Computing Eigenvalues of Random Matrices

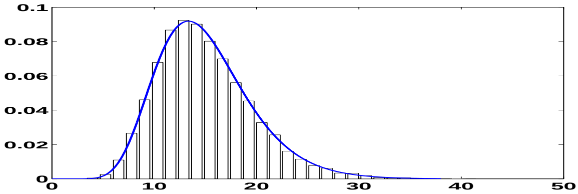
Plamen Koev
Department of Mathematics
North Carolina State University

Joint work with:
Cy Chan, Vesselin Drensky, Alan Edelman, Iain Johnstone, Raymond Kan

Foundations of Computational Mathematics, Hong Kong, June 26, 2008

Goal

- Efficiently computing the distributions of
  - eigenvalues and functions thereof
  - of Wishart, Jacobi, Laguerre
  - any Σ
  - real, complex, any β?
  - efficiently
- Example: λ_{max} of 4×4 Wishart with 7 DOF



- Exact vs Empirical with 20,000 replications

(Central) Wishart Matrix: Real Case

- Definition:

$$A = X^T X \in \mathbb{R}^{m \times m}, \quad \text{where } X \sim \mathcal{N}(0, I_n \otimes \Sigma) \in \mathbb{R}^{n \times m}$$

(Central) Wishart Matrix: Real Case

- Definition:

$$A = X^T X \in \mathbb{R}^{m \times m}, \quad \text{where } X \sim \mathcal{N}(0, I_n \otimes \Sigma) \in \mathbb{R}^{n \times m}$$

- Joint element density*:

$$|\Sigma|^{-\frac{n}{2}} |A|^{\frac{n-m+1}{2}-1} e^{-\mathrm{tr}(\frac{1}{2}\Sigma^{-1}A)}$$

(Central) Wishart Matrix: Real Case

- Definition:

$$A = X^T X \in \mathbb{R}^{m\times m}, \quad \text{where } X \sim \mathcal{N}(0, I_n \otimes \Sigma) \in \mathbb{R}^{n\times m}$$

- Joint element density*:

$$|\Sigma|^{-\frac{n}{2}} |A|^{\frac{n-m+1}{2}-1} e^{-\mathrm{tr}(\frac{1}{2}\Sigma^{-1}A)}$$

- Joint eigenvalue density:

$$(\det \Sigma)^{-\frac{n}{2}} \prod_{i=1}^{m} \lambda_i^{\frac{n-m+1}{2}-1} \prod_{j<k} |\lambda_k - \lambda_j|^1 \cdot {}_0F_0^{(1)}(-\tfrac{1}{2}\Lambda, \Sigma^{-1}),$$

where

$${}_0F_0^{(1)}(X, Y) = \sum_{k=0}^{\infty} \sum_{\kappa \vdash k} \frac{1}{k!} \cdot \frac{C_\kappa^{(1)}(X) C_\kappa^{(1)}(Y)}{C_\kappa^{(1)}(I)}$$

(Central) Wishart Matrix: Complex Case

- Definition:

$$A = X^* X \in \mathbb{C}^{m \times m}, \quad \text{where } X \sim \mathcal{CN}(0, I_n \otimes \Sigma) \in \mathbb{C}^{n \times m}$$

- Joint element density*:

$$|\Sigma|^{-\frac{n}{1}} |A|^{\frac{n-m+1}{1}-1} e^{-\mathrm{tr}(\frac{1}{1}\Sigma^{-1}A)}$$

- Joint eigenvalue density:

$$(\det \Sigma)^{-\frac{n}{1}} \prod_{i=1}^{m} \lambda_i^{\frac{n-m+1}{1}-1} \prod_{j<k} |\lambda_k - \lambda_j|^2 \cdot {}_0F_0^{(2)}(-\tfrac{1}{1}\Lambda, \Sigma^{-1}),$$

where

$${}_0F_0^{(2)}(X, Y) = \sum_{k=0}^{\infty} \sum_{\kappa \vdash k} \frac{1}{k!} \cdot \frac{C_\kappa^{(2)}(X) C_\kappa^{(2)}(Y)}{C_\kappa^{(2)}(I)}$$

Wishart Matrix: Real/Complex/Quaternion ($\beta = 1, 2, 4$)

- Definition:

$$A = X^D X, \quad \text{where } X \sim \mathcal{N}^{(\beta)}(0, I_n \otimes \Sigma)$$

- Joint element density*:

$$|\Sigma|^{-\frac{n\beta}{2}} |A|^{\frac{(n-m+1)\beta}{2}-1} e^{-\mathrm{tr}(\frac{\beta}{2}\Sigma^{-1}A)}$$

- Joint eigenvalue density:

$$(\det \Sigma)^{-\frac{n\beta}{2}} \prod_{i=1}^{m} \lambda_i^{\frac{(n-m+1)\beta}{2}-1} \prod_{j<k} |\lambda_k - \lambda_j|^\beta \cdot {}_0F_0^{(\beta)}(-\tfrac{\beta}{2}\Lambda, \Sigma^{-1}),$$

where

$${}_0F_0^{(\beta)}(X, Y) = \sum_{k=0}^{\infty} \sum_{\kappa \vdash k} \frac{1}{k!} \cdot \frac{C_\kappa^{(\beta)}(X) C_\kappa^{(\beta)}(Y)}{C_\kappa^{(\beta)}(I)}$$

Wishart Matrix: Real/Complex/Quaternion ($\beta = 1, 2, 4$)

- Definition:

$$A = X^D X, \quad \text{where } X \sim \mathcal{N}(0, I_n \otimes \Sigma)$$

- Joint element density*:

$$|\Sigma|^{-\frac{n\beta}{2}} |A|^{\frac{(n-m+1)\beta}{2}-1} e^{-\mathrm{tr}(\frac{\beta}{2}\Sigma^{-1}A)}$$

- Joint eigenvalue density (well defined for any $\beta > 0$):

$$(\det \Sigma)^{-\frac{n\beta}{2}} \prod_{i=1}^{m} \lambda_i^{\frac{(n-m+1)\beta}{2}-1} \prod_{j<k} |\lambda_k - \lambda_j|^\beta \cdot {}_0F_0^{(\beta)}(-\tfrac{\beta}{2}\Lambda, \Sigma^{-1}),$$

where

$${}_0F_0^{(\beta)}(X, Y) = \sum_{k=0}^{\infty} \sum_{\kappa \vdash k} \frac{1}{k!} \cdot \frac{C_\kappa^{(\beta)}(X) C_\kappa^{(\beta)}(Y)}{C_\kappa^{(\beta)}(I)}$$

Introducing the Beta-Wishart Matrix

- Def. (for any $\beta > 0$): A matrix with joint eigenvalue density

$$(\det \Sigma)^{-\frac{n\beta}{2}} \prod_{i=1}^{m} \lambda_i^{\frac{(n-m+1)\beta}{2}-1} \prod_{j<k} |\lambda_k - \lambda_j|^{\beta} \cdot {}_0F_0^{(\beta)}(-\tfrac{\beta}{2}\Lambda, \Sigma^{-1})$$

Introducing the Beta-Wishart Matrix

- Def. (for any $\beta > 0$): A matrix with joint eigenvalue density

$$(\det \Sigma)^{-\frac{n\beta}{2}} \prod_{i=1}^{m} \lambda_i^{\frac{(n-m+1)\beta}{2}-1} \prod_{j<k} |\lambda_k - \lambda_j|^{\beta} \cdot {}_0F_0^{(\beta)}(-\tfrac{\beta}{2}\Lambda, \Sigma^{-1})$$

- Extends the Beta-Laguerre model (Dumitriu/Edelman)
- Extends the Beta-Jacobi model (Dumitriu/K)
- All one needs to derive distributions of λ_i's, det, tr, etc.

$\lambda_{\max}$ of a Beta-Wishart Matrix

- Joint eigenvalue density

$$(\det \Sigma)^{-\frac{n\beta}{2}} \prod_{i=1}^{m} \lambda_i^{\frac{(n-m+1)\beta}{2}-1} \prod_{j<k} |\lambda_k - \lambda_j|^{\beta} \cdot {}_0F_0^{(\beta)}\left(-\frac{\beta}{2}\Lambda, \Sigma^{-1}\right)$$

$\lambda_{\max}$ of a Beta-Wishart Matrix

- Joint eigenvalue density

$$(\det \Sigma)^{-\frac{n\beta}{2}} \prod_{i=1}^{m} \lambda_i^{\frac{(n-m+1)\beta}{2}-1} \prod_{j<k} |\lambda_k - \lambda_j|^\beta \cdot {}_0F_0^{(\beta)}(-\tfrac{\beta}{2}\Lambda, \Sigma^{-1})$$

- Distribution of $\lambda_{\max}$

$$P(\lambda_{\max} < x) = c \qquad \int_{[0,x]^m} \prod_{i=1}^{m} \lambda_i^{\frac{(n-m+1)\beta}{2}-1} \prod_{j<k} |\lambda_k - \lambda_j|^\beta {}_0F_0^{(\beta)}(-\tfrac{\beta}{2}\Lambda, \Sigma^{-1}) d\Lambda$$

$\lambda_{\max}$ of a Beta-Wishart Matrix

- Joint eigenvalue density

$$(\det \Sigma)^{-\frac{n\beta}{2}} \prod_{i=1}^{m} \lambda_i^{\frac{(n-m+1)\beta}{2}-1} \prod_{j<k} |\lambda_k - \lambda_j|^{\beta} \cdot {}_0F_0^{(\beta)}(-\tfrac{\beta}{2}\Lambda, \Sigma^{-1})$$

- Distribution of $\lambda_{\max}$

$$\begin{aligned} P(\lambda_{\max} < x) = c \quad & \int_{[0,x]^m} \prod_{i=1}^{m} \lambda_i^{\frac{(n-m+1)\beta}{2}-1} \prod_{j<k} |\lambda_k - \lambda_j|^{\beta} {}_0F_0^{(\beta)}(-\tfrac{\beta}{2}\Lambda, \Sigma^{-1}) d\Lambda \\ = c x^{\frac{mn\beta}{2}} & \int_{[0,1]^m} \prod_{i=1}^{m} \lambda_i^{\frac{(n-m+1)\beta}{2}-1} \prod_{j<k} |\lambda_k - \lambda_j|^{\beta} {}_0F_0^{(\beta)}(\Lambda, -\tfrac{x\beta}{2}\Sigma^{-1}) d\Lambda \end{aligned}$$

$\lambda_{\max}$ of a Beta-Wishart Matrix

- Joint eigenvalue density

$$(\det \Sigma)^{-\frac{n\beta}{2}} \prod_{i=1}^{m} \lambda_i^{\frac{(n-m+1)\beta}{2}-1} \prod_{j<k} |\lambda_k - \lambda_j|^{\beta} \cdot {}_0F_0^{(\beta)}(-\tfrac{\beta}{2}\Lambda, \Sigma^{-1})$$

- Distribution of $\lambda_{\max}$

$$P(\lambda_{\max} < x) = c \int_{[0,x]^m} \prod_{i=1}^{m} \lambda_i^{\frac{(n-m+1)\beta}{2}-1} \prod_{j<k} |\lambda_k - \lambda_j|^{\beta} {}_0F_0^{(\beta)}(-\tfrac{\beta}{2}\Lambda, \Sigma^{-1}) d\Lambda$$

$$= c x^{\frac{mn\beta}{2}} \int_{[0,1]^m} \prod_{i=1}^{m} \lambda_i^{\frac{(n-m+1)\beta}{2}-1} \prod_{j<k} |\lambda_k - \lambda_j|^{\beta} {}_0F_0^{(\beta)}(\Lambda, -\tfrac{x\beta}{2}\Sigma^{-1}) d\Lambda$$

$$= c' x^{\frac{mn\beta}{2}} {}_1F_1^{(\beta)}(\tfrac{n\beta}{2}; \tfrac{(n+m-1)\beta}{2}; -\tfrac{x\beta}{2}\Sigma^{-1}) d\Lambda$$

It all comes down to computing ${}_pF_q^{(\beta)}$

$$ {}_pF_q^{(\beta)}(a; b; X, Y) = \sum_{k=0}^{\infty} \sum_{\kappa \vdash k} c_\kappa(a, b) \frac{C_\kappa^{(\beta)}(X) C_\kappa^{(\beta)}(Y)}{C_\kappa^{(\beta)}(I)} $$

- No alternative expressions known, except for $\beta = 2$
- Approximate by truncating for $k \leq M$; still very hard

Previous Best Algorithm for ${}_pF_q$ on a 5×5 matrix

Electronic Transactions on Numerical Analysis.
Volume 11, pp. 121-130, 2000.

ISSN 1068-9613.

ETNA
Kent State University
etna@mcs.kent.edu

APPROXIMATION OF HYPERGEOMETRIC FUNCTIONS WITH MATRICIAL ARGUMENT THROUGH THEIR DEVELOPMENT IN SERIES OF ZONAL POLYNOMIALS*

R. GUTIÉRREZ†, J. RODRIGUEZ‡, AND A. J. SÁEZ§

Previous Best Algorithm for ${}_pF_q$ on a 5×5 matrix

Electronic Transactions on Numerical Analysis.
Volume 11, pp. 121-130, 2000.

ISSN 1068-9613.

ETNA
Kent State University
etna@mcs.kent.edu

APPROXIMATION OF HYPERGEOMETRIC FUNCTIONS WITH MATRICIAL ARGUMENT THROUGH THEIR DEVELOPMENT IN SERIES OF ZONAL POLYNOMIALS*

R. GUTIÉRREZ†, J. RODRIGUEZ‡, AND A. J. SÁEZ§

more time would be needed. (We spent about 8 days to obtain the 627 zonal polynomials of degree 20 with a 350 MHz Pentium II processor.)

Previous Best Algorithm for ${}_pF_q$ on a 5×5 matrix

Electronic Transactions on Numerical Analysis.
Volume 11, pp. 121-130, 2000.

ISSN 1068-9613.

ETNA
Kent State University
etna@mcs.kent.edu

APPROXIMATION OF HYPERGEOMETRIC FUNCTIONS WITH MATRICIAL ARGUMENT THROUGH THEIR DEVELOPMENT IN SERIES OF ZONAL POLYNOMIALS*

R. GUTIÉRREZ†, J. RODRIGUEZ‡, AND A. J. SÁEZ§

more time would be needed. (We spent about 8 days to obtain the 627 zonal polynomials of degree 20 with a 350 MHz Pentium II processor.)

- New result (2.33 GHz Pentium):

$>>$ tic; mhg(20, 2, [ ], [ ], [1 : 5]/15), toc
ans = 2.71828182845905
elapsed_time = 0.03100000000000

Computing ${}_pF_q$ efficiently

$$ {}_pF_q^{(\beta)}(\cdot;\cdot;X) = \sum_{k=0}^{\infty} \sum_{\kappa \vdash k} a_\kappa \cdot C_\kappa^{(\beta)}(x_1, \ldots, x_n) $$

- $C_\kappa^{(\beta)}(X) = C_\kappa^{(\beta)}(x_1, \ldots, x_n) =$ Jack function
- orthogonal basis of $\Pi(x_1, \ldots, x_n)$
- Indexed by partitions κ:
  Leading terms: x_1, x_1^2, $x_1 x_2$, x_1^3, $x_1^2 x_2$, $x_1 x_2 x_3$, ...
- Even a single $C_\kappa^{(\alpha)}$ very hard—sum of exp many terms

Naive Cost of Jacks ($\beta = 2$; $C_\kappa^{(1)} = s_\kappa$)

Degree	Partition κ	s_κ	Number of terms
1	(1)	$x_1 + \cdots + x_n$	$\mathcal{O}(n)$
2	(2)	$\sum_{i \le j} x_i x_j$	$\mathcal{O}(n^2)$
2	$(1,1)$	$\sum_{i < j} x_i x_j$	$\mathcal{O}(n^2)$
3	$(1,1,1)$	$\sum_{i<j<k} x_i x_j x_k$	$\mathcal{O}(n^3)$
$\lvert\kappa\rvert$	κ	$\sum x_1^{\kappa_1} \cdots x_n^{\kappa_n}$	$\mathcal{O}\left(n^{\lvert\kappa\rvert}\right)$

- New result: $\mathcal{O}(n)$ each

Computing the Schur Polynomial "Cleverly" ($\beta = 2$)

- s_κ of higher degree "contain" s_λ of lower degree. Redundancy.

Computing the Schur Polynomial "Cleverly" ($\beta = 2$)

- s_κ of higher degree "contain" s_λ of lower degree. Redundancy.
- E.g.:

$$
\begin{aligned}
s_{(1,1)}(x_1, \ldots, x_n) &= \sum_{i<j} x_i x_j \\
&= x_1 x_2 + (x_1 + x_2)x_3 + \cdots + (x_1 + \cdots + x_{n-1})x_n
\end{aligned}
$$

Computing the Schur Polynomial “Cleverly” ($\beta = 2$)

- s_κ of higher degree “contain” s_λ of lower degree. Redundancy.
- E.g.:

$$\begin{aligned} s_{(1,1)}(x_1, \ldots, x_n) &= \sum_{i<j} x_i x_j \\ &= x_1 x_2 + (x_1 + x_2) x_3 + \cdots + (x_1 + \cdots + x_{n-1}) x_n \end{aligned}$$

- In general:

$$s_\kappa(x_1, x_2, \ldots, x_n) = \sum_{\lambda < \kappa} s_\lambda(x_1, x_2, \ldots, x_{n-1}) \cdot x_n^{|\kappa| - |\lambda|}$$

- Connection with representation theory:
  - s_κ are the irreducible characters of $\mathrm{GL}_n(\mathbb{C})$
  - the characters of $\mathrm{GL}_{n-1}(\mathbb{C})$ induce those of $\mathrm{GL}_n(\mathbb{C})$
- Result long known (Macdonald), but missed for 40 years

Computing the Schur Polynomial "Cleverly"

- Example: $s_{(1,1)}(x_1, \ldots, x_n)$

$$= \sum_{i<j} x_i x_j \qquad \left(\sim n^2 \text{ operations } \right)$$

$$= \underbrace{x_1}_{s_1} x_2 + (\underbrace{x_1 + x_2}_{s_2}) x_3 + (\underbrace{x_1 + x_2 + x_3}_{s_3}) x_4 + \cdots + (\underbrace{x_1 + \cdots + x_{n-1}}_{s_{n-1}}) x_n$$

Computing the Schur Polynomial "Cleverly"

- Example: $s_{(1,1)}(x_1, \ldots, x_n)$

$$= \sum_{i<j} x_i x_j \qquad \left(\sim n^2 \text{ operations } \right)$$

$$= \underbrace{x_1}_{s_1} x_2 + (\underbrace{x_1 + x_2}_{s_2}) x_3 + (\underbrace{x_1 + x_2 + x_3}_{s_3}) x_4 + \cdots + (\underbrace{x_1 + \cdots + x_{n-1}}_{s_{n-1}}) x_n$$

- New cost: $3n - 2$ instead of n^2
- Generalizes to all κ:
  Cost of $s_\kappa(x_1, \ldots, x_n)$ goes down from $\mathcal{O}(n^{|\kappa|})$ to $\mathcal{O}(N_\kappa n)$
- This trick made all the difference and works for all $\beta > 0$
- For $\beta = 2$ it gets better:
  We can get rid of $N_\kappa \equiv \{\#\lambda | \lambda < \kappa\}$

Analogy with the FFT

- Idea: $(\mathrm{DFT})_{ij}$—char. of $\mathbb{Z}/n\mathbb{Z}$ $\longleftrightarrow$ s_λ—char. of $\mathrm{GL}_n(\mathbb{C})$
- Our main identity

$$s_\kappa(x_1, x_2, \ldots, x_n) = \sum_{\lambda < \kappa} s_\lambda(x_1, x_2, \ldots, x_{n-1}) \cdot x_n^{|\kappa|-|\lambda|}$$

Analogy with the FFT

- Idea: $(\mathrm{DFT})_{ij}$—char. of $\mathbb{Z}/n\mathbb{Z}$ $\longleftrightarrow$ s_λ—char. of $\mathrm{GL}_n(\mathbb{C})$
- Our main identity

$$s_\kappa(x_1, x_2, \ldots, x_n) = \sum_{\lambda < \kappa} s_\lambda(x_1, x_2, \ldots, x_{n-1}) \cdot x_n^{|\kappa|-|\lambda|}$$

in matrix form is $\mathcal{S}_n = \mathcal{S}_{n-1} \cdot Y_n(x_n)$, where

Analogy with the FFT

- Idea: $(\mathrm{DFT})_{ij}$—char. of $\mathbb{Z}/n\mathbb{Z}$ $\longleftrightarrow$ s_λ—char. of $\mathrm{GL}_n(\mathbb{C})$
- Our main identity

$$s_\kappa(x_1, x_2, \ldots, x_n) = \sum_{\lambda < \kappa} s_\lambda(x_1, x_2, \ldots, x_{n-1}) \cdot x_n^{|\kappa|-|\lambda|}$$

in matrix form is $\mathcal{S}_n = \mathcal{S}_{n-1} \cdot Y_n(x_n)$, where

$$Y = \left[\begin{array}{cccc|cccc|cccc|cccc} 1 & x & x^2 & x^3 & & & & & & & & & & & & \\ & 1 & x & x^2 & x & x^2 & x^3 & x^4 & & & & & & & & \\ & & 1 & x & & x & x^2 & x^3 & x^2 & x^3 & x^4 & x^5 & & & & \\ & & & 1 & & & x & x^2 & & x & x^2 & x^3 & x^3 & x^4 & x^5 & x^6 \end{array}\right]$$

$$\begin{bmatrix} 1 & x & x^2 & x^3 \\ & 1 & x & x^2 \\ & & 1 & x \\ & & & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & -x & & \\ & 1 & -x & \\ & & 1 & -x \\ & & & 1 \end{bmatrix}$$

- Matrix-vector multipl. now costs $\mathcal{O}(n)$ instead of $\mathcal{O}(nN_\kappa)$

Our New Fast Algorithm

- $A \equiv$ lower shift matrix ($a_{i+1,i} = 1$); $B = A^T$
- Structure of Y_n :

$$\begin{aligned}
U_n(x_n) &= I_{(N+1)^{n-1}} + x_n(A \otimes B_{n-1}) + \cdots + x_n^N(A^N \otimes B_{n-1}^N) \\
&= \left(I_{(N+1)^{n-1}} - x_n(A \otimes B_{n-1})\right)^{-1}, \\
C_n(x_n) &= U_n(x_n)K_{n-1}(x_n), \\
K_n(x_n) &= I_{N+1} \otimes C_n(x_n), \\
B_n &= B_{n-1} \otimes I_{N+1} = B \otimes I_{(N+1)^{n-2}}, \\
Q_n(x_n) &= \left(I_{(N+1)^{n-1}} \mid x_n B_n \mid \ \ldots \ \mid x_n^N B_n^N\right) \\
Y_n &= Q_n(x_n)K_n(x_n)
\end{aligned}$$

Our New Fast Algorithm

- $A \equiv$ lower shift matrix ($a_{i+1,i} = 1$); $B = A^T$
- Structure of Y_n :

$$
\begin{aligned}
U_n(x_n) &= I_{(N+1)^{n-1}} + x_n(A \otimes B_{n-1}) + \cdots + x_n^N(A^N \otimes B_{n-1}^N) \\
&= \left(I_{(N+1)^{n-1}} - x_n(A \otimes B_{n-1})\right)^{-1}, \\
C_n(x_n) &= U_n(x_n)K_{n-1}(x_n), \\
K_n(x_n) &= I_{N+1} \otimes C_n(x_n), \\
B_n &= B_{n-1} \otimes I_{N+1} = B \otimes I_{(N+1)^{n-2}}, \\
Q_n(x_n) &= \left(I_{(N+1)^{n-1}} \mid x_n B_n \mid \ldots \mid x_n^N B_n^N\right) \\
Y_n &= Q_n(x_n)K_n(x_n)
\end{aligned}
$$

- New algorithm:

```
for i=n:-1:1
    for all λ, |λ| ≤ M, in reverse lex. order
        s_λ = s_λ + s_λ(i) x_n
```

$$s_\lambda = s_\lambda + s_{\lambda^{(i)}} x_n$$

(where $\lambda^{(i)} \equiv (\lambda_1, \ldots, \lambda_i - 1, \ldots, \lambda_n)$)

- Final cost: $\mathcal{O}(n)$ per each s_λ, optimal

Open Problem: Can we extend the FFT idea to $\beta \neq 1$?

- How does one multiply quickly by the matrix ($\alpha \equiv 2/\beta$):

$$A = \begin{bmatrix} 1 & 1 & \frac{\alpha+1}{2} & \frac{(\alpha+1)(2\alpha+1)}{6} \\ & 1 & 1 & \frac{\alpha+1}{2} \\ & & 1 & 1 \\ & & & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & -\alpha & & \\ & 1 & -\alpha & \\ & & 1 & -\alpha \\ & & & 1 \end{bmatrix}^{-1/\alpha}$$

Open Problem: Beta-Wishart Matrix Model?

- *A* Beta-Wishart, $A = X^*X, \quad X \sim N^{(\beta)}(0, I_n \otimes \Sigma)$
- To get $\lambda_i(X^*X)$ right suffices to get $\sigma_i(X)$ right

$$\begin{bmatrix} G & G & G & G \\ G & G & G & G \\ G & G & G & G \\ G & G & G & G \end{bmatrix} \times \Sigma^{1/2}$$

Open Problem: Beta-Wishart Matrix Model 2?

- A Beta-Wishart, $A = X^*X, \quad X \sim N^{(\beta)}(0, I_n \otimes \Sigma)$
- To get $\lambda_i(X^*X)$ right suffices to get $\sigma_i(X)$ right

$$H \times \begin{bmatrix} \chi_{4\beta} & G & G & G \\ 0 & G & G & G \\ 0 & G & G & G \\ 0 & G & G & G \end{bmatrix} \times \Sigma^{1/2}$$

Open Problem: Beta-Wishart Matrix Model 3?

- A Beta-Wishart, $A = X^*X, \quad X \sim N^{(\beta)}(0, I_n \otimes \Sigma)$
- To get $\lambda_i(X^*X)$ right suffices to get $\sigma_i(X)$ right

$$Q_4^{(\beta)} \begin{bmatrix} \chi_{4\beta} & \chi_{3\beta} & 0 & 0 \\ 0 & \chi_{3\beta} & \chi_{2\beta} & 0 \\ 0 & 0 & \chi_{2\beta} & \chi_{\beta} \\ 0 & 0 & 0 & \chi_{\beta} \end{bmatrix} \times Q_3^{(\beta)} \times \Sigma^{1/2}$$

Conclusions

- For small matrices takes less than a second to get the distributions
- Papers and software at: http://www4.ncsu.edu/~pskoev
- Impact on important applications:
  - 3D target classification
  - Genomics
  - Wireless communications

Future work

- New algorithms based on saddle point approximations
- Automatic convergence detection
- FFT generalization to zonal polynomials
- Tracy–Widom finite inference