)}$$

```
ap.haar_integral_unitary()
ap.haar_integral_orthogonal()
ap.haar_integral_symplectic()
```

Supported groups and ensembles

- Unitary group
- Orthogonal group
- Symplectic group

} **Classical compact groups**

```
ap.haar_integral_unitary()
ap.haar_integral_orthogonal()
ap.haar_integral_symplectic()
```

```
ap.haar_integral_circular_orthogonal()
ap.haar_integral_circular_symplectic()
```

Supported groups and ensembles

- Unitary group
- Orthogonal group
- Symplectic group

} **Classical compact groups**

- Circular orthogonal ensemble
- Circular symplectic ensemble

} **Circular ensembles**

Supported groups and ensembles

```
ap.haar_integral_unitary()
ap.haar_integral_orthogonal()
ap.haar_integral_symplectic()
```

- Unitary group
- Orthogonal group
- Symplectic group

} **Classical compact groups**

```
ap.haar_integral_circular_orthogonal()
ap.haar_integral_circular_symplectic()
```

- Circular orthogonal ensemble
- Circular symplectic ensemble

} **Circular ensembles**

```
ap.haar_integral_free_orthogonal()
ap.haar_integral_free_symmetric()
```

- Free orthogonal group
- Free symmetric group

} **Quantum groups**

Supported groups and ensembles

```
ap.haar_integral_unitary()
ap.haar_integral_orthogonal()
ap.haar_integral_symplectic()
```

- Unitary group
- Orthogonal group
- Symplectic group

} **Classical compact groups**

```
ap.haar_integral_circular_orthogonal()
ap.haar_integral_circular_symplectic()
```

- Circular orthogonal ensemble
- Circular symplectic ensemble

} **Circular ensembles**

```
ap.haar_integral_free_orthogonal()
ap.haar_integral_free_symmetric()
```

- Free orthogonal group
- Free symmetric group

} **Quantum groups**

```
ap.haar_integral_permutation()
ap.haar_integral_centered_permutation()
```

- Random permutation matrices
- Random centered permutation matrices

} **Symmetric group**

Some auxiliary functions

- Returns the Murnaghan-Nakayama rule for the characters of the irreducible representations of the symmetric group
- Generates the set of non-crossing partitions of a given size
- Generates the permutations of the hyperoctahedral group
- Generates the permutations of the Young subgroup associated with a given integer partition

```
ap.murn_naka_rule()
```

```
ap.non_crossing_partitions()
```

```
ap.HyperoctahedralGroup()
```

```
ap.YoungSubgroup()
```

haarpy

A Python library for Weingarten calculus and integration of classical compact groups and ensembles

Thank you!